UNIVERSIDADE FEDERAL DE SANTA CATARINA - CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Análise e Comparação de Técnicas de Classificação de Tópicos de Aplicativos Móveis Criados com App Inventor

Pablo Daniel Riveros Strapasson

Florianópolis, 2023

Universidade Federal de Santa Catarina
Departamento de Informática e Estatística

Análise e Comparação de Técnicas de Classificação de Tópicos de Aplicativos Móveis Criados com App Inventor

Trabalho de Conclusão de Curso de Graduação em Ciências da Computação, do Departamento de Informática e Estatística, do Centro Tecnológico da Universidade Federal de Santa Catarina, requisito parcial à obtenção do título de Bacharel em Ciências da Computação.

Autor: Pablo Daniel Riveros Strapasson
Orientadora: Prof.ª Dr.ª rer. nat. Christiane Gresse von Wangenheim, PMP

Florianópolis
2023

AGRADECIMENTOS

Agradeço primeiramente à Universidade Federal de Santa Catarina por oferecer um ensino superior público e de qualidade. Agradeço também a todo o corpo docente e técnicos administrativos do curso de Ciência da Computação, que colaboraram com a minha formação profissional.

Quero agradecer especialmente aos meus pais, Claudio e Gloria, e a minha irmã, Sabrina, por todo apoio, incentivo, amor e carinho nessa caminhada pela universidade. Aos meus amigos Alan, Guilherme, Yasmin, Julia, Camila, Adriane e Cristiane, agradeço pela amizade e companheirismo, vocês tornaram essa jornada mais divertida. A Renata agradeço por todo amor, carinho e companheirismo.

Agradeço também a professora Christiane por ter aceitado e dedicado seu tempo e esforço em me orientar e auxiliar na realização deste trabalho, e a todos que de alguma forma contribuíram com esse trabalho e com a minha formação acadêmica.

Resumo

A criatividade é considerada uma das principais habilidades do século XXI. E é muito importante estimular o desenvolvimento da criatividade também na Educação Básica, uma vez que facilita e colabora para um aprendizado mais contextualizado e significativo. Embora seja associada principalmente a artes, a criatividade também pode ser desenvolvida no ensino de computação, como, por exemplo, com o desenvolvimento de aplicativos móveis com App Inventor. Assim, nesse contexto, se faz necessário um modelo de avaliação da criatividade do artefato criado pelo aluno como resultado da aprendizagem. Dentro da criatividade há o aspecto da novidade, em que uma das principais características é a originalidade. No contexto de aplicativos móveis a originalidade pode ser analisada, por exemplo, a partir dos tópicos abordados pelo aplicativo. Essa classificação de tópicos pode ser automatizada a partir de textos presentes nos artefatos computacionais adotando algoritmos de *machine learning*. Atualmente existem diversos algoritmos para este fim, como *Decision Tree, Support Vector Machine*, dentre outros. Portanto este estudo tem por objetivo analisar e comparar o desempenho de várias destas técnicas de classificação de textos para, dessa forma, auxiliar na evolução de modelos de avaliação de originalidade que foquem no conteúdo presente em aplicativos criados com App Inventor. Como resultado, este trabalho apresenta uma comparação de desempenho dos modelos selecionados em diversos experimentos, testando várias alternativas de pré-processamento. Estes resultados podem ser utilizados em pesquisas voltadas à automação da avaliação da criatividade no contexto do ensino de computação e assim contribuir ao desenvolvimento da criatividade dos estudantes na Educação Básica.

Palavras-chave: Criatividade; App Inventor; Tópicos de Aplicativos; Classificação de Texto; *Machine Learning*; Desempenho

Lista de Figuras

Lista de Tabela

Sumário

1. INTRODUÇÃO

1.1 Contextualização

Nos dias atuais a criatividade é considerada uma das principais habilidades e ela é essencial para estimular o desenvolvimento da criatividade também na Educação Básica, uma vez que facilita e colabora para um aprendizado mais contextualizado e significativo (P21, 2019). A criatividade está presente em diversos currículos escolares ao redor do mundo, incluindo o Brasil. Ela também está sendo incluída em avaliações padronizadas como o exame PISA desde 2022 (OECD, 2019).

Por vezes a criatividade é associada ao ramo das artes, tais como música, pintura, cinema, fotografia e literatura. Porém, criatividade é a qualidade de quem tem capacidade, talento e inteligência para criar, inventar ou fazer inovações na área em que atua, podendo essa área ser qualquer uma (CRIATIVIDADE, 2020). Portanto ela também está presente e pode ser desenvolvida como parte da computação como diz a Base Nacional Comum Curricular (BNCC):

> "Exercitar a curiosidade intelectual e recorrer à abordagem própria das ciências, incluindo a investigação, a reflexão, a análise crítica, a imaginação e a criatividade, para investigar causas, elaborar e testar hipóteses, formular e resolver problemas e criar soluções (inclusive tecnológicas) com base nos conhecimentos das diferentes áreas."(MEC, 2018).

Uma das formas para o desenvolvimento da criatividade por meio da computação é o desenvolvimento, por parte dos alunos, de aplicativos, utilizando plataformas como o App Inventor, como parte do ensino. Porém, para guiar o desenvolvimento da criatividade se faz necessário a utilização de alguma forma de avaliação do artefato criado e de *feedback* dos resultados dessa avaliação (MISHRA; HENRIKSEN, 2013). Contudo, avaliar um aspecto complexo como a criatividade não é uma tarefa simples.

A definição de criatividade de um artefato normalmente é definida em termos de novidade, adequação e condensação (ALVES et al., 2020). Dentro do aspecto de novidade a originalidade é uma das principais características, sendo, um artefato, considerado original quando apresenta algo inédito ou surpreendente.

Embora a criatividade seja considerada uma das principais competências do século XXI, e a originalidade uma das suas principais características, ainda existem

poucas pesquisas sobre como avaliar o aspecto de originalidade de um artefato dentro de um contexto educacional, em específico, dentro do ensino de computação. As abordagens existentes (SCAICO et al., 2013)(GAL et al., 2017) (MUSTAFARAJ et al., 2017)(TURBAK et al., 2017) focam na avaliação da originalidade do código fonte. Contudo, a originalidade de um app pode ser analisada de diversos pontos de vista , como, por exemplo, tópicos abordados, design de interface, funcionalidades, dentre outros (ALVES et al., 2020).

Assim, um dos aspectos se refere a avaliar o grau de originalidade de um artefato por meio da classificação dos tópicos abordados pelo aplicativo. Nesse contexto um aplicativo é considerado original se aborda tópicos novos (ou menos comumente abordados) comparado a aplicativos já existentes.

Essa classificação de tópico(s) do app pode ser automatizada utilizando processamento de linguagem natural (*natural language processing - NLP*). Nesse contexto, os textos utilizados para a classificação dos tópicos abordados por um aplicativo são extraídos de todos os elementos textuais presentes no aplicativo, seja em componentes de interface como em outros tipos de componentes.

Uma solução para a classificação dos tópicos, abordados por um aplicativo, por meio dos textos presentes neste artefato é o uso de técnicas de classificação de tópicos. Existem diversas técnicas de modelagem e de classificação de tópicos, como, por exemplo, *Latent Dirichlet Allocation (LDA), Decision Trees, Random Forest, Logistic Regression, Naive Bayes e K Nearest Neighbour (KNN)*, entre outras (BIANCHI, 2020). Cada uma destas técnicas apresenta diferentes abordagens para a descoberta/classificação de tópicos, como o uso de técnicas Bayesianas, no caso do LDA (SCARPA, 2017), e técnicas de *Machine Learning* e análise estatística para a classificação de tópicos, como no caso de *Decision Trees* e *Random Forest*. Também existem métodos como *Bidirectional Encoder Representations from Transformers (BERT)* (DEVLIN, 2018) que utilizam técnicas de *transformers* para a descoberta de relações contextuais entre as palavras de um texto.

Assim, o objetivo deste trabalho é, sistematicamente, analisar e comparar o desempenho de diferentes técnicas aplicadas aos elementos textuais de apps criados com App Inventor. E com base nos resultados definir qual das técnicas apresenta um melhor desempenho na classificação de tópicos dos aplicativos com base nos textos apresentados nas telas e em elementos de texto de componentes presentes nos

aplicativos analisados para possibilitar a automação da avaliação da originalidade de aplicativos em termos de tópicos

1.2 Objetivos

Objetivo Geral

O objetivo deste trabalho é sistematicamente analisar e comparar o desempenho de técnicas de modelagem de tópicos, como LDA, e diferentes técnicas de *Machine Learning* para classificação de tópicos, como *Random Forest, Logistic Regression, Naive Bayes* e *Support Vector Machine*, e transformers, como BERT, aplicados a dados (elementos textuais) extraídos de apps em português criados com a plataforma App Inventor.

Objetivos Específicos

Os objetivos específicos são:
O1. Analisar os conceitos em relação a análise das técnicas de *machine learning* no contexto da avaliação da originalidade/novidade de aplicativos Android/App Inventor na Educação Básica.
O2. Levantar o estado da arte em relação às técnicas de classificação de tópicos sendo utilizadas para classificação de apps pelos tópicos.
O3. Treinar, analisar e comparar técnicas de classificação de tópicos aplicados a apps do App Inventor.

1.3 Metodologia

Do ponto de vista da natureza da pesquisa, este trabalho se classifica como uma pesquisa aplicada (ou tecnológica), que tem por objetivo gerar produtos inéditos, com finalidades imediatas, com base em conhecimentos prévios. Para desenvolver os resultados esperados no presente projeto, será adotada uma combinação de metodologias de pesquisa de acordo com o respectivo objetivo/meta a ser alcançado.

Etapa 1 - Síntese da fundamentação teórica

É realizada uma análise de literatura de conceitos básicos em relação a originalidade/novidade, características de apps do App Inventor, como também de conceitos de *Machine learning* e *Deep Learning*.

Atividade 1.1: Analisar conceitos de originalidade e detecção da novidade de tópico em apps do App Inventor.

Atividade 1.2: Sintetizar conceitos de *Machine Learning/Deep Learning*

Etapa 2 - Levantamento do estado da Arte

Analisar o estado da arte em relação a abordagens existentes voltadas à modelagem/classificação de tópicos em apps. Para esta etapa é utilizada a técnica de mapeamento sistemático de literatura (PETERSEN et al., 2015).

Atividade 2.1: Definir o protocolo da revisão sistemática.

Atividade 2.2: Executar a busca.

Atividade 2.3: Analisar e interpretar as informações extraídas.

Etapa 3 - Análise e comparação das técnicas

Seguindo a abordagem de estudo de caso (WOHLIN et al., 2012)(YIN, 2017), é definido um estudo de caso para sistematicamente avaliar técnicas de classificação de tópico e compará-las. Isto envolve tanto a definição da avaliação, quanto a definição de medidas adequadas de desempenho. A preparação do conjunto de dados é feita com base em apps disponíveis na galeria do App Inventor e desenvolvidas pela iniciativa computação na escola/INCoD/INE/UFSC. Durante a execução da análise são aplicadas várias técnicas de classificação de tópicos, medindo o desempenho de cada técnica de classificação para que ao final sejam comparados os resultados obtidos.

Atividade 3.1: Definição da avaliação.

Atividade 3.2: Preparação do conjunto(s) de dados.

Atividade 3.3: Execução/Treinamento de várias técnicas.

Atividade 3.4: Avaliação e comparação de desempenho dos modelos.

1.4 Estrutura do documento

Este trabalho está estruturado da seguinte forma. No capítulo 2 é apresentada a fundamentação teórica sobre originalidade, tópicos de aplicativos, plataforma App

Inventor de desenvolvimento de apps, e modelos de classificação de tópicos. O capítulo 3 apresenta o estado da arte da área de classificação de apps de acordo com o conteúdo dos aplicativos. No capítulo 4 é apresentada uma proposta de solução para o problema de classificação de aplicativos criados no App Inventor, bem como a realização de experimentos e avaliação dos modelos testados. Já o capítulo 5 apresenta as considerações finais do trabalho, indicando quais atividades foram realizadas, os objetivos alcançados e possíveis trabalhos futuros dentro do escopo deste trabalho.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados conceitos relevantes para o desenvolvimento deste trabalho. Inicialmente é feita uma análise sobre o conceito de originalidade em termos de tópicos de aplicativos criados com App Inventor. Para isso é revisado o conceito de originalidade, seguido da revisão do conceito de tópicos de aplicativos, apresentando tópicos de apps usados pela Google Play e Apple Store. É também apresentada a ferramenta App Inventor, com foco nos elementos textuais usados por ela. Com relação a classificação de tópicos, são apresentados também os principais conceitos além das técnicas *machine learning* tipicamente utilizadas para essa função, assim como métricas geralmente utilizadas para a avaliação do desempenho dessas técnicas.

2.1 Originalidade de Tópicos de Apps com App Inventor

2.1.1 Criatividade/Originalidade

O conceito de criatividade, diz respeito à qualidade, ao talento e à inteligência de se criar, inovar e inventar e pode-se ver o crescente aumento na importância da criatividade no século XXI, sendo considerada, cada vez mais, uma das principais competências para a atualidade (CAVALLO et al., 2016). E por mais que, comumente ela seja associada às artes em geral, como, por exemplo, pintura e música, ela também pode ser associada à ciência e tecnologia, como, o desenvolvimento de aplicativos móveis, uma vez que envolvem *design* e inovação (BENNET et al., 2011).

A criatividade pode ser analisada da perspectiva de quatro aspectos (RHODES, 1961): pessoa, processo, ambiente e produto, e das interações entre elas. A perspectiva da pessoa criativa engloba aspectos como personalidade, hábitos, intelecto, temperamento, crenças, comportamentos, dentre outras particularidades pertencentes ao indivíduo. O aspecto do processo versa sobre a percepção, a motivação e o aprendizado vivenciado. O ambiente aponta sobre a relação entre os indivíduos e o seu ambiente. Já o produto criativo diz respeito ao produto final do processo criativo, que de alguma forma é comunicado aos outros, seja qual for a

linguagem escolhida, palavras, quadros, sons, produtos tecnológicos, dentre outros (GARCÊS, 2013).

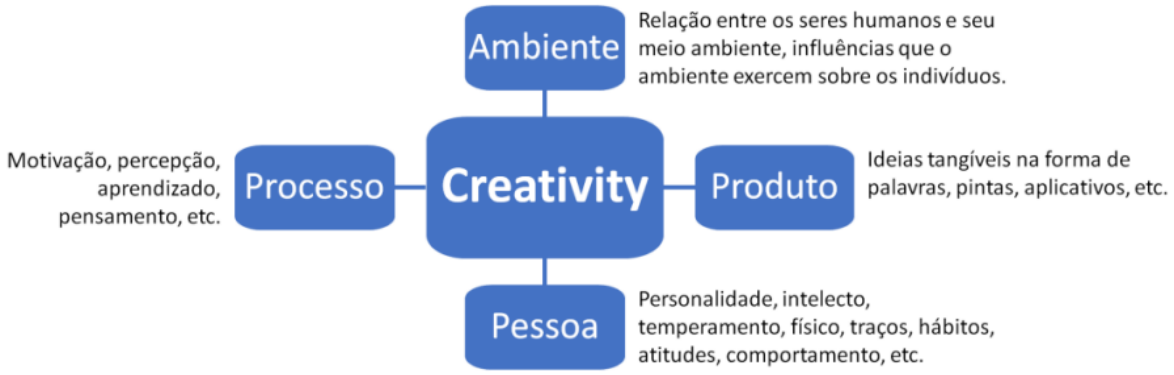

Figura 1. Perspectivas da criatividade (ALVES et al., 2021).

Atualmente não existe um modelo de avaliação consolidado que categorize um produto como sendo criativo ou não. No entanto, é possível destacar algumas características que geralmente são analisadas para classificar um produto como sendo criativo ou não. Essas características podem ser divididas em três dimensões: Novidade, Adequação e Condensação (ALVES et al., 2020) (Figura 2).

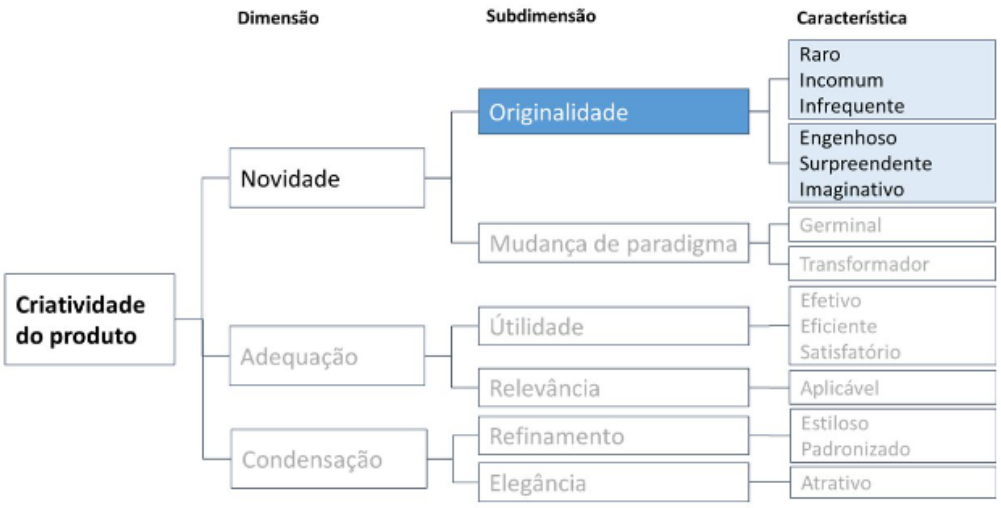

Figura 2. Características de um produto criativo (ALVES et al., 2020)

A dimensão de novidade se refere aos produtos considerados originais, que quebram e mudam paradigmas estabelecidos. A originalidade, por sua vez, diz

respeito ao incomum, surpreendente, imaginativo, ao que é pouco frequentemente visto. É um dos componentes associados ao conceito de novidade, que, por sua vez, é uma das dimensões da criatividade. A mudança de paradigma refere-se aos produtos, que de alguma forma, revolucionaram uma área, produtos que mudaram ou redefiniram as concepções básicas dentro de uma teoria científica bem estabelecida. Assim como a originalidade, a mudança de paradigma também é associada à dimensão de Novidade.

Dentro da dimensão de adequação são consideradas as características de utilidade e relevância de um produto. A utilidade mede até que ponto um produto atende de forma efetiva, eficiente e satisfatória às necessidades de uma situação ou problema para qual o produto foi feito. Já a relevância do produto se refere ao quão aplicável um produto é em seu contexto de atuação.

Na dimensão de condensação são avaliados aspectos referentes ao quão bem feito um produto é, unindo conceitos polares de simplicidade e complexidade. O refinamento diz respeito ao quão estilosa é a solução encontrada para o projeto e design de um produto, que às vezes pode já existir, simplificando uma solução para uma tarefa muito complexa. Já a elegância é uma medida para o quão atrativo e elegante é o design de um produto (ALVES et al., 2020).

Ao se analisar um aplicativo, diversas dimensões, além do código fonte, podem ser consideradas para classificar o aplicativo como criativo ou não, tais como o objetivo do aplicativo, design do aplicativo como um todo e o escopo de funcionalidades e de conteúdo abordado, que pode ser traduzido em tópicos (Figura 3).

Figura 3. Dimensões de um aplicativo (ALVES et al., 2020)

Com base no conceito de originalidade é possível definir algumas características que poderiam ser utilizadas para julgar um tópico de aplicativo como sendo original ou não. Um escopo de aplicativo que seja considerado incomum, imaginativo, engenhoso, que é surpreendente pode ser considerado como um escopo original, portanto ao se definir um tópico para a classificação deste aplicativo deve-se criar um tópico tão incomum, imaginativo, engenhoso e surpreendente quanto o escopo do produto, criando um tópico totalmente original.

2.1.2 Tópicos de aplicativos

Todo aplicativo é projetado e desenvolvido para resolver algum problema, facilitar a vida dos usuários na realização de alguma tarefa em alguma área da sua vida ou para entretenimento dos usuários. Estas áreas podem ser traduzidas em tópicos, ou categorias, como, por exemplo, alimentação, esportes, finanças, educação, etc. Um aplicativo pode ser classificado, por exemplo, de acordo com o conteúdo abordado por este app.

Observa-se que atualmente a maioria dos celulares e dispositivos móveis vendidos no Brasil possuem, predominantemente, os sistemas operacionais Android

(91,6%), desenvolvido pela Google, ou iOS (2%), desenvolvido pela Apple (CARDOSO, 2020), sendo a produção de aplicativos voltada principalmente para estes dois sistemas. Esses aplicativos são tipicamente disponibilizados por meio das lojas online (Google Play Store e Apple App Store). Ambas as lojas de apps definem um conjunto de categorias para a classificação de cada app. As categorias utilizadas pela Google Play Store para a classificação de aplicativos submetidos à loja e suas definições são apresentadas na Tabela 1.

Tabela 1. Categorias e definições do Google Play Store (GOOGLE, 2022)

CATEGORIA	Definição	Exemplos
Art and Design	Aplicativos que fornecem ferramentas para arte, design e criação gráfica.	Cadernos de desenhos, livros de colorir, ferramentas de pintura e ferramentas de arte e design.
Augmented Reality	Aplicativos que dão suporte à realidade aumentada	Calculadoras gráficas, mapas 3D, medidores de distância 3D.
Auto and Vehicles	Aplicativos relacionados a automóveis, bicicletas ou outros tipos de veículos.	Compra de automóveis, seguros, comparação de preços, segurança na estrada, notícias e opiniões relacionadas.
Beauty	Aplicativos que fornecem informações sobre beleza.	Tutoriais de maquiagem, ferramentas de transformação visual, cabelos, compras de produtos de beleza e simuladores de maquiagem.
Books and Reference	Aplicativos que fornecem interatividade para conteúdo tradicionalmente oferecido na forma impressa.	Leitores de livros, livros de referência, livros didáticos, dicionários, dicionários de sinônimos e wikis.
Business	Aplicativos que auxiliam na administração de uma empresa ou fornecem um meio de colaboração, busca de emprego, edição ou compartilhamento de conteúdo.	Editor/leitor de documentos, rastreamento de pacote, computador remoto, gerenciamento de e-mails, busca de emprego.
Comics	Aplicativos que fornecem informações ou conteúdo de quadrinhos/manga.	Personagens, títulos.
Communication	Aplicativos que fornecem suporte para comunicação.	Mensagens, bate-papo/mensagens instantâneas, telefones, agenda de endereços, navegadores, gerenciamento de chamadas.
Dating	Aplicativos de namoro e relacionamentos.	Encontre sua cara metade, paquera, construção de um relacionamento, conhecer novas pessoas, à procura de um amor.
Daydream	Aplicativos que dão suporte à realidade virtual.	Jogos de Realidade Virtual, Viagens VR.
Education	Aplicativos que fornecem uma experiência de aprendizagem interativa sobre uma habilidade ou assunto específico.	Preparações para exames, materiais de estudo, vocabulário, jogos educativos, aprendizado de idiomas.
Entertainment	Aplicativos que são projetados para entreter e informar o usuário, podem ter áudio, visual ou outro conteúdo de entretenimento.	Streaming de vídeo, filmes, TV e entretenimento interativo.
Events	Aplicativos para compra e venda de ingressos de eventos.	Ingressos para shows, eventos esportivos e cinema, além de revenda de ingressos.
Finance	Aplicativos que realizam transações financeiras ou auxiliam o usuário em questões financeiras comerciais ou pessoais.	Transações bancárias, pagamentos, localizadores de caixas eletrônicos, notícias financeiras, seguros, tributos, portfólio/compra e venda de ações e calculadoras de gorjeta.
Food and Drink	Aplicativos que fornecem recomendações, instruções, receitas ou críticas relacionadas à preparação, consumo ou revisão de alimentos ou bebidas..	Receitas, restaurantes, guias de nutrição, descoberta e degustação de vinhos e receitas de bebidas.
Games	Aplicativos de jogos, que oferecem atividades interativas para um ou vários jogadores para fins de entretenimento.	Ação, arcade, aventura, cartas, cassino, casual, corrida, educacional, esportes, estratégia, jogos

		de palavras, música, perguntas e respostas, quebra-cabeça, RPG, simulação, tabuleiro.
Health and Fitness	Aplicativos relacionados a uma vida saudável, incluindo treino, dieta, nutrição, gerenciamento de estresse e condicionamento físico.	Condicionamento físico pessoal, acompanhamento de exercícios, dicas nutricionais e de dietas, saúde e segurança etc.
House and Home	Aplicativos que fornecem informações sobre habitação, design de interiores e questões domésticas.	Pesquisa de casa e apartamento, melhorias no lar, decoração interna, hipotecas e imobiliárias.
Kids	Aplicativos voltados para o público infantil.	Jogos apropriados para a idade, histórias interativas, materiais educativos, revistas.
Libraries and Demo	Bibliotecas de softwares e demonstrações técnicas.	Bibliotecas de software e demonstrações técnicas.
Lifestyle	Aplicativos relacionados a um assunto específico de interesse geral.	Guias de estilo, dicas e ferramentas para planejamento de festas e casamentos e guias práticos.
Maps and Navigation	Aplicativos que fornecem informações geográficas sobre locais ou para ajudar um usuário a viajar para um local físico.	Ferramentas de navegação, GPS, mapeamento, ferramentas de trânsito e transporte público.
Medical	Aplicativos focados em educação médica, gerenciamento de informações ou referência de saúde para pacientes ou profissionais de saúde.	Referências clínicas e de medicamentos, calculadoras, manuais de serviços de saúde, publicações e notícias médicas.
Music and Audio	Aplicativos para descobrir, ouvir, gravar, executar ou compor músicas ou sons.	Serviços de música, rádios e players de música.
News and Magazines	Aplicativos que oferecem revistas ou conteúdo de notícias.	Jornais, agregadores de notícias, revistas e blogs.
Parenting	Aplicativos que fornecem informações ou suporte sobre paternidade/maternidade.	Gravidez, monitoramento e cuidado infantil e cuidados com o bebê.
Personalization	Aplicativos que dão suporte a customização de aparência e funções do sistema operacional do aparelho.	Planos de fundo, planos de fundo interativos, tela inicial, tela de bloqueio, toques.
Photography	Aplicativos que auxiliam na captura, edição, gerenciamento, armazenamento ou compartilhamento de fotos.	Câmeras, ferramentas de edição de fotos, gerenciamento e compartilhamento de fotos.
Productivity	Aplicativos que tornam um processo ou tarefa específica mais organizado ou eficiente.	Bloco de notas, lista de tarefas, teclado, impressão, calendário, backup, calculadora e conversão.
Shopping	Aplicativos que suportam a compra de bens de consumo ou melhoram/apoiam a experiência de compra.	Compras on-line, leilões, cupons, comparação de preços, listas de compras, avaliações de produtos.
Social	Aplicativos que contribuem para o desenvolvimento da comunidade.	Redes sociais e check-in.
Sports	Aplicativos relacionados a atividades esportivas profissionais, amadoras, universitárias ou recreativas.	Notícias e comentários esportivos, acompanhamento de placares, fantasy games e cobertura de jogos.
Tools/Utilities	Aplicativos que permitem ao usuário resolver um problema ou concluir uma tarefa específica.	Ferramentas para dispositivos Android.
Travel and Local	Aplicativos que auxiliam o usuário em qualquer aspecto de viagem ou local, como planejamento, compra ou rastreamento.	Ferramentas para reservas de viagem, transporte compartilhado, chamada de táxis, guias de cidades, informações sobre empresas locais, ferramentas para organização de viagens e reservas de excursões.
Video Players & Editors	Aplicativos que auxiliam na captura, edição, gerenciamento, armazenamento ou compartilhamento de vídeos.	Players de vídeo, editores de vídeo e armazenamento de mídia.
Watch apps	Aplicativos para relógios inteligentes.	Monitoramento de saúde cardíaca e exercícios físicos, gravador de sono.
Watch faces	Aplicativos de customização de relógios inteligentes.	Personalização de plano de fundo.
Weather	Aplicativos que fornecem previsões, alertas e informações relacionadas às condições meteorológicas.	Informações meteorológicas.

Por outro lado, a Tabela 2 apresenta as categorias utilizadas para a classificação de apps iOS na Apple App Store.

Tabela 2. Categorias e definições da Apple App Store (APPLE, 2022)

CATEGORIA	Definição	Exemplos
Apple Watch Apps	Aplicativos para relógios inteligentes.	Monitoramento de saúde cardíaca e exercícios físicos, personalização de plano de fundo.
AR Apps	Aplicativos que dão suporte a realidade aumentada.	Calculadoras gráficas, mapas 3D, medidores de distância 3D.
Books	Aplicativos que fornecem ampla interatividade para conteúdos tradicionalmente oferecidos em formato impresso.	Histórias, quadrinhos, eReaders, livros de mesa de centro, romances gráficos.
Business	Aplicativos que auxiliam na administração de uma empresa ou fornecem um meio de colaboração, busca de emprego, edição ou compartilhamento de conteúdo.	Gerenciamento de documentos (PDFs, digitalização, visualização/edição de arquivos), telefonia VoIP, ditado, desktop remoto, recursos de busca de emprego, gerenciamento de recursos do cliente, colaboração, planejamento de recursos empresariais, ponto de venda.
Developer Tools	Aplicativos que fornecem ferramentas para desenvolvimento, gerenciamento e distribuição de aplicativos.	Codificação, teste, depuração, gerenciamento de fluxo de trabalho, edição de texto e código..
Education	Aplicativos que fornecem uma experiência de aprendizagem interativa sobre uma habilidade ou assunto específico.	Aritmética, alfabeto, escrita, aprendizagem precoce e educação especial, sistema solar, vocabulário, cores, aprendizagem de idiomas, preparação para testes padronizados, geografia, portais escolares, treinamento de animais de estimação, astronomia, artesanato.
Entertainment	Aplicativos interativos projetados para entreter e informar o usuário, podem ter áudio, visual ou outro conteúdo de entretenimento.	Televisão, filmes, segundas telas, fã-clubes, teatro, toques, manipulação de voz, serviços de bilheteria, criação artística.
Finance	Aplicativos que realizam transações financeiras ou auxiliam o usuário em questões financeiras comerciais ou pessoais.	Gestão financeira pessoal, mobile banking, investimento, lembretes de contas, orçamentos, gestão de dívidas, impostos, finanças de pequenas empresas, seguros.
Food & Drink	Aplicativos que fornecem recomendações, instruções, receitas ou críticas relacionadas à preparação, consumo ou revisão de alimentos ou bebidas.	Coleções de receitas, guias de culinária, resenhas de restaurantes, chefs/receitas de celebridades, alergia alimentar e dietética, resenhas de álcool, guias de cervejarias, cozinha internacional.
Games	Aplicativos de jogos, que oferecem atividades interativas para um ou vários jogadores para fins de entretenimento.	Jogos de ação, aventura, tabuleiro, cartão, música, quebra-cabeça, corrida, role playing, simulação, esportes, estratégia.
Graphics & Design	Aplicativos que fornecem ferramentas para arte, design e criação gráfica.	Design gráfico vetorial, edição de imagem, desenho e ilustração.
Health & Fitness	Aplicativos relacionados a uma vida saudável, incluindo treino, gerenciamento de estresse, condicionamento físico e atividades recreativas.	Yoga, diagramas musculares, acompanhamento de exercícios, corrida, ciclismo, gerenciamento de estresse, gravidez, meditação, perda de peso, pilates, acupuntura/acupressão, medicina oriental/chinesa.
Kids	Aplicativos projetados para crianças de até 11 anos.	Jogos apropriados para a idade, histórias interativas, materiais educativos, revistas.
Lifestyle	Aplicativos relacionados a um assunto específico ou serviço de interesse geral.	Artesanato, hobbies, paternidade, moda, imoveis, melhoria da casa.
Magazines & Newspapers	Aplicativos que oferecem assinaturas com renovação automática de conteúdos de revistas ou jornais.	Jornais, revistas e outros periódicos recorrentes.

Medical	Aplicativos focados em educação médica, gerenciamento de informações ou referência de saúde para pacientes ou profissionais de saúde.	Guia esquelético, guia muscular, anatomia, manutenção de registros médicos, doenças, referência de sintomas, dispositivos complementares (pressão arterial, pulso e assim por diante), rastreamento de saúde.
Music	Aplicativos para descobrir, ouvir, gravar, executar ou compor músicas ou sons.	Criação musical, rádio, educação, edição de som, descoberta musical, composição, escrita de letras, bandas e artistas de gravação, videoclipes e shows, ingressos para shows.
Navigation	Aplicativos que fornecem informações para ajudar um usuário a viajar para um local físico..	Assistência à condução, assistência a pé, mapas topográficos, marítimos, registos/assistência de pilotos, marés oceânicas, atlas rodoviário, localizadores de combustível, mapas de transporte público.
News	Aplicativos que oferecem informações sobre eventos atuais ou desenvolvimentos em áreas de interesse, como política, entretenimento, negócios, ciências, tecnologia, etc.	Televisão, vídeo, rádio ou canais de notícias ou programas online, leitores de RSS.
Photo & Video	Aplicativos que auxiliam na captura, edição, gerenciamento, armazenamento ou compartilhamento de fotos e vídeos.	Captura, edição, efeitos especiais, compartilhamento, imagem, impressão, criação de cartões comemorativos, manuais.
Productivity	Aplicativos que tornam um processo ou tarefa específica mais organizado ou eficiente.	Gerenciamento de tarefas, gerenciamento de calendário, tradução, anotações, impressão, gerenciamento de senhas, armazenamento em nuvem, clientes de e-mail, geradores de fluxogramas, ditado de áudio, simulação, visualização de dados.
Reference	Aplicativos que auxiliam o usuário a acessar ou recuperar informações.	Atlas, dicionário, tesauro, citações, enciclopédia, pesquisa geral, animais, direito, religião, instruções, política.
Safari Extensions	Aplicativos que oferecem extensões para ajudar a aprimorar e personalizar a experiência de navegação no navegador Safari.	Marcadores de conteúdo, gerenciadores de senhas, bloqueadores de anúncios, localizadores de economia.
Shopping	Aplicativos que suportam a compra de bens de consumo ou melhoram/apoiam a experiência de compra.	Comércio, mercado, cupom, revisão de produtos, aplicativos com Apple Pay.
Social Networking	Aplicativos que conectam pessoas por meio de texto, voz, foto ou vídeo e contribuem para o desenvolvimento da comunidade.	Conexões interpessoais, mensagens de texto, mensagens de voz, comunicação por vídeo, compartilhamento de fotos e vídeos, namoro, blogs, comunidades de interesse especial, aplicativos complementares para serviços de redes sociais tradicionais.
Sports	Aplicativos relacionados a atividades esportivas profissionais, amadoras, universitárias ou recreativas.	Companheiros de esportes de fantasia, equipes/conferências universitárias, equipes/ligas profissionais, atletas, rastreadores de pontuação, instrução, notícias esportivas.
Travel	Aplicativos que auxiliam o usuário em qualquer aspecto da viagem, como planejamento, compra ou rastreamento.	Rastreamento de voos, relógios multi-horário, guias da cidade, compras de hotéis/aluguel de carros/tarifas aéreas, planejamento de férias, transporte público, recompensas de viagem.
Utilities	Aplicativos que permitem ao usuário resolver um problema ou concluir uma tarefa específica.	Calculadoras (padrão, ponta, financeira), relógios, medição, hora, navegação na web, lanternas, bloqueios de tela, scanners de código de barras, ferramentas de conversão de unidades, gerenciamento de senhas, controles remotos.
Weather	Aplicativos que fornecem previsões, alertas e informações relacionadas às condições meteorológicas.	Radar, previsão, tempestades, marés, mau tempo, clima local.

Comparando as categorias usadas em ambas as lojas de aplicativos móveis, (Tabela 3), é possível notar que a Play Store possui uma variedade maior de categorias (38), oferecendo aos usuários uma classificação mais específica do

conteúdo dos apps, enquanto a App Store da Apple apresenta uma classificação mais genérica dos aplicativos, possuindo menos categorias (29), para a classificação dos aplicativos mantidos em sua loja.

Mesmo observando quantidades diferentes de categorias, observa-se que ambas as classificações são bem parecidas, apresentando categorias iguais, como por exemplo, *Business* e *Weather*. Outros mesmo usando termos diferentes se referem também às mesmas categorias (como p.ex. *Art and Design* e *Graphics & Design*). Há ainda categorias como *Books*, *Reference*, *News*, *Magazines & Newspapers*, que na Apple Store são categorias distintas, e que na Play Store elas são combinadas em categorias únicas (*Books & Reference* e *News & Magazines*).

Tabela 3. Comparação entre categorias do Google Play Store e Apple App Store

GOOGLE PLAY STORE	APPLE APP STORE
Art and Design	Graphics & Design
Augmented Reality	AR Apps
Auto and Vehicles	-
Beauty	-
Books and Reference	Books, Reference
Business	Business
Comics	-
Communication	-
Dating	-
Daydream	-
Education	Education
Entertainment	Entertainment
Events	-
Finance	Finance
Food and Drink	Food & Drink
Games	Games
Health and Fitness	Health & Fitness
House and Home	-
Kids	Kids
Libraries and Demo	-
Lifestyle	Lifestyle
Maps and Navigation	Navigation
Medical	Medical
Music and Audio	Music
News and Magazines	News, Magazines & Newspapers
Parenting	-
Personalization	-
Photography	Photo & Video
Productivity	Productivity

Shopping	Shopping
Social	Social Networking
Sports	Sports
Tools/Utilities	Utilities
Travel and Local	Travel
Video Players & Editors	Photo & Video
Watch apps	Apple Watch Apps
Watch faces	Apple Watch Apps
Weather	Weather
-	Safari Extensions
-	Developer Tools

Ainda, existem algumas categorias que são abordadas apenas em uma das lojas, como, por exemplo, *Parenting* e *Beauty*, que são presentes na Google Play, e *Safari Extensions* e *Developer Tools*, que estão presentes apenas na Apple App Store.

A indicação de categorias é feita pelo próprio desenvolvedor do aplicativo ao submeter o app à loja de aplicativos, portanto não há nenhuma ferramenta que classifique os aplicativos submetidos de forma automática. Porém ambas empresas disponibilizam um guia para auxiliar os desenvolvedores a classificar seus aplicativos (GOOGLE, 2022)(APPLE, 2022). Na Google Play um aplicativo pode ter até 5 categorias, chamadas de *tag* no site. São especificados passos para adicionar novas tags, bem como uma tabela com as tags existentes com exemplos de aplicativos para cada *tag*. Na Apple App Store cada aplicativo submetido pode ter até 2 categorias associadas, uma categoria primária e outra secundária, além de dicas para classificar aplicativos de games ou voltados para o público infantil.

2.1.3 App Inventor

App Inventor (https://appinventor.mit.edu/) é uma plataforma de desenvolvimento de aplicativos para celulares Android desenvolvida pelo Google e atualmente é mantida pelo *Massachusetts Institute of Tecnology (MIT)*. Sua proposta é facilitar o desenvolvimento de aplicativos para pessoas que não possuem familiaridade com linguagens de programação (PATTON et al., 2019). A ferramenta implementa o conceito de *drag-and-drop* para a codificação de aplicativos, em que o usuário arrasta e conecta blocos de código para criar a lógica do aplicativo. Isso torna a criação de aplicativos mais acessível, uma vez que não é necessário que o usuário

aprenda e compreenda a sintaxe de linguagens de programação textuais. A parte lógica do aplicativo é desenvolvida na área de blocos, onde o usuário, por meio de blocos, que representam funções, condicionais, laços e blocos específicos de cada componente, constrói as funcionalidades do app (Figura 4).

Figura 4. Área de blocos do App Inventor

Assim, como a parte lógica do aplicativo, a ferramenta App Inventor possibilita o desenvolvimento de toda a interface gráfica do app por meio da interface web da ferramenta (Figura 5).

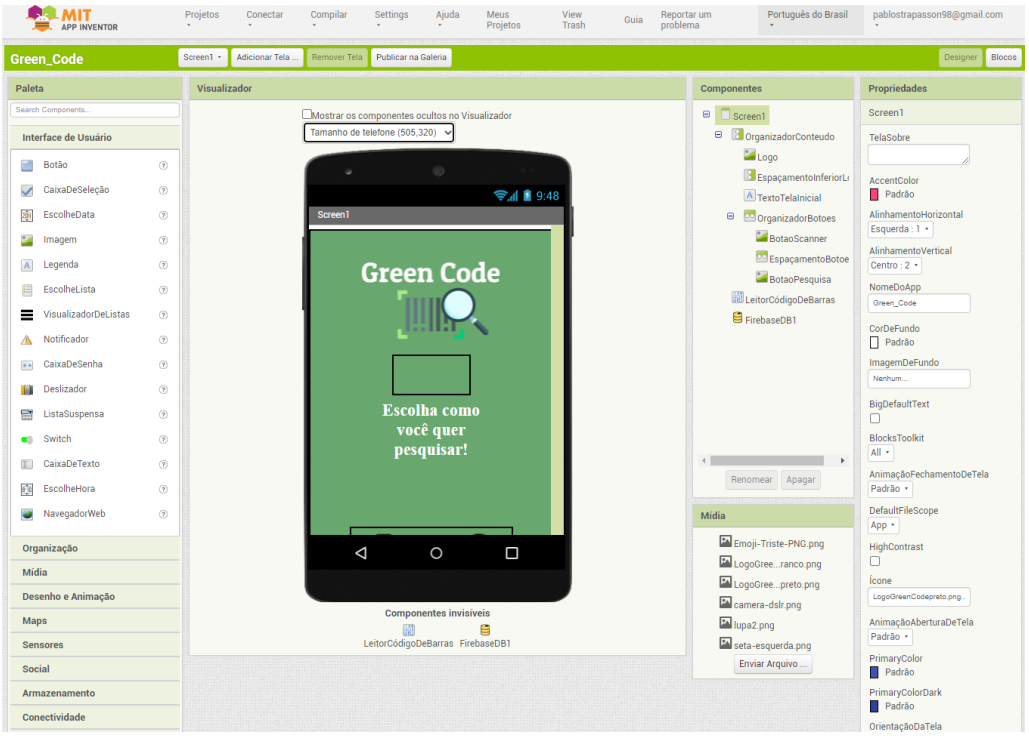

Figura 5. Área de desenvolvimento de interface gráfica do App Inventor

O desenvolvimento da interface gráfica é feita na área *Designer* onde são configurados todos os componentes de interface, tais como, botões, caixas de textos, labels, imagens e organizadores de componentes. Nessa área, também são adicionados e configurados componentes de conectividade e recursos do celular, tais como, banco de dados, câmera, sensores, mapas, conexão bluetooth, dentre outros.

Os aplicativos desenvolvidos no App Inventor podem ser compartilhados, de forma gratuita na *Gallery* do App Inventor, onde ficam disponíveis para *download* por qualquer usuário. Na submissão de um aplicativo na *gallery* do App Inventor pode-se incluir algumas informações sobre o app a ser publicado, tais como, descrição do app, créditos ao(s) seu(s) autor(es), informações extras e uma imagem relacionada ao aplicativo (Figura 6).

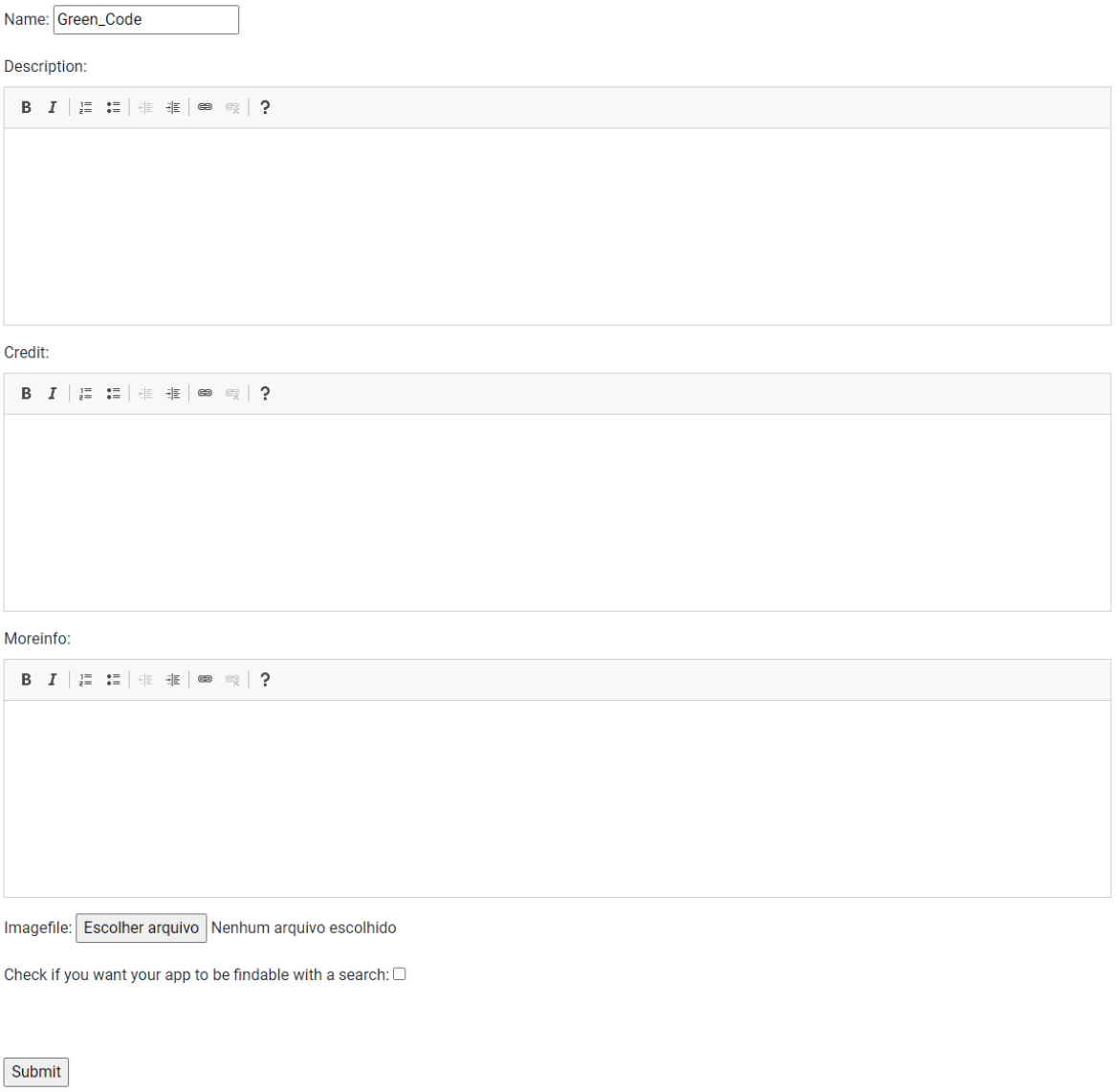

Figura 6. Área de submissão de aplicativos do App Inventor

Mesmo tendo a possibilidade de incluir uma descrição do aplicativo, observa-se que muitos apps não possuem ou possuem uma descrição muito carente de detalhes (Figura 7). Isso acontece muito provavelmente pelo fato de que vários desses apps são compartilhados por crianças/jovens que estão aprendendo a desenvolver aplicativos e não possuem uma noção de documentação adequada. Outro fator é a falta de um campo para a indicação de categoria do app, não havendo nem mesmo uma sugestão para indicação de uma categoria dentro do campo de descrição do app. A falta de uma revisão dos apps compartilhados, também contribui a essa questão, aumentando a quantidade de apps na galeria sem ao menos uma descrição.

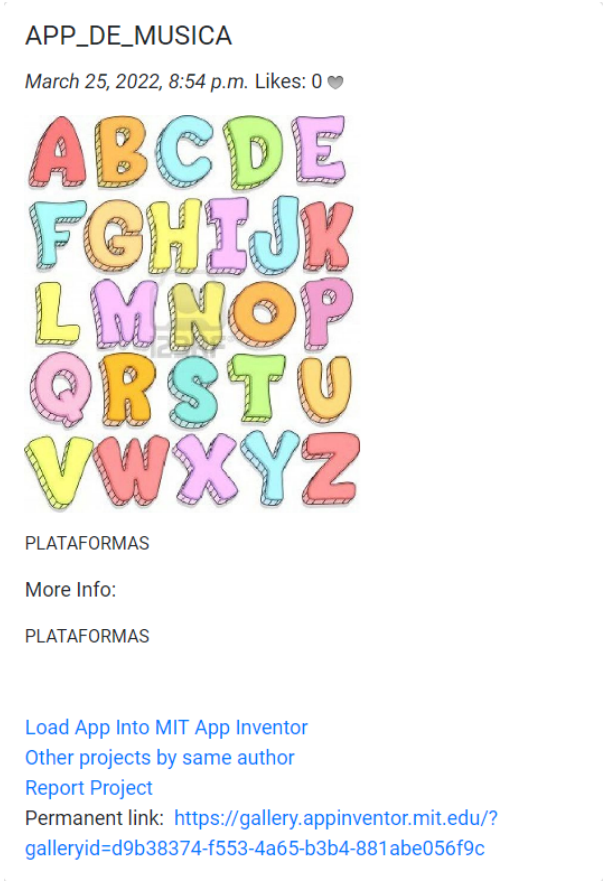

Figura 7. Exemplos de aplicativo e descrição na *gallery* do App inventor

Esse processo de compartilhamento de aplicativos do App Inventor difere dos processos de disponibilização de apps dentro das lojas convencionais de aplicativos, como Google Play Store e Apple App Store, onde na submissão de um aplicativo são exigidas algumas informações sobre o aplicativo, como, descrição breve do app (de até 80 caracteres), uma descrição completa (de até 4000 caracteres), um idioma, e uma categoria para o aplicativo. Nessas lojas os apps também passam por um processo de revisão do app para, se aprovado, ser disponibilizado na loja virtual. Além disso, usuários podem acrescentar revisões referentes aos apps nas lojas, o que novamente não é possível na App Inventor Gallery.

2.1.3.1 Elementos textuais em apps no App Inventor

Dentro do App Inventor existem diversos componentes de interface que possuem elementos textuais. Assim, uma alternativa para o problema de classificação de aplicativos do App Inventor, pode ser, extrair os elementos textuais encontrados

nos componentes presentes no aplicativo. Após isso analisar e identificar, por meio de técnicas de classificação de tópicos, os tópicos abordados por esse aplicativo, e assim, determinar a categoria mais adequada para este app. A Tabela 4 mostra todos os componentes presentes no App Inventor que possuem elementos de textos, bem como quais propriedades desses elementos são textuais.

Tabela 4: Elementos textuais de apps do App Inventor

Componente?	Elemento?	Propriedade
SCREEN		
		Title
		AboutScreen
DESIGNER		
User Interface	Button	Text
	Checkbox	Text
	DatePicker	Text
	Label	Text
	ListPicker	Text e ElementsFromString
	PasswordTextBox	Text e Hint
	Spinner	ElementsFromString e Prompt
	Switch	Text
	TextBox	Text e Hint
	TimePicker	Text
Media	ImagePicker	Text
Maps	Marker	Description
Social	ContactPicker	Text
	EmailPicker	Text
	PhoneNumberPicker	Text
	Texting	Message
Connectivity	ActivityStarter	ResultName
BLOCKS		
Text	String	Text

A Tabela 5 apresenta um exemplo de textos extraídos de elementos textuais de um app, Green Code, desenvolvido no App Inventor, para identificar e listar ingredientes e aditivos utilizados na fabricação de alimentos.

Tabela 5. Exemplo de textos extraídos do aplicativo Green Code

App	Elemento	Texto (extraído de cada componente)	Automaticamente extraído
Green Code	Label text	Escolha como você quer pesquisar!, Digite o código de barras ou o nome do produto!, Nome:, Código:, Ingredientes:, Aditivos:, Informações sobre os Aditivos:, Não encontramos seu produto! Tente novamente,	Corante Caramelo IV Acidulante ácido fosfórico Aroma natural/extrato de noz de cola Corante Caramelo IV: É um corante artificial que confere coloração característica encontrada nos refrigerantes. É produzido através da mistura de açúcar com ácidos e amônia, na presença de elevada temperatura e pressão. Esse corante forma subprodutos que geram danos à saúde, um deles é conhecido como 4-MI (4-Metilimidazol) que pode causar câncer de pulmão, fígado,
	Button text	Pesquisar, Nova Consulta, Nova Consulta	

	String	/codigo, /nome, /ingredientes, /lista_aditivos, None None None None /aditivos None None None /codigo /nome /ingredientes /aditivos /lista_aditivos, None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None None /codigo /nome /ingredientes /aditivos /lista_aditivos	tireoide e leucemia. Acidulante ácido fosfórico: Produzido quimicamente, é responsável pela diminuição do pH do produto e regulação do teor de açúcar, destacando o paladar ácido e também operando como conservante. Esse ácido pode causar danos à saúde dos ossos. Aroma natural/extrato de noz de cola: É um aditivo aromatizante retirado diretamente de matérias-primas naturais, normalmente na forma de óleos. São utilizados para promover ou acentuar o sabor do produto. Não apresenta perigo de toxidade nesse tipo de aromatizante. None None None None /aditivos None None None /codigo /nome /ingredientes /aditivos /lista_aditivos Screen1 Escolha como você quer pesquisar! None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None None /codigo /nome /ingredientes /aditivos /lista_aditivos Screen2 Digite o código de barras ou o nome do produto! ex: 7894900011531 Pesquisar Screen3 Nome: Código: Ingredientes: Aditivos: Informações sobre os Aditivos: Nova Consulta Screen4 Não encontramos seu produto! Tente novamente Nova Consulta
	TextBox	ex: 7894900011531	

2.2 Classificação de Tópicos

Para a classificação de tópicos utiliza-se o processo de mineração de textos, que consiste na descoberta de conhecimento utilizando técnicas de extração e análise de textos, com o intuito de extrair padrões interessantes e não triviais e conhecimento a partir de documentos de textos (SANTOS, 2019). O processo de mineração de dados pode ser aplicado em diversos contextos, tais como, ferramentas de detecção de spam, identificação de gênero, análise de sentimentos, definição de categorias, classificação de textos em tópicos, dentre outras aplicações (SANTOS, 2019).

A classificação de textos consiste na atividade de categorizar um conjunto de textos extraídos com relação a um conjunto de categorias pré-existentes (Rizzi et al., 2001). Essa classificação pode ser realizada por meio de técnicas supervisionadas ou não-supervisionadas. Técnicas supervisionadas utilizam um conjunto de treinamento com dados já rotulados que são usados para o treinamento com o objetivo de gerar uma função que seja capaz de classificar um dado desconhecido de acordo com categorias previamente definidas. Já em técnicas não-supervisionadas os dados utilizados para o treinamento não são rotulados, portanto o modelo aprende tentando descobrir relações entre os dados para então conseguir classificar outros dados desconhecidos (SILVA, 2021).

Os modelos de classificação de textos podem ser divididos em dois tipos: *single-label*, em que cada documento pode ser associado a somente uma categoria,

ou *multi-label* em que o documento analisado pode ser relacionado a uma ou mais categorias (SANTOS, 2019).

O processo de classificação pode ser dividido em três grandes passos: Coleta de dados, Pré-processamento dos dados e Classificação. A etapa de coleta de dados resume-se na busca de documentos que sejam relevantes ao domínio da aplicação. A etapa de pré-processamento dos dados consiste na aplicação de uma sequência de ações que transformam e filtram o conjunto de dados, extraídos no passo anterior que normalmente estão em linguagem natural, em um conjunto de termos úteis. O processo de pré-processamento é considerado de grande importância para a classificação de textos, também sendo a mais custosa, uma vez que os documentos analisados não são estruturados tendo uma quantidade muito grande de termos para serem processados, ou seja, possui uma alta dimensionalidade de termos extraídos.

Uma forma de otimizar o desempenho nesses modelos é a redução da dimensionalidade de termos sem que se perca as características dos documentos. Essa redução é feita primeiramente por meio da tokenização do conteúdo, nessa etapa as palavras do documento são isoladas em *tokens*. Os *tokens* podem ser formados por palavras simples ou compostas, bem como por um número específico de caracteres. Depois da tokenização do texto os sinais de pontuação, caracteres especiais e números são identificados e eliminados, em seguida os tokens restantes passam por uma conversão das letras maiúsculas para minúsculas (SILVA, 2004). Após isso são identificados termos como advérbios, artigos, preposições, pronomes, dentre outros, que são irrelevantes para a tarefa de classificação, esses termos são chamados de *stopwords* e são eliminados nesta etapa.

Também podem ser realizadas algumas outras tarefas, tais como, redução de afixos, por meio de técnicas como *stemmin*g ou *lemmatization,* ou seleção de termos mais importantes para a representação de cada documento. Essa seleção pode ser feita por meio de análises sintáticas ou semânticas, adotando uma abordagem linguística, ou através da contagem de termos adotando, assim, uma abordagem estatística.

Na técnica de *stemming,* para redução de afixos, as palavras são analisadas individualmente e reduzidas até o seu *stem* que é o radical da palavra, também chamado de raiz da palavra. Essa redução é feita por meio de algumas regras e podem vir a reduzir a palavras gramaticalmente incorretas, mas que ainda possuem valor para a análise. Já na técnica de *lemmatization* as palavras são reduzidas até o

seu *lemma*, sendo que a palavra resultante dessa redução sempre será uma palavra gramaticalmente correta além de levar em conta também o texto ao redor e a classe gramatical da palavra para realizar a redução, resultando numa redução mais informativa do que na técnica de *stemming* (KHYANI et al., 2021).

Outra tarefa que pode ser realizada ainda na etapa de pré-processamento é a classificação das palavras de acordo com a sua função gramatical dentro da frase, para isso pode ser utilizada a técnica de *Part-of-Speech Tagging* que se utiliza da definição da palavra bem como o seu contexto para classificar a palavra de acordo com a sua função semântica. Essa classificação pode ser útil para a diferenciação de palavras iguais que podem ter duas funções/significados diferentes dentro de um texto.

Após a limpeza inicial o conteúdo do documento é transformado em um vetor indexado pelas palavras presente no texto e com os valores de frequência de cada palavra no texto. Esse passo se faz necessário uma vez que os métodos de classificação não conseguem trabalhar com o texto puro. O vetor resultante é chamado de *feature vector* e normalmente possui alta dimensionalidade, o que prejudica o processo de classificação uma vez que aumenta a complexidade dos métodos utilizados e reduz a sua precisão por conta de termos redundantes e irrelevantes para a classificação (SHAH; PATEL, 2016). Uma forma de resolver esse problema é através da aplicação de técnicas de *feature extraction* que buscam reduzir a dimensionalidade do *feature vector* selecionando, através de alguns critérios, os termos mais relevantes para a classificação (ZEBARI et al., 2020).

Após o pré-processamento é realizada a etapa de classificação. Para essa etapa existem várias técnicas para a classificação de tópicos, algumas são apresentadas nas próximas seções. Originalmente diversas técnicas de classificação foram concebidas para realizar uma classificação binária dos dados. Contudo diversos problemas de classificação possuem múltiplas classes para a classificação dos dados. Tendo em vista esse problema algumas estratégias podem ser utilizadas para a conversão de um classificador binário para um classificador de múltiplas classes.

Normalmente as abordagens mais usadas consistem na combinação de classificadores binários, decompondo o problema multiclasse original em vários subproblemas binários, essa abordagem tende a reduzir a complexidade do problema original o que pode, também, favorecer classificadores que apresentam dificuldades

em trabalhar com muitas categorias. Outra possibilidade é a adaptação dos algoritmos de classificação para que consigam trabalhar em um ambiente multiclasse. Essa estratégia nem sempre é possível ou adequada, uma vez que adiciona complexidade ao algoritmo aumentando seu custo computacional (LORENA; CARVALHO, 2008).

2.2.1 Latent Semantic Analysis (LSA)

Latent Semantic Analysis (LSA) (DUMAIS, 2004) é um método de processamento de linguagem natural que faz uso de técnicas de estatística para identificar correlações entre diferentes termos em um documento ou até similaridade entre documentos. O método LSA é baseado no método algébrico de decomposição em valores singulares, ou *singular value decomposition (SVD)*, de uma matriz.

Como primeiro passo, o método LSA constrói uma matriz esparsa onde as linhas são indexadas pelos documentos, ou diferentes pedaços de um documento, caso se esteja analisando apenas um documento, e as colunas pelas palavras. A matriz é então preenchida com os valores da contagem de cada palavra em cada documento ou pedaço de documento. Nesta etapa não são realizadas nenhum tipo de fatoração de termos ou análise morfológica dos termos. Após a construção da matriz de ocorrência é aplicada um método SVD para a fatoração da matriz, sem que ela perca sua estrutura de similaridade entre as colunas. Os documentos podem então ser comparados tomando o produto escalar entre as normatizações dos vetores colunas, que representam os documentos, caso sejam semelhantes o valor será perto de 1, caso contrário, o valor será perto de 0 (WIEMER-HASTINGS, 2004).

2.2.2 Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) (BLEI et al., 2003), é um método estatístico generativo muito utilizado para a modelagem de tópicos. O LDA baseia-se na ideia de que um documento é representado por uma mistura de tópicos latentes, distribuídos a partir de uma distribuição de Dirichlet, e também que cada palavra pode ser

associada a um tópico e cada tópico é caracterizado por uma distribuição de palavras (BLEI et al., 2003).

LDA é um processo imaginário, assume-se que os tópicos são definidos *a priori,* antes da geração de qualquer documento, e consistem em uma coleção fixa de palavras, às quais são associadas probabilidades de acordo com a representatividade dessa palavra dentro do tópico. Enquanto os documentos são formados por palavras, pertencentes a uma distribuição de tópicos, escolhidas aleatoriamente.

O objetivo do algoritmo LDA é inferir a estrutura de tópicos de um documento observando as palavras que o compõem. Inicialmente é definido um número K de tópicos, o algoritmo então inicia percorrendo o documento e atribuindo aleatoriamente as palavras a um dos K tópicos. Dentro dos tópicos todas as palavras do documento possuem uma probabilidade de ocorrer, as palavras com maior probabilidade tendem a co-ocorrer com maior frequência, e portanto são usadas para interpretar semanticamente e nomear o tópico (SOUZA, 2022).

Com base na co-ocorrência das palavras é atribuída ao documento a distribuição de tópicos, cada um com o seu percentual de relevância dentro do documento, e assim os tópicos mais relevantes são utilizados para classificar o documento.

Como o LDA assume que um documento é composto por vários tópicos ele pode ser considerado como um modelo de filiação mista. Essa capacidade de detectar a heterogeneidade dos tópicos presentes em um documento aumenta a generalização dos resultados obtidos do LDA, fazendo com que seu desempenho frente a outros modelos mais simples, baseados na atribuição de um único tópico ao documento, seja superior (KASZUBOWSKI, 2016).

2.2.3 Árvores de Decisão

Árvore de decisão (KOTSIANTIS et al., 2007) é um algoritmo de *machine learning* supervisionado muito difundido no meio, utilizado para classificação e regressão de dados. As árvores de decisão são formadas por nós, que, assim como seu nome indica, são interligados hierarquicamente em forma de árvore através de arestas, que são regras ”se-então”, tendo um nó raiz (*root node*), sendo esse o mais importante, nós de decisão (*decision nodes*), que indicam atributos e com base em

seu valor decisões a serem tomadas durante o processo de classificação, e nós folhas (*leaf nodes*), que são os resultados finais (SACRAMENTO, 2021). No contexto de *machine learning* o nó raiz diz respeito a um atributo da base de dados e nós folhas à classe ou valor resposta (Figura 8).

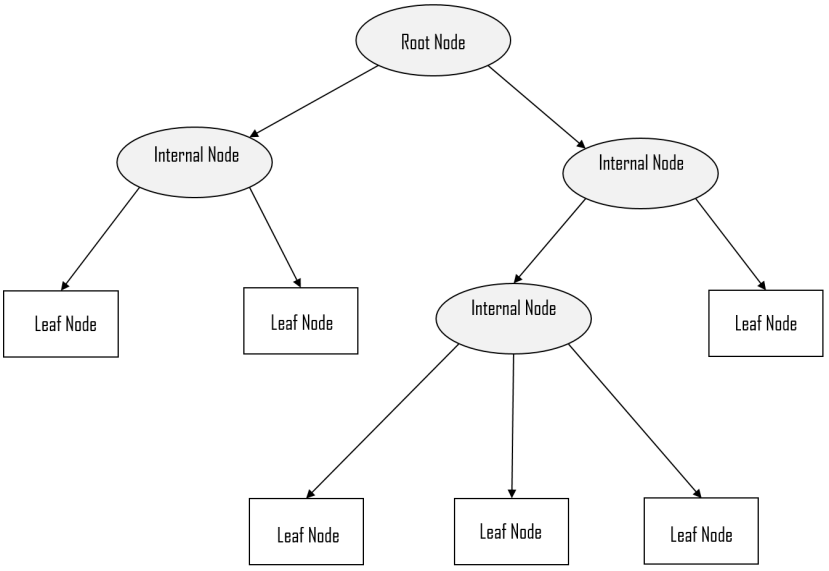

Figura 8. Representação de uma árvore de decisão (HUAWEI, 2021)

Para a construção da árvore de decisão é utilizado o conceito de “dividir-para-conquistar”, nesse caso o espaço do problema é subdividido em espaços menores de acordo com os atributos presentes nesses problemas, para assim criar uma fronteira de decisão (CARVALHO, 2005). Inicialmente define-se um nó raiz, para isso são realizados cálculos de ganho de informação e entropia, o ganho de informação se trata na escolha de um atributo que melhor divide o conjunto de dados, enquanto a entropia diz respeito a desorganização dos dados num sistema. Com base nos resultados desses cálculos é escolhido um atributo para ser o nó raiz da árvore e então o conjunto de dados é dividido de acordo com o valor desse atributo, e assim sucessivamente, estendendo a árvore até que todos os dados sejam de uma mesma classe, configurando assim um nó folha (SACRAMENTO, 2021).

Após a construção da árvore é importante avaliar o modelo, estimando a proporção de acertos e erros, além também de verificar se o modelo não generaliza demais os dados ou se não ocorreu *overfitting*, que se trata do sobre-treinamento do modelo, fazendo com que o modelo seja incapaz de generalizar dados, em ambos os casos o modelo é deficiente em se adaptar a novos dados e situações.

2.2.4 Floresta Aleatória - *Random Forest*

O algoritmo de *Random Forest* (BREIMAN, 2001), ou floresta aleatória em português, é um algoritmo de aprendizagem supervisionado que consiste na combinação de diversos modelos de árvores de decisão para a obtenção de resultados mais precisos apesar do seu maior custo computacional. É muito utilizado em tarefas de classificação e regressão.

Para a construção de uma *random forest*, são criadas diversas árvores de decisão de forma aleatória, assim como sugere o nome. Durante a criação das árvores ao invés de buscar um atributo que melhor subdivide o espaço do problema o algoritmo divide os atributos do espaço desse problema em subconjuntos menores e aleatórios de variáveis. Dentro desse subconjunto é escolhido o melhor atributo para a divisão dos dados que será usado como nó raiz da árvore que está sendo construída, e assim sucessivamente até se ter uma floresta (BREIMAN, 2001).

O resultado gerado por uma *random forest* consiste na agregação de todos os resultados das árvores dentro do modelo. Em problemas de regressão pode se calcular a média de todos os resultados das árvores dentro do modelo, já em problemas de classificação o valor da moda pode ser usado como resultado final. Em outros tipos de problemas pode ser feita uma organização em cascata das árvores onde cada resultado depende do anterior.

Este processo possibilita uma grande diversidade tornando o modelo capaz de boa generalização sem a necessidade da utilização de métodos de redução, como método de poda (RIQUETI et al., 2018).

2.2.4 Regressão Logística - *Logistic Regression*

O método de regressão logística (KLEINBAUM; KLEIN, 2010) utiliza um modelo de regressão para calcular ou prever a probabilidade de algum evento ocorrer, ou seja, de uma variável dependente assumir certo valor de acordo com o valor de variáveis independentes.

Outro uso de um modelo de regressão logística é estimar a probabilidade de um evento ocorrer, para uma observação específica das variáveis independentes, contra a probabilidade desse evento não ocorrer, por exemplo, para as variáveis independentes A = X e B = Y, a probabilidade P do evento E ocorrer contra a probabilidade de E não acontecer é de P/(1 - P). Também pode ser usado para calcular o peso que uma ou mais variáveis independentes têm sobre o valor de probabilidade de um evento ocorrer. Outro cenário para o uso de regressão logística é na classificação de observações, para isso o modelo estima a probabilidade da observação pertencer a uma categoria específica (GONZALEZ, 2018).

A principal característica dos modelos de regressão logística é a natureza categórica e normalmente binária da variável dependente que se quer prever o valor. Ainda existem exemplos onde a variável dependente é categórica porém não binária, nesses casos se existe alguma ordem natural entre os valores da variável dependente diz-se que se trata de uma variável categórica ordinal, caso não haja essa relação, tem-se uma variável categórica nominal (KLEINBAUM; KLEIN, 2010).

2.2.5 Naives Bayes

Naives Bayes (RISH, 2001) é um algoritmo de aprendizado de máquina supervisionado de classificação probabilística baseado no Teorema de Bayes. Uma das principais características do algoritmo de *Naives Bayes* é a suposição de que as variáveis do problema são independentes, não possuindo nenhuma correlação entre si, resultando em um peso igual das variáveis para a determinação do resultado. Por exemplo, se uma fruta é considerada uma maçã por ser vermelha e redonda o algoritmo não leva em conta a correlação entre esses fatores para realizar a classificação (BERRAR, 2018). Outra característica é seu melhor funcionamento quando as variáveis usadas são categóricas, ou seja, palavras, do que numéricas.

O algoritmo funciona determinando primeiramente, e com base em um conjunto de dados de treinamento, uma tabela de probabilidades com a frequência dos preditores com relação às variáveis de saída. Existem ainda algumas variações do algoritmo de *Naives Bayes*, *Bernoulli Naives Bayes, Gaussian Naives Bayes* e *Multinomial Naive Bayes*. *Bernoulli Naives Bayes* é utilizado quando as variáveis apresentam características binárias. *Gaussian Naives Bayes* assume que os dados

são contínuos e obedecem uma distribuição gaussiana, *Multinomial Naive Bayes* utiliza vetores de características para representar os documentos e assume que as classes seguem uma distribuição multinomial. O algoritmo trabalha verificando a frequência das palavras dentro de um texto e com base nessas frequências, atribui esse texto a uma classe (JUAN; NEY, 2002).

2.2.6 K Nearest Neighbour - KNN

O algoritmo de *K Nearest Neighbour* (COVER; HART, 1967) é um algoritmo de aprendizado de máquina muito simples em termos de complexidade computacional, sendo largamente utilizado em tarefas de classificação e de regressão.

O funcionamento do algoritmo é simples e consiste na classificação de um novo registro de acordo com a classificação dos K registros vizinhos mais próximos advindos de um conjunto de treinamento. O algoritmo possui dois argumentos: um número K, que representa o número de vizinhos considerados para a classificação do novo registro, e uma função de cálculo de distância para calcular a distância entre os registros e determinar os vizinhos. O desempenho do algoritmo está diretamente relacionado com a escolha de um número adequado de K, este número ideal depende da natureza dos dados (KATARIA; SINGH, 2013).

KNN é dito como um algoritmo de aprendizagem preguiçoso, uma vez que ao invés de aprender uma função discriminatória a partir dos dados de treinamento o KNN memoriza os dados utilizados no treinamento e com base nessas memória realiza a classificação ou previsão de dados somente se estes tiverem proximidade com os dados de treinamento. Uma vantagem de algoritmos baseados na memorização dos dados é a rápida adaptação do algoritmo através de novos dados de treinamento, contudo a complexidade computacional para a classificação de novos dados cresce de forma linear ao aumento desse conjunto de treinamento. Essa desvantagem, contudo, pode ser minimizada com a utilização de estruturas de dados mais eficientes, tais como kd-trees, e com a utilização de dados com poucas dimensões (CUNNINGHAM; DELANY, 2021).

2.2.7 Support Vector Machine (SVM)

Support Vector Machine (SVM) (MAMMONE et al., 2009) ou máquina de vetor de suporte, em portugues, é um algoritmo de *machine learning* baseado na Teoria de Aprendizado Estatístico, desenvolvido por Vapnik em 1995, podendo ser aplicado em diversas áreas, tais como, classificação de textos, detecção de rostos, aplicações de Bioinformática, dentre outras (LORENA; CARVALHO, 2003).

Originalmente as SVM's são utilizadas para a classificação binária dos dados sendo divididas em SMV's lineares e não-lineares. Nas SVM's lineares são utilizadas quando o conjunto de dados de treinamento são linearmente separáveis, ou seja, os dados podem ser separados em duas classes por meio de uma função linear ótima $Ax + B = 0$. Modelos gerados por SVM's lineares normalmente apresentam problemas quando confrontados com dados mais gerais onde a função linear é incapaz de dividir os dados de forma ótima (LORENA, CARVALHO, 2007). Nesses casos são definidas SVM's não-lineares, que são generalizações de SVM's lineares onde os dados do conjunto de treinamento são mapeados do seu espaço original para um novo espaço com dimensão maior, chamado de espaço de características, onde os dados possuem maior probabilidade de serem linearmente separáveis (MAMMONE et al., 2009).

Para problemas que envolvam a classificação dos dados em mais de duas classes, algumas estratégias podem ser utilizadas para se trabalhar com *support vector machines,* decomposição "um-contra-todos" e decomposição "todos-contra-todos". Na estratégia "um-contra-todos", é criada e associada uma SVM para cada classe existente, a classe associada é dita como "positiva" enquanto as demais como "negativa", assim a classe predita é dada pelo índice da SVM que produz a maior saída. Já na estratégia "todos-contra-todos", sendo k o número de classes, são criadas $k(k-1)/2$ SVM's onde uma classe i é definida como "positiva" e outra j como "negativa" e $i \neq j$. A união desses modelos é dada por um esquema de votação por maioria, cada modelo devolve uma classe como resultado e a classe com maior número de indicações é a solução final da classificação. Outra solução para a união dos modelos é a utilização de um grafo acíclico direcionado, de forma que o problema é decomposto em várias classificações binárias, uma em cada nó do grafo (LORENA; CARVALHO, 2003).

Os modelos de *support vector machine,* em geral, apresentam características atrativas como sua boa capacidade de generalização de seus classificadores; robustez com dados com grandes dimensões, tais como imagens; convexidade da função objetivo, aplicações com SVM's implicam na otimização de uma função quadrática com um único mínimo global; além de uma base teórica bem definida e estabelecida (LORENA; CARVALHO, 2003).

2.2.8 Bidirectional Encoder Representations from Transformers (BERT)

BERT (*Bidirectional Encoder Representations from Transformers*) (DEVLIN, 2018) é um algoritmo pré-treinado de *deep learning* do Google para processamento de linguagem natural, que foi desenvolvido em 2018 e possui código aberto. Uma das inovações do BERT é o fato de ser "profundamente bidirecional" capacitando o modelo a ter um profundo senso sobre o contexto em que as palavras se encontram e, consequentemente, do real significado delas dentro do texto. Para isso o BERT faz uso de um *transformer* que consiste em um mecanismo de atenção que visa apreender as relações contextuais entre as palavras em um documento, a principal diferença para os modelos direcionais é que enquanto esses modelos lêem as palavras sequencialmente o *transform* do BERT lê toda uma sequência de palavras de uma só vez, o que dá maior conhecimento sobre o contexto das palavras de acordo com os seus arredores (DEVLIN et al., 2018).

A arquitetura do BERT é composta de duas partes (Figura 8): um tokentizador/codificador e um decodificador. O tokentizador é responsável pela quebra das palavras em sub-palavras enquanto o codificador fica responsável em produzir diferentes representações do texto de entrada. Nessa parte o BERT é treinado de forma não supervisionada, esse treinamento com dados não rotulados visa adquirir a capacidade de predizer palavras, além de tornar possível o treinamento com dados rotulados. A segunda parte, o decodificador, por sua vez, é treinado de forma supervisionada, ou seja, com dados rotulados, essa técnica de treinamento é chamado de *fine-tuning* permitindo com que o BERT seja preparado para a realização de alguma tarefa específica, tais como, análise de sentimentos, classificação de textos, tradução de documentos, dentre outras (ANCHIÊTA et al, 2021).

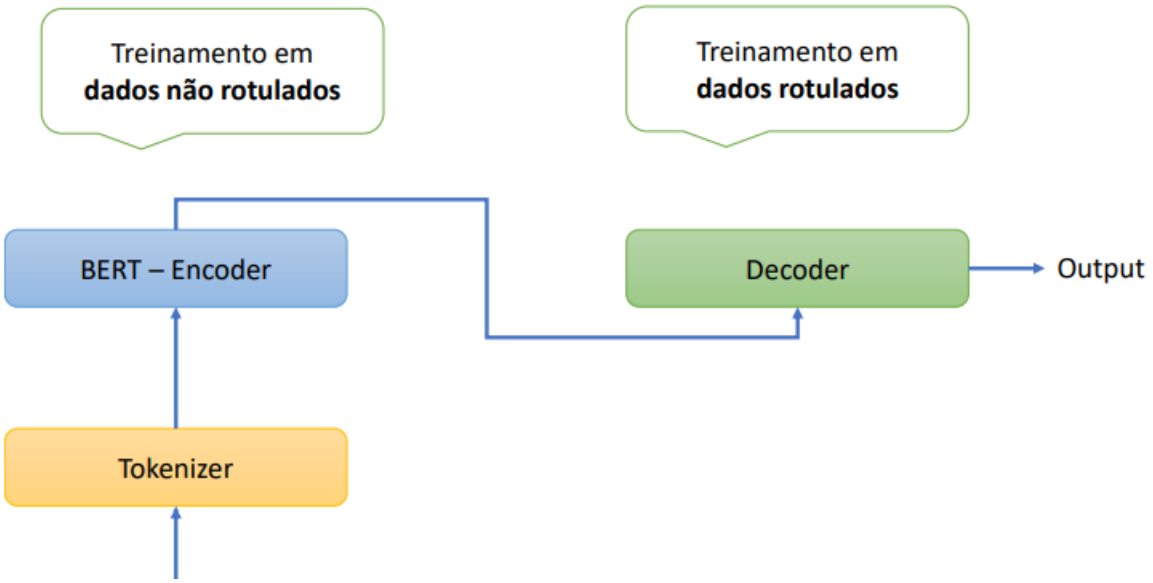

Figura 9. Arquitetura do BERT (ANCHIÊTA, 2021)

2.3 Avaliação de desempenho

O desempenho de técnicas de classificação de texto é tipicamente avaliado por meio de métricas de desempenho, tais como, acurácia, precisão, sensibilidade e pontuação F1. Essas medidas são baseadas nos acertos e erros dos modelos criados. Isto inclui: verdadeiros positivos (*true positives* ou TP), verdadeiros negativos (*true negatives*, ou TN), falsos positivos (*false positives* ou FP), falsos negativos (*false negatives* ou FN).

Acurácia, do inglês accuracy, mede a proporção total de acertos do modelo (SOKOLOVA; LAPALME, 2009). A acurácia é calculada seguindo a fórmula:

$$acurácia = (TP + TN) / (TP + TN + FP + FN)$$

Precisão, do inglês *precision*, mede a proporção de casos que são corretamente classificados (POWERS, 2011). A precisão é calculada seguindo a fórmula:

$$precisão = TP / TP + FP$$

Sensibilidade, ou *recall* no inglês, é a métrica que reflete a proporção entre os casos corretamente classificados a uma categoria e todos os casos que de fato

pertencem a esta categoria (POWERS, 2011). Para calcular a sensibilidade usa-se a fórmula:

$$sensibilidade = TP/(TP + FN)$$

Pontuação F1, ou *F1 score* no inglês consiste na média harmônica entre os valores de precisão e de sensibilidade de um modelo, quanto mais perto de 1 o seu valor, melhor o modelo consegue performar (SOKOLOVA; LAPALME, 2009). Para o cálculo do *F1-score* a seguinte fórmula é utilizada:

$$pontuação\ F1 = 2 * ((precisão * sensibilidade) / (precisão + sensibilidade))$$

3. ESTADO DA ARTE

Nessa seção é apresentado o levantamento do estado de arte em relação a seguinte pergunta relacionadas ao foco do presente trabalho: Quais pesquisas existem que apresentam a análise de desempenho de técnicas (como LDA, ou de ML/DL) aplicadas para classificar o tópico a partir de textos extraídos do aplicativo (criados com App Inventor)*?*

3.1 Definição do Protocolo de Revisão

O objetivo deste mapeamento é responder a seguinte pergunta de pesquisa: Quais abordagens existem para analisar e classificar, de forma automática, o tópico de um aplicativo móvel (criado com App Inventor) a partir de textos extraídos desse aplicativo?

Esta pergunta é refinada nas seguintes questões de análise:

AQ1. Quais abordagens existem para a classificação de tópicos de aplicativos móveis a partir dos textos extraídos do app e quais as suas características?

AQ2. Qual o desempenho dessas técnicas?

Critérios de inclusão e exclusão de artigos. Conforme o foco da pesquisa são definidos os seguintes critérios de inclusão e exclusão de artigos científicos.

- São incluídos somente artigos que apresentam abordagens de classificação de tópicos com base unicamente no texto do aplicativo.
- São excluídos artigos que apresentem abordagens de classificação de tópicos que sejam baseados na descrição, revisões, resenhas, etc., uma vez que no contexto do App Inventor essas informações não existem ou são escassas.
- São excluídos artigos com abordagens voltadas para a detecção de plágio, análise de sentimentos, aplicações de *chatbots*, etc.
- São considerados apenas artigos publicados nos últimos dez anos (desde 2012), levando em consideração o avanço recente especificamente em relação a aplicativos móveis.
- São incluídos artigos que apresentam uma avaliação de ou comparativos de desempenho das abordagens propostas.

Fonte. Para o levantamento dos artigos utilizados nesta pesquisa foram utilizados os principais bancos de dados e bibliotecas virtuais dentro da área da computação, tais como as bibliotecas virtuais ACM, IEEE Xplore e Scopus, com acesso por meio do Portal Capes, além do motor de busca de artigos científicos Google Scholar.

Critérios de qualidade. São considerados apenas artigos que apresentam informações substanciais e relevantes para a resolução das questões de análises. Portanto artigos cujo conteúdo se resumem a uma única página são excluídos.

Termos de busca. Com base na questão de pesquisa, várias pesquisas adicionais foram utilizadas com o intuito de calibrar a *string* de busca, identificando os termos mais relevantes para a busca bem como seus sinônimos (Tabela 6) para assim diminuir ao máximo o risco de não incluir algum artigo relevante.

Tabela 6. Termos de busca.

Termo	**Sinônimos**	**Tradução (inglês)**
classificação de tópico	classificação de texto, classificação de tópico, categorização de tópico, extração de palavras-chaves	“topic classification", “text classification”, “text categorization”, “keyword extraction”
aplicativo móvel	App, iOS, Android, App Inventor	mobile application, app, iOS, Android, App Inventor

Após a definição dos termos relevantes e seus sinônimos foi definida a *string* de busca padrão a ser aplicada nas bases de dados e bibliotecas virtuais:

```
("topic classification" OR "topic categorization" OR "text classification" OR "text categorization" OR "keyword extraction") AND ("mobile application" OR "app" OR "ios" OR "android" OR "app inventor")
```

A Tabela 7 apresenta as *strings* de busca formatadas para cada repositório. Para a busca no Google Scholar foi definida uma *string* especial uma vez que existe uma limitação no número de palavras para a busca de trabalhos.

Tabela 7. *Strings* de busca

Repositório	***String* de busca**
ACM Digital Library https://dl.acm.org/	[[Title: "topic classification"] OR [Title: "topic categorization"] OR [Title: "text classification"] OR [Title: "text categorization"] OR [Title: "keyword extraction"]] AND [[Abstract: "topic classification"] OR [Abstract: "topic categorization"] OR [Abstract: "text classification"] OR [Abstract: "text categorization"] OR [Abstract: "keyword extraction"]] AND [[Title: "mobile application"] OR [Title: "app"] OR [Title: "ios"] OR [Title: "android"] OR [Title: "app inventor"]] AND [[Abstract: "mobile application"] OR [Abstract: "app"] OR [Abstract: "ios"] OR [Abstract: "android"] OR [Abstract: "app inventor"]] AND [Publication Date: (01/01/2012 TO 12/31/2022)]
IEEE Xplore Digital Library https://ieeexplore.ieee.org/Xplore/home.jsp	("All Metadata":"topic classification" OR "All Metadata":"topic categorization" OR "All Metadata":"text classification" OR "All Metadata":"text categorization" OR "All Metadata":"keyword extraction") AND ("All Metadata":"mobile application" OR "All Metadata":"app" OR "All Metadata":"ios" OR "All Metadata":"android" OR "All Metadata":"app inventor")--2012-2023
Scopus https://www.scopus.com/home.uri	(TITLE-ABS-KEY ("topic classification" OR "topic categorization" OR "text classification" OR "text categorization" OR "keyword extraction") AND TITLE-ABS-KEY ("mobile application" OR "app" OR "ios" OR "android" OR "app inventor")) AND PUBYEAR > 2011
Google Scholar https://scholar.google.com.br/	"topic classification" "topic categorization" "text classification" "text categorization" "keyword extraction" "mobile application" "app" (2012-2022)

3.2 Execução de busca

A busca foi realizada em outubro de 2022. A Tabela 8 apresenta os resultados das buscas nas fontes.

Tabela 8. Número de artigos identificados por repositório e por fase de seleção.

Fonte	**Número de resultados de busca**	**Número de resultados analisados**	**Número de documentos potencialmente relevantes**	**Número de documentos relevantes**
ACM	0	0	0	0
IEEE Xplore	44	44	5	0
SCOPUS	140	140	8	0
Google Scholar	81900	200	13	0
Total				**0**

Durante a busca foram encontrados diversos artigos de revisão da literatura acerca de métodos para classificação de texto e/ou extração de palavras-chaves de textos, como, por exemplo, (ONAN et al., 2016)(DASARI; RAO, 2012)(DENG et al.,

2019)(MINAEE et al., 2021). Em outros artigos a classificação de textos era utilizada para a classificação de resenhas de aplicativos móveis como ferramenta de *feedback* para os desenvolvedores dos aplicativos. Nesses casos a classificação de textos é usada na análise de sentimentos e identificação de novos requisitos funcionais ou de erros reportados por usuários nas resenhas dos apps (AL KILANI et al., 2019)(GUZMAN et al., 2015)(SOUMIK et al., 2019)(TRIANTAFYLLOU et al., 2020).

Em relação a classificação de aplicativos móveis foram encontradas apenas pesquisas referentes a classificação com base nas descrições dos apps e em alguns casos com base nas API's (*application program interface*) desses aplicativos (Tabela 9).

Tabela 9. Classificação dos trabalhos encontrados em relação ao tipo de entrada

Título do artigo	**Referência**	**Usa como entrada**	
		Descrição do app	**API**
ClassifyDroid: Large scale Android applications classification using semi-supervised Multinomial Naive Bayes	(DONG et al., 2016)		x
Exploring Multiple Genres Text Classification: Classifying 61 genres of Mobile App Description based on Naïve Bayes and Count Vectorizer	(QIU et al., 2022)	x	
Research on Android Multi-classification Based on Text	(ZHANG et al., 2021)	x	
Design and Implementation of Application Classification Based on Deep Learning	(YANG et al., 2019)	x	

Analisando as pesquisas voltadas à classificação de aplicativos móveis, mesmo que não usem como entrada termos e textos extraídos do próprio aplicativo, observa-se a utilização de diversas técnicas de classificação (Tabela 10). Entre elas, por exemplo, *Naive Bayes* e suas variantes (DONG et al., 2016)(QIU et al., 2022), *Support Vector Machine* (SVM), *Random Forest*, *K Nearest Neighbors* (KNN) (QIU et al., 2022); *TextCNN, LSTM, RCNN, CRNN* (ZHANG et al., 2021)(YANG et al., 2019) e *BERT, BERT+Highway+GRU* (YANG et al., 2019).

Tabela 10. Técnicas e conjunto de dados adotados nos trabalhos correlatos

Referência	**Técnica(s) adotada(s)**	**Conjunto de dados**
(DONG et al., 2016)	*Multinomial Naive Bayes, Semi-supervised Multinomial Naive Bayes*	15.590 aplicativos e 10 categorias

(QIU et al., 2022)	*Multinomial Naive Bayes, Support Vector Machine, Random Forest, K Nearest Neighbors*	240.475 descrições de aplicativos e 60 categorias
(ZHANG et al., 2021)	*TextCNN, LSTM, RCNN, CRNN*	560.000 pedaços de informação descritiva de aplicativos e 17 categorias
(YANG et al., 2019)	*TextCNN, RCNN, BERT, BERT+Highway+GRU*	31.249 aplicativos e 14 categorias

O conjunto de dados utilizado por Dong et al. (2016) consiste em um conjunto contendo inicialmente 19.036 aplicativos extraídos por meio de um *web crawling* de uma loja de aplicativos chinesa (Mobile Market App Store) e classificados dentro de um conjunto de 28 categorias. Uma limpeza é aplicada a esse conjunto de dados eliminando categorias com menos de 300 aplicativos associados, resultando em um conjunto com 15.590 aplicativos e 10 categorias. Qiu et al. (2022) utiliza um arquivo .csv com um total de 240.475 descrições de aplicativos, sendo cada uma associada a uma das 60 categorias presentes. Já Zhang et al. (2021) constrói seu conjunto de dados por meio da extração de descrições de aplicativos da loja Appbao. No total são 560.000 pedaços de informação descritiva de aplicativos e 17 categorias. O conjunto de dados utilizado por Yang et al. (2019) é formado por um total de 31.249 aplicativos, extraídos de lojas de aplicativos populares da China, e associadas a 14 categorias.

Tabela 11. Medidas de desempenho adotadas nos trabalhos correlatos

Referência	**Medida(s) de desempenho avaliada(s)**	**Desempenho relatado**
(DONG et al., 2016)	Acurácia	Acurácia *Multinomial Naive Bayes:* 0.3928 - 0.8993 Acurácia *Semi-supervised Multinomial Naive Bayes:* 0.5166 - 0.8183
(QIU et al., 2022)	Acurácia	Acurácia *Multinomial Naive Bayes:* 0.54932 Acurácia *SVM:* 0.54248 Acurácia *Random Forest:* 0.40568 Acurácia *KNN:* 0.4942
(ZHANG et al., 2021)	pontuação F1 média	pontuação F1 média *TextCNN:* 0.873 pontuação F1 média *LSTM:* 0.881 pontuação F1 média *RCNN:* 0.8688 pontuação F1 média *CRNN:* 0.911
(YANG et al., 2019)	Acurácia, Precisão, Sensibilidade, pontuação F1	*TextCNN:* Acurácia 0.818, Precisão 0.7922, Sensibilidade 0.7943, pontuação F1 0.7932 *RCNN:* Acurácia 0.87, Precisão 0.8165, Sensibilidade 0.8024, pontuação F1 0.8094

		BERT: Acurácia 0.8915, Precisão 0.8741, Sensibilidade 0.8884, pontuação F1 0.8812 *BERT+Highway+GRU:* Acurácia 0.8936, Precisão 0.882, Sensibilidade 0.8892, pontuação F1 0.8856

Quanto às medidas de desempenho utilizadas, Dong et al. (2016) utiliza a acurácia como um comparativo de desempenho, assim como Qiu et al. (2022). Já Zhang et al. (2021) faz uso da pontuação F1 como métrica de comparação de desempenho. Yang et al. (2019) utiliza além da pontuação F1 e a acurácia métricas como precisão e sensibilidade, para comparar os métodos estudados.

Dentre os resultados obtidos pelos autores, para a métrica de acurácia, Dong et al. (2016), Qiu et al. (2022) e Yang et al. (2019) apresentaram valores que variam de 0.3928 a 0.8993 dependendo dos parâmetros utilizados em seus experimentos. É possível ver um desempenho melhor dos modelos *Multinomial Naive Bayes* e BERT em relação aos outros modelos usados. Esse desempenho superior pode se dar pela natureza desses modelos, uma vez que *Multinomial Naive Bayes* assume uma distribuição multinomial, e portanto consegue trabalhar melhor em cenários de mais de uma classe de classificação. Já o desempenho do BERT pode ser explicado por ser pré-treinado e conseguir captar mais informação sobre o contexto em que as palavras estão.

Para pontuação F1 Zhang et al. (2021) e Yang et al. (2019) encontraram valores superiores a 0.79 em seus testes, assim como os valores para precisão e sensibilidade encontrados por Yang et al. (2019). Para essas métricas os métodos que apresentaram melhor desempenho foram o BERT, por seu maior conhecimento sobre o contexto do texto, e CRNN, segundo Zhang et al. (2021), por ser um modelo que utiliza operações de convolução e rede neural cíclica, acaba por conseguir classificar melhor as amostras.

Analisando os resultados relatados na literatura é possível observar um desempenho maior de métodos como *Multinomial Naive Bayes*, BERT e CRNN na classificação de textos. Porém deve-se levar em conta também a natureza dos dados utilizados, como o tipo de textos que foram utilizados, uma vez que há grande diferença entre textos de descrição de aplicativos e API's. Outro fator que pode influenciar nos resultados é o tamanho do conjunto de dados utilizado. No contexto dos trabalhos analisados o impacto desse fator pode ter sido mínimo uma vez que tratavam-se de conjuntos de dados muito grandes, porém algo que pode ter causado

mais impacto é a proporção desses conjuntos de dados destinados para treinamento e teste.

Como resultado da revisão na literatura observa-se que não foi encontrado nenhum trabalho de pesquisa que tenha como foco a classificação de aplicativos por meio de textos e termos extraídos do próprio aplicativo. Porém, no contexto do presente trabalho voltado a aplicativos criados no App Inventor isso é uma necessidade, uma vez que muitos aplicativos compartilhados na *gallery* do App Inventor não apresentam um texto de descrição e não há a possibilidade de registro de uma resenha sobre o app.

Ameaças à validade da revisão da literatura: Assim como qualquer revisão e mapeamento sistemático da literatura existem algumas possíveis ameaças à validade dos resultados obtidos. Algumas dessas ameaças foram identificadas e estratégias de mitigação foram aplicadas para minimizar seus impactos. São elas:

- **Viés de publicação:** Resultados positivos têm maior probabilidade de serem publicados do que resultados negativos, o que faz com que mapeamentos sistemáticos possam vir a sofrer de um possível viés. Uma vez que este mapeamento tem como objetivo a revisão do estado da arte da classificação de aplicativos essa ameaça apresenta pouca influência e impacto para este estudo.

- **Identificação de estudos:** Outra ameaça é a omissão de estudos relevantes. Como estratégia de mitigação desse risco foi elaborada uma *string* de busca abrangente, fazendo uso de conceitos relevantes e sinônimos. Também foram utilizadas diversas bases de dados (*ACM Digital Library, IEEE Xplore Digital Library, Scopus e Google Scholar*), mitigando ainda mais esse risco. Apesar das estratégias adotadas esse risco ainda representa certa influência, uma vez que em bases com grande volume de dados a busca foi limitada aos primeiros 200 resultados.

- **Seleção e extração de dados de estudo:** Ameaças referentes à seleção e extração de dados foram mitigadas por meio do fornecimento de uma definição

detalhada dos critérios de inclusão/exclusão e de qualidade dos artigos. Foi definido e documentado um protocolo rígido para a seleção do estudo, que foi realizada e revisada cuidadosamente.

4. ESTUDO EXPERIMENTAL

4.1 Definição do estudo

O objetivo do estudo neste trabalho é comparar o desempenho de diferentes técnicas de classificação de textos com base em termos e textos extraídos diretamente de aplicativos móveis criados na plataforma App Inventor. Os textos utilizados para a classificação dos apps são extraídos diretamente de aplicativos em português desenvolvidos no App Inventor a partir do arquivo .aia.

Tabela 12. Exemplos de elementos textuais extraídos de projetos App Inventor

Nome do app	link para gallery	elementos textuais extraídos
Green Code	https://gallery.appinventor.mit.edu/?galleryid=49ef1527-9c36-469e-9773-6a4fada68fcc	Corante Caramelo IV Acidulante ácido fosfórico Aroma natural/extrato de noz de cola Corante Caramelo IV: É um corante artificial que confere coloração característica encontrada nos refrigerantes. É produzido através da mistura de açúcar com ácidos e amônia, na presença de elevada temperatura e pressão. Esse corante forma subprodutos que geram danos à saúde, um deles é conhecido como 4-MI (4-Metilimidazol) que pode causar câncer de pulmão, fígado, tireoide e leucemia. Acidulante ácido fosfórico: Produzido quimicamente, é responsável pela diminuição do pH do produto e regulação do teor de açúcar, destacando o paladar ácido e também operando como conservante. Esse ácido pode causar danos à saúde dos ossos. Aroma natural/extrato de noz de cola: É um aditivo aromatizante retirado diretamente de matérias-primas naturais, normalmente na forma de óleos. São utilizados para promover ou acentuar o sabor do produto. Não apresenta perigo de toxidade nesse tipo de aromatizante. None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None None /codigo /nome /ingredientes /aditivos /lista_aditivos Screen1 Escolha como você quer pesquisar! None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None /codigo None /nome None /ingredientes None /aditivos None /lista_aditivos None None /codigo /nome /ingredientes /aditivos /lista_aditivos Screen2 Digite o código de barras ou o nome do produto! ex: 7894900011531 Pesquisar Screen3 Nome: Código: Ingredientes: Aditivos: Informações sobre os Aditivos: Nova Consulta Screen4 Não encontramos seu produto! Tente novamente Nova Consulta
CantinaApp	https://gallery.appinventor.mit.edu/?galleryid=6424949763997696	Screen2 CantinaApp Bebidas Natália Boell Bebidas Água 500ml - R$ 2,50 Água de coco 350ml - R$ 3,50 Refri Lata 220ml - R$ 2,50 Refri Lata 350ml - R$ 3,50 Refri Garrafa 175ml - R$2,00 Refri Garrafa 500ml - 5,00 Refri Garrafa 1,5L - R$ 7,50 Refri Garrafa 2L - R$ 9,00 Suco Natural 200ml - R$ 3,00 Suco Natural 500ml - R$ 7,00 H2OH 500ml - R$ 4,50 H2OH 1,5L - R$ 8,00 Energético Energy Monster 500ml - R$ 7,00 Voltar Screen2 CantinaApp Cafés Natália Boell Cafés Café Expresso Pequeno - R$ 3,00 Café Expresso Grande - R$ 4,50 Cappuccino - R$ 5,50 Coffe Nut - R$ 9,00 Achocolatado - R$ 2,50 Café com Leite Pequeno - R$ 2,00 Café com Leite Grande - R$ 4,00 Chocolate Quente - R$ 5,00 Café Mocha - R$ 6,50 Cappuccino Ice - R$ 7,00 Café Expresso Caramelo - R$ 6,50 Café Expresso da Casa - R$ 7,00 Café Expresso Canela - R$ 5,50 Voltar Screen2 CantinaApp Doces Natália Boell Doces Croissant de Chocolate - R$ 4,50 Folhado - R$ 4,00 Bananinha - R$ 2,50 Brigadeiro -R$ 2,00 Waffles - R$ 3,00 Bolo no Pote - R$ 5,50 Donuts - R$ 3,50 Nega Maluca - R$ 2,50 Cuca - R$ 2,50 Rocambole - R$ 5,00 Orelha de Gato - R$ 2,50 Tortinha de Chocolate - R$ 3,50 Mini Churros - R$ 1,00 Voltar Screen2 CantinaApp Guloseimas Natália Boell Guloseimas Bala de Coco - R$ 0,20 Bala de Banana - R$ 0,20 Pirulitos - R$ 0,25 Pirulito Blong - R$ 0,55 Bala Chiclete - R$ 0,10 Nucita Napolitano - R$ 0,50 Bala de Yogurte - R$ 0,10 Trento - R$ 2,00 Paçoquita - R$ 0,50 Bolacha Danix - R$ 2,50 Wafer Prestígio - R$ 3,00 Bala de Canela - R$ 0,10 Chiclete Big Big - R$ 0,10 Voltar Screen2 CantinaApp

		Lanches Natália Boell Lanches Pastel Frito - R$ 3,50 Coxinha - R$ 3,50 Espetinho - R$ 4,00 Bolinho de Carne - R$ 4,00 X - Burguer - R$ 4,50 Pastel Assado - R$ 4,00 Pão Pizza - R$ 4,00 Cachorro Quente -R$ 4,50 Pizza - R$ 3,50 Pão de Queijo - R$ 2,00 Enroladinho de Salsicha - R$ 4,00 Enrolado de Queijo - R$ 3,50 Enrolado de Salsicha - R$ 3,50 Voltar Screen2 CantinaApp Promoções Natália Boell Promoções Segunda-Feira: Refri 2L - R$ 7,00 Chocolate Quente - R$ 3,50 Terça-Feira: Pastel Frito - R$ 3,00 Àgua de Coco 350ml - R$ 2,00 Quarta-Feira: Croissant de Chocolate - R$ 3,00 Pizza - R$ 2,00 Quinta-Feira: Coxinha - R$ 3,00 Doritos - R$ 6,00 Sexta-Feira: Petit Gâteau - R$ 5,50 Açai com Sorvete - R$ 6,00 Voltar Screen2 CantinaApp Salgadinhos Natália Boell Salgadinhos Doritos - R$ 8,00 Baconzitos - R$ 6,50 Cheetos - R$ 7,00 Yokitos - R$ 5,50 Fandangos - R$ 7,50 Pingo - R$ 3,50 Cebolitos - R$ 6,50 Stiksy - R$ 4,50 Ruffles - R$ 6,00 Pringles - R$ 9,00 Cheetos Pipoca - R$ 5,50 Pipoca Salgada - R$ 3,50 Pipoca Doce - R$ 3,50 Voltar Screen2 CantinaApp Saudáveis Natália Boell Saudáveis Salada de Frutas - R$ 5,00 Tapioca - R$ 3,50 Coxinha Fit - R$ 4,50 Panqueca Fit - 6,50 Panqueca Vegetariana - R$ 5,50 Tortinha Beringela - 8,00 Brownie Fit - R$ 7,00 Bolo de Cacau - R$ 4,50 Bolo de Banana - R$ 4,00 Empanado Fit - R$ 5,00 Sanduíche Natural - R$ 4,00 Kibe Vegano - R$ 4,50 Vitamina - R$ 7,50 Voltar Screen2 CantinaApp scrTela1 Natália Boell Cardápio Cantina Natália Boell Saudaveis Bebidas Sorvetes Lanches Salgadinhos Doces Promocoes Guloseimas Cafes CantinaApp Screen2 Natália Boell Menu Cantina Bebidas Lanches Doces Guloseimas Cafés Saudáveis Sorvetes Salgadinhos Promoções Sair Screen2 CantinaApp Sorvetes Natália Boell Sorvetes Picolé de Frutas - 3,50 Picolé Clássicos - 3,00 Banana Split - R$ 8,50 Sundae - R$ 6,50 Moka - R$ 5,50 Petit Gâteau - R$ 8,00 Milk Shake 250ml - R$ 4,00 Milk Shake 900ml - R$ 8,00 Fondue de Sorvete - R$ 10,00 Açai 250ml - R$ 3,50 Açai 500ml - R$ 6,00 Açai com Sorvete - R$ 8,00 Casquinha - R$ 3,50 Voltar

Para a etapa de pré processamento primeiramente é feita a limpeza dos textos brutos extraídos, bem como eliminação de sinais de pontuação, números e caracteres especiais, além da conversão de letras maiúsculas em minúsculas. A partir desse ponto são eliminados os termos de pouca importância para classificação, como artigos, preposições, pronomes e advérbios, chamados também de *stopwords*.

Com o texto limpo são testadas várias alternativas de filtragens a mais, tais como, eliminação de termos que não sejam substantivos, ou utilização do texto bruto para a classificação, assim como *stemming* e *lemmatization* do conteúdo textual. Também será utilizada a técnica de redução de dimensionalidade LSA.

Após a etapa de pré processamento e de acordo com a revisão da literatura, foram escolhidas várias técnicas de classificação de texto que serão aplicadas, incluindo:

- técnicas de *machine learning*: *Decision Tree, Random Forest, Logistic Regression, Multinomial Naives Bayes, K Nearest Neighbour, Support Vector Machine* (SVM).
- técnicas de *deep learning*: BERT.
- técnica de modelagem de tópicos: LDA

O desempenho destas técnicas é medido e comparado por meio de acurácia, precisão, sensibilidade e pontuação F1.

4.2 Definição de tópicos de apps criados com App Inventor

Para a classificação dos aplicativos se faz necessário a definição das categorias. A Tabela 13 apresenta as categorias que serão utilizadas para a classificação dos aplicativos App Inventor. Elas foram definidas com base no mapeamento das categorias usadas nas principais lojas online de aplicativos, Google Play e Apple App Store.

Tabela 13. Categorias criadas para App Inventor e suas definições

CATEGORIA	Definição
Animais e pets	Aplicativos de animais domesticados ou não, identificação de animais, cuidados com animais.
Automóveis, veículos e transporte	Aplicativos relacionados a automóveis, bicicletas ou outros tipos de veículos e transporte.
Beleza e moda	Aplicativos que fornecem informações, história, agendamento ou tutoriais de beleza.
Cidadania e questões sociais	Aplicativo para tratar de questões cívicas, denúncias, transparência de recursos públicos e similares.
Comes-e-bebes	Aplicativos que fornecem recomendações, instruções, receitas ou críticas relacionadas à preparação, consumo ou revisão de alimentos ou líquidos/sucos.
Comunicação	Aplicativos que fornecem suporte para comunicação como acesso a chats ou aplicativos de chats, contatos, telefones.
Design, pintura e fotografia	Aplicativos para esboços, ferramentas de pintura e design, livros para colorir ou auxiliam na captura, edição, gerenciamento, armazenamento ou compartilhamento de fotos.
Educação	Aplicativos que abordam o ensino de uma habilidade, assunto específico ou são voltados à questões de estudo, como notas de provas, sala de aula, instituições de ensino.
Engenharia, física e construção	Aplicativo relacionado a construções, cálculo de materiais para obra, planejamento de obras, cálculo de medidas, motores, máquinas.
Entretenimento	Aplicativos que informam o usuário sobre algum evento ou apresentam conteúdo de entretenimento visual ou outro conteúdo de entretenimento.
Espiritualidade, crença e adivinhação	Aplicativos de adivinhação, horóscopo, questões religiosas, filosóficas e existenciais.
Ferramentas do celular	Aplicativos de lanterna, leitor de códigos e similares.
Finanças e trabalho	Aplicativos que realizam transações financeiras ou auxiliam o usuário em questões financeiras comerciais, profissionais ou pessoais, ou divulgam e encontram empregos, trabalhos, vagas, portfólios e similares.
Medicina e saúde	Aplicativos focados em educação médica, gerenciamento de informações ou informações de saúde para pacientes ou profissionais de saúde, cuidados do bebê e infantis.
Meio ambiente e botânica	Aplicativo de reciclagem, descarte ou cuidado e informações sobre o meio ambiente ou plantas, identificação plantas, cuidados com plantas.

Matemática	Aplicativos para fornecer informações ou realizar cálculos algébricos, trigonométricos e geométricos.
Música	Aplicativo para ouvir rádio, músicas e similares.
Produtividade	Aplicativos que tornam um processo ou tarefa específica mais organizado ou eficiente.
Robótica, computação física e automação	Aplicativos para controlar robô, comunicar-se com placas física (ex.: Arduino) e automação residencial/predial.
Vida saudável e esporte	Aplicativos relacionados a uma vida saudável ou prática de um esporte, treino, dieta, nutrição, gerenciamento de estresse, condicionamento físico, indicadores de peso (ex.: IMC).
Turismo e geografia, tempo e meteorologia	Aplicativos que fornecem informações ou mapas sobre um local, pontos de interesse ou fornecem previsões, alertas e informações relacionadas às condições meteorológicas de um local.

Além disso foi verificada a existência de diversos aplicativos na *Gallery* do App Inventor de apps desenvolvidos no âmbito da iniciativa Computação na Escola/INCoD/INE/UFSC que pudessem ser classificados para alguma dessas categorias. Como resultado foram excluídas algumas categorias como, maternidade; casa e lar; revistas, notícias, livros ou dicionários por não serem tópicos normalmente abordados pelos aplicativos criados pelo público alvo em questão.

Outras categorias foram mescladas por serem correlacionadas entre si, ampliando, assim, o escopo da categoria, como meio ambiente e botânica; design, pintura e fotografia; finanças e trabalho; turismo e geografia, tempo e meteorologia. Também foram criadas algumas categorias novas, também no intuito de generalizar seu escopo: automóveis, veículos e transporte; beleza e moda; espiritualidade, crença e adivinhação; cidadania e questões sociais; engenharia, física e construção. Além disso, foram criadas também as categorias matemática; música; robótica, computação física e automação para atender aplicativos que não se encaixam no escopo das demais categorias.

4.3 Preparação do conjunto de dados

Para o conjunto de dados, usado neste trabalho, são utilizados dados de um total de 1.672 aplicativos, em português brasileiro, criados e publicados na *gallery* do App Inventor, além de apps desenvolvidos no âmbito da iniciativa Computação na Escola/INCoD/INE/UFSC. O uso do App Inventor bem como os apps produzidos na plataforma são licenciados sob a licença *Creative Commons* o que o torna livres para a utilização, modificação e compartilhamento. Os dados dos 1.672 aplicativos estão

compilados em uma planilha incluindo as informações relevantes conforme apresentado na Tabela 14.

Tabela 14. Composição do conjunto de dados

Coluna	Informação	Exemplo
1	ID de tópico	4
2	**Tópico**	Medicina e Saúde
3	Tópico em inglês	Medicine and Health
4	Nome do aplicativo	IMC
5	**Texto extraído do aplicativo**	Muito abaixo do peso Abaixo do peso Peso normal Acima do peso Obesidade I Obesidade II (severa) Obesidade III (mórbida) None None None None Screen1 Calculadora de IMC Digite seu Peso: Digite sua Altura: IMC: Situação: Calcular Limpar Campos Screen1 Screen1

Para o treinamento dos modelos e classificação são usados somente as seguintes informações: tópico e texto bruto extraído do aplicativo (marcado em negrito na Tabela 14).

4.4 Treinamentos

4.4.1 Treinamento com LDA

Para a modelagem de tópicos com o método *Latent Dirichlet Allocation* (LDA) foram utilizados os dados textuais dos aplicativos presentes no conjunto de dados. Foram feitos três experimentos de modelagem: o primeiro utilizando apenas os substantivos dos textos, o segundo e terceiro experimentos utilizaram os textos bruto com diferença apenas no número de *features*, ou palavras, utilizadas, sendo 10.000 *features* no segundo experimento e 17.260 *features* no terceiro. Em todos os experimentos os dados textuais tiveram seus caracteres especiais e números removidos, além da conversão para letras minúsculas, e o parâmetro de número de tópicos a serem gerados foi configurado para 21, conforme a quantidade dos tópicos definidos no contexto de apps com App Inventor (Tabela 13). A Tabela 15 apresenta os resultados da modelagem de tópicos com LDA no primeiro experimento.

Tabela 15. Modelagem com LDA só com substantivos

Tópicos	10 principais palavras relacionadas
Tópico #0	produto , faixa , produtos , passos , agenda , hora , grupo , mensagens , suspensão , distância
Tópico #1	seg , compromisso , hora , evento , dia , compromissos , horário , atividade , eventos , prioridade
Tópico #2	corrente , tensão , resistência , busca , resistor , futebol , circuito , serie , potência , som
Tópico #3	nota , aluno , média , aprovado , pedidos , notas , cep , geografia , biologia , música
Tópico #4	cidade , feira , preço , segunda , sexta , umidade , carne , quarta , cerveja , pizza
Tópico #5	categoria , contato , grupo , frase , música , mapas , sorte , biscoito , cardápio , contas
Tópico #6	peso , altura , obesidade , grau , magreza , kg , normal , potência , classificação , iluminação
Tópico #7	mensagem , rádio , chat , limite , favoritos , frente , sensor , carro , papo , volume
Tópico #8	protocolo , cidade , saldo , avaliação , corrente , alternativo , conta , passo , entrada , pedido
Tópico #9	conta , inglês , tamanho , aposentadoria , porto , tradutor , peças , km , gasolina , reais
Tópico #10	quadro , look , brasil , socorro , graus , militar , animais , situação , dom , denúncia
Tópico #11	triângulo , setor , contato , torre , temperatura , solo , endereço , ângulos , ângulo , categoria
Tópico #12	receita , leite , bolo , xícara , chá , chocolate , ingredientes , açúcar , sopa , farinha
Tópico #13	cálculo , local , locais , férias , turma , observações , histórico , dom , valores , salário
Tópico #14	kg , meses , juros , quantidade , instituto , altura , preço , parcelas , mês , peso
Tópico #15	tempo , base , questão , tarefa , tarefas , quadrado , força , trabalho , altura , informações
Tópico #16	luz , quarto , sala , linha , cozinha , controle , liga , valores , banheiro , conexão
Tópico #17	calculadora , localização , endereço , latitude , longitude , local , estação , ano , dia , telefones
Tópico #18	ano , colher , ingredientes , preparo , assinatura , dose , chá , sopa , minutos , sal
Tópico #19	contatos , carta , largura , email , telefone , cliente , comprimento , seleção , quantidade , obra
Tópico #20	foto , material , cal , supermercado , praia , operação , mapa , hora , rio , estado

Analisando os tópicos gerados pelo modelo de LDA nota-se uma dificuldade do modelo em gerar tópicos com uma distinção clara de temas, tendo tópicos que podem ser associados às mesmas categorias, como o Tópico #18 e Tópico #12 que podem ser associados a categoria Comes-e-Bebes.

Para o segundo e terceiro experimento os resultados obtidos podem ser vistos nas Tabelas 16 e 17.

Tabela 16. Modelagem com LDA com texto bruto - 10.000 *features*

Tópicos	**10 principais palavras relacionadas**
Tópico #0	screen , de , não , para , for , em , que , textbox , sim , com
Tópico #1	dica , caixadetexto , para , screen , do , área , triângulo , de , altura , lado
Tópico #2	none , screen , de , nome , data , do , for , lista , text , hora
Tópico #3	png , none , not , found , jpg , screen , pt , text , pgm , atenção
Tópico #4	none , material , cal , rpmdia , operação , vm , com , não , um , endereço
Tópico #5	de , do , que , da , com , no , em , os , se , para
Tópico #6	de , screen , com , tela , whatsapp , do , br , não , android , intent
Tópico #7	screen , para , dica , caixadetexto , texto , de , none , você , voltar , do
Tópico #8	http , com , br , www , https , globoesporte , screen , view , globo , action
Tópico #9	screen , de , questão , cabo , para , da , frente , lista , disjuntor , salvar
Tópico #10	screen , none , digite , valor , número , de , resultado , calcular , limpar , voltar
Tópico #11	none , on , off , ligar , bluetooth , desligar , screen , dispositivo , luz , quarto
Tópico #12	data , de , compromisso , linha , screen , prevista , para , da , feira , entendi
Tópico #13	de , que , screen , para , em , da , descrição , não , com , um
Tópico #14	peso , screen , seu , obesidade , imc , altura , do , com , de , grau
Tópico #15	de , que , kg , média , da , cm , até , corrente , led , altura
Tópico #16	seg , mg , avaliação , de , produto , vitamina , endereco , preco , mcg , tipo
Tópico #17	senha , login , ok , usuário , nome , none , cidade , screen , não , cadastro
Tópico #18	de , tela , do , screen , da , para , resultado , jpg , praia , none
Tópico #19	palhoça , estação , florianópolis , screen , mp , do , de , que , carro , são
Tópico #20	de , screen , em , com , ingredientes , leite , uma , chá , receita , colher

Tabela 17. Modelagem com LDA com texto bruto - 17.260 *features*

Tópicos	**10 principais palavras relacionadas**
Tópico #0	de , for , text , não , jpg , label , com , tela , que , pgm
Tópico #1	pista , campodetexto , para , num , the , el , change , fit , treinosuperior , text
Tópico #2	none , cidade , de , fisic , encontrou , lancar , assinatura , não , lampada , quadro
Tópico #3	de , paulo , são , rio , carta , por , janeiro , sua , screen , não

Tópico #4	com , www , https , de , met , youtube , watch , android , intent , view
Tópico #5	de , para , dica , caixadetexto , screen , em , com , ingredientes , leite , do
Tópico #6	screen , none , digite , para , de , número , valor , não , resultado , nome
Tópico #7	txt , nome , screen , palhoça , estação , florianópolis , de , do , passos , da
Tópico #8	de , da , kg , média , até , que , cm , not , found , altura
Tópico #9	none , com , ok , de , não , atenção , screen , do , senha , um
Tópico #10	descrição , de , screen , pt , en , inglês , para , do , um , porto
Tópico #11	screen , de , para , com , que , um , pedir , texto , do , da
Tópico #12	none , de , screen , material , png , resultado , cal , do , endereço , for
Tópico #13	de , screen , seg , http , rádio , com , ml , da , colher , radio
Tópico #14	de , da , feira , screen , silva , ano , santos , none , tela , mat
Tópico #15	none , você , linha , categoria , de , em , olá , foi , pagdeve , partindo
Tópico #16	screen , png , tela , com , none , localização , voltar , de , http , para
Tópico #17	http , png , on , off , ligar , bluetooth , desligar , globoesporte , com , luz
Tópico #18	screen , peso , imc , obesidade , seu , altura , do , textbox , para , digite
Tópico #19	de , screen , que , para , da , do , se , não , você , em
Tópico #20	de , corrente , que , do , com , tensão , em , no , led , da

Assim como no primeiro experimento, o modelo apresentou dificuldade em modelar tópicos com temas distintos, mesmo com a adição de informação para a modelagem. Isso se deve ao fato das *features* adicionadas não possuírem valor semântico para auxiliar o modelo a gerar tópicos mais precisos, funcionando de forma contrária, fazendo com que o modelo gerasse tópicos confusos e sem significado. Com base nos resultados dos experimentos é possível afirmar que para a tarefa de modelagem de tópicos utilizando textos curtos, como é a natureza dos textos presentes em aplicativos, o modelo de LDA não apresenta desempenho satisfatório.

4.4.2 Treinamento com *Machine Learning*

Para analisar e comparar técnicas de *machine learning* no contexto de classificação de aplicativos, são realizados treinamentos de métodos clássicos de *machine learning*, incluindo:

- *Decision Tree*
- *Random Forest*
- Regressão Logística
- *Multinomial Naive Bayes*
- *Gaussian Naive Bayes*
- *K Nearest Neighbour*
- *Support Vector Machine*

Para o treinamento dos métodos citados o conjunto de dados foi dividido em duas partes: conjunto de treinamento, com 70% do conjunto de dados, e conjunto de validação, composto pelos 30% restantes.

Além dos métodos de classificação clássicos é também realizado experimentos com o método de *deep learning* BERT. Para o treinamento do BERT o conjunto de treinamento é composto por 90% do conjunto de dados enquanto o conjunto de validação é composto pelos 10% restantes.

Os experimentos foram realizados dentro da plataforma Colaboratory do Google, um serviço de nuvem que oferece recursos computacionais para a execução de códigos em Python pelo navegador, muito utilizado para pesquisas em aprendizado de máquina, análise de dados, dentre outros. Para a importação e manipulação dos dados foi utilizado a biblioteca *open-source* Pandas.

Foram realizados diversos experimentos, alguns utilizando o texto pré-processado e limpo, e outros o texto bruto, apenas com remoção de números e caracteres especiais e conversão para letra minúscula. Em alguns testes foi utilizado métodos de *stemming* e *lemmatization* dos textos, e foi utilizado também o método LSA para a redução de dimensão.

O primeiro passo após a importação dos dados consistiu no pré-processamento dos dados. Nesta etapa primeiramente foi feita a remoção de números e quaisquer caracteres especiais presentes nos textos, em seguida todos os caracteres foram convertidos para sua forma minúscula. Também foram removidas *stopwords* e palavras cujo tamanho fosse menor que dois caracteres. Os dados pré-processados foram utilizados no primeiro experimento. A Tabela 18 apresenta dois exemplos de aplicativos com o texto bruto extraído e o texto pré-processado.

Tabela 18. Exemplos de resultados do pré-processamento

Nome do Aplicativo	Texto Bruto	Texto Pré-processado
Quadro Verde	None Screen1 CROP CROP None Viewer Galeria jkbjhb Text for Label1 ic_info.png Sobre ic_exit.png Sair ic_folder.png ic_x.png ic_edit.png Galeria Adicionar disciplina None Usabilidade ADICIONAR CANCELAR #000000 #D1D1D1 Excluir disciplina EXCLUIR CANCELAR ic_delete.png None Disciplina " " adicionada! Disciplina " " já cadastrada. Por favor, utilize outro nome. Disciplina já existente OK Sobre Autor: Diogo F N Machado QuadroVerde v1.0 OK Sair QuadroVerde Quadro Verde Versão 1.0 Diogo F. N. Machado QuadroVerde Recentes Disciplinas Galeria Viewer x	galeria galeria disciplina disciplina disciplina disciplina disciplina existente autor quadro disciplinas galeria
Diversi Food	Usuário ou senha vazios! Usuário None Senha None Screen2 Usuário não cadastrado Senha não confere cadastrarUsuario telaLogin DIVERSIFOOD Usuário Senha Entrar Cadastre-se Screen3 Screen4 Screen1 telaPrincipal BEM VINDO! Cadastrar receita Procurar receita Sair Por favor preencha todos os campos! Imagem da receita Nome da receita Ingredientes Modo de preparo Tempo estimado Número de porções cadastro_sucesso Screen2 cadastroReceita Cadastrar receita Clique aqui para escolher uma imagem para a receita Nome da receita Ingredientes Modo de preparo Tempo estimado Número de porções Cadastrar Voltar Nome da receita None Screen5 Não há receita com esse título! Screen2 procurarReceita PROCURAR RECEITA Título da receita Procurar Voltar Nome da receita None Imagem da receita None Ingredientes None Modo de preparo None Tempo estimado None Número de porções None Screen4 Screen5 Título da receita Ingredientes: Ingredientes da receita Modo de preparo: Modo de preparo da receita Tempo Estimado (minutos): tempo Porções: Porções Voltar Preencha todos os campos! Usuário Senha Usuário cadastrado com sucesso! Screen1 cadastrarUsuario Cadastro de usuário Usuário Senha Cadastrar Voltar cadastro_receita_falha Seu cadastro falhou! Tente novamente Cadastrar nova receita Voltar para tela principal Screen3 Screen2 cadastro_sucesso Receita cadastrada com sucesso! Cadastrar nova receita Voltar para tela principal	receita receita receita receita ingredientes preparo tempo estimado porções receita receita receita ingredientes preparo tempo estimado porções receita receita título receita título receita receita receita ingredientes preparo tempo estimado porções título receita ingredientes ingredientes receita preparo preparo receita tempo estimado minutos tempo porções porções receita falha receita receita receita

Os aplicativos, que após o processo de pré-processamento não possuem mais textos, são removidos.

Em seguida foi aplicado o método *count vectorizer,* da biblioteca *open-source* Scikit-learn. Este método de *feature extraction* converte a coleção de documentos de textos em uma matriz de frequências de palavras. O vocabulário utilizado é composto por todas as palavras, ou *features*, que aparecem nos textos, gerando um vocabulário com 4.530 *features*. Após a geração do modelo de *bag-of-words* são construídos então os conjuntos de teste e de validação.

Para os modelos de *machine learning* foi utilizada a implementação da biblioteca Scikit-learn. Como alguns desses métodos foram concebidos para a resolução de problemas de classificação binária, é necessário adaptá-los para que consigam resolver problemas de classificação multiclasse. Para isso foi utilizado o classificador *OneVsRestClassifier* da biblioteca Scikit-learn, um classificador que

utiliza a estratégia “Um-contra-Todos” para a generalização dos modelos para problemas multiclasse. Nele o problema de classificação multiclasse é reduzido a vários problemas de classificação binária, é criado um modelo para cada classe e a classificação consiste em determinar se o dado pertence a essa classe associada ou não.

Para a obtenção das medidas de desempenho foi usado o *classification report* do Sckit-learn que retorna um relatório do desempenho obtido pelo modelo. Além do valor de acurácia, o relatório apresenta para as métricas de precisão, sensibilidade e pontuação F1 o cálculo de dois tipos de média, uma média aritmética (*macro avg*) e uma média ponderada (*weighted avg*), que leva em conta o número de instâncias verdadeiras para cada classe, levando, assim, em consideração o desequilíbrio no número de instâncias para cada classe que possa existir.

A Tabela 19 apresenta os resultados obtidos após o treinamento dos métodos de classificação baseados em *machine learning,* além do modelo de *deep learning* BERT, bem como alguns parâmetros utilizados.

Tabela 19. Resultados dos modelos com texto composto por substantivos apenas

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5632	**macro avg: 0.71** weighted avg: 0.71	macro avg: 0.49 weighted avg: 0.56	macro avg: 0.53 weighted avg: 0.59
Random Forest	Nº árvores: 100 Critério: entropia	0.6959	**macro avg: 0.71** **weighted avg: 0.72**	macro avg: 0.61 weighted avg: 0.70	macro avg: 0.62 weighted avg: 0.68
Regressão Logística	Solucionador: Lbfgs	0.6767	macro avg: 0.66 weighted avg: 0.68	macro avg: 0.57 weighted avg: 0.68	macro avg: 0.60 weighted avg: 0.66
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.7109**	macro avg: 0.67 weighted avg: 0.71	**macro avg: 0.63** **weighted avg: 0.71**	**macro avg: 0.64** **weighted avg: 0.70**
Gaussian Naive Bayes	Prévia: None	0.4711	macro avg: 0.59	macro avg: 0.42	macro avg: 0.46

			weighted avg: 0.66	weighted avg: 0.47	weighted avg: 0.52
KNN	Nº vizinhos: 10 Função distância: minkowski	0.5075	macro avg: 0.49 weighted avg: 0.53	macro avg: 0.38 weighted avg: 0.51	macro avg: 0.39 weighted avg: 0.47
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6724	macro avg: 0.70 weighted avg: 0.70	macro avg: 0.57 weighted avg: 0.67	macro avg: 0.60 weighted avg: 0.66
BERT		0.6346	macro avg: 0.61 weighted avg: 0.65	macro avg: 0.50 weighted avg: 0.63	macro avg: 0.50 weighted avg: 0.60

Utilizando apenas os substantivos presentes nos textos, o método de *multinomial naive bayes* apresentou o melhor desempenho para as métricas de acurácia, sensibilidade e pontuação F1, perdendo apenas na precisão para o modelo de *random forest*. Esse desempenho pode ser explicado pela utilização de textos sem *stopwords*, o que auxilia o algoritmo a ter um desempenho melhor, uma vez que ele utiliza a frequência das palavras, sem atribuir peso às palavras para classificar um texto. O modelo de BERT obteve um desempenho mediano se comparado aos demais métodos de *machine learning*. Isto talvez deve-se ao fato do modelo precisar de um conjunto de dados muito maior para desempenhar melhor.

O segundo experimento consistiu na execução dos modelos com os dados textuais brutos, sendo realizadas apenas a remoção de caracteres especiais e conversão para letras minúsculas. Primeiramente os modelos foram executados com um vocabulário com 10.000 *features,* em seguida foi utilizado um vocabulário contendo todas as palavras presentes nos textos, gerando um vocabulário com 17.260 *features*. Os resultados obtidos são apresentados nas Tabelas 20 e 21.

Tabela 20. Resultados dos modelos com texto bruto - 10.000 *features*

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5498	macro avg: 0.64 weighted avg:	macro avg: 0.50 weighted avg:	macro avg: 0.49 weighted avg:

			0.71	0.55	0.58
Random Forest	Nº árvores: 100 Critério: entropia	**0.7052**	**macro avg: 0.72** weighted avg: 0.73	**macro avg: 0.60** **weighted avg: 0.71**	**macro avg: 0.60** **weighted avg: 0.68**
Regressão Logística	Solucionador: Lbfgs	0.6673	macro avg: 0.62 weighted avg: 0.68	macro avg: 0.55 weighted avg: 0.67	macro avg: 0.56 weighted avg: 0.66
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	0.6992	macro avg: 0.68 weighted avg: 0.72	macro avg: 0.56 weighted avg: 0.70	macro avg: 0.57 **weighted avg: 0.68**
Gaussian Naive Bayes	Prévia: None	0.4143	macro avg: 0.62 **weighted avg: 0.74**	macro avg: 0.34 weighted avg: 0.41	macro avg: 0.39 weighted avg: 0.49
KNN	Nº vizinhos: 10 Função distância: minkowski	0.4084	macro avg: 0.36 weighted avg: 0.45	macro avg: 0.31 weighted avg: 0.41	macro avg: 0.29 weighted avg: 0.39
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6414	macro avg: 0.61 weighted avg: 0.64	macro avg: 0.52 weighted avg: 0.64	macro avg: 0.53 weighted avg: 0.61
BERT	--	--	--	--	--

Utilizando os textos brutos observa-se um leve aumento na acurácia do modelo *random forest*, enquanto os modelos restantes tiveram uma queda na acurácia, muito possivelmente devido a presença das *stopwords* e palavras com baixo valor semântico. Quanto às métricas de precisão, os modelos de *random forest, multinomial naive bayes* e *gaussian naive bayes* demonstraram um aumento de desempenho. Já em relação às métricas de sensibilidade e pontuação F1 praticamente todos os modelos tiveram um desempenho pior utilizando texto bruto com 10.000 *features*.

Random Forest foi o modelo que obteve melhor desempenho, tendo a melhor acurácia, sensibilidade e pontuação F1 e a melhor média macro de precisão e segunda melhor média ponderada em precisão.

Nesse experimento não foi possível a execução e medição de desempenho do modelo BERT, uma vez que a biblioteca utilizada não permite a especificação de um número arbitrário de *features* a ser considerado.

Tabela 21. Resultados dos modelos com texto bruto - 17.260 *features*

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5299	macro avg: 0.64 weighted avg: 0.71	macro avg: 0.45 weighted avg: 0.53	macro avg: 0.47 weighted avg: 0.56
Random Forest	Nº árvores: 100 Critério: entropia	0.6912	**macro avg: 0.73** **weighted avg: 0.74**	**macro avg: 0.59** **weighted avg: 0.69**	**macro avg: 0.58** **weighted avg: 0.67**
Regressão Logística	Solucionador: Lbfgs	0.6693	macro avg: 0.62 weighted avg: 0.68	macro avg: 0.56 weighted avg: 0.67	macro avg: 0.57 weighted avg: 0.66
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.6932**	macro avg: 0.69 weighted avg: 0.73	macro avg: 0.55 **weighted avg: 0.69**	macro avg: 0.56 **weighted avg: 0.67**
Gaussian Naive Bayes	Prévia: None	0.4143	macro avg: 0.62 **weighted avg: 0.74**	macro avg: 0.34 weighted avg: 0.41	macro avg: 0.39 weighted avg: 0.49
KNN	Nº vizinhos: 10 Função distância: minkowski	0.4024	macro avg: 0.34 weighted avg: 0.44	macro avg: 0.30 weighted avg: 0.40	macro avg: 0.29 weighted avg: 0.38
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6335	macro avg: 0.63 weighted avg: 0.64	macro avg: 0.51 weighted avg: 0.63	macro avg: 0.52 weighted avg: 0.60
BERT		0.6786	macro avg: 0.54 weighted avg: 0.62	macro avg: 0.52 weighted avg: 0.68	macro avg: 0.50 weighted avg: 0.63

Utilizando 17.260 *features* não foi observada uma grande mudança de desempenho. Alguns modelos tiveram uma leve queda no desempenho geral, como *decision tree, random forest, multinomial naive bayes,* KNN e SVM. Por outro lado, outros mantiveram um desempenho quase idêntico, como regressão logística e *gaussian naive bayes*, apesar do modelo de regressão logística ter apresentado um leve aumento na sua acurácia. Destaca-se o aumento de acurácia e sensibilidade do modelo BERT, e um leve aumento na média ponderada da pontuação F1, enquanto a precisão teve uma piora.

Apesar da queda de desempenho, o modelo de *multinomial naive bayes* foi o que apresentou a melhor acurácia. Já o modelo *random forest* foi o que obteve a melhor precisão, sensibilidade e pontuação F1 em todas as médias. Nota-se que o aumento de informação nos textos não auxiliou de forma positiva no desempenho dos modelos, uma vez que as *features* adicionadas representam palavras com baixo valor semântico para a classificação.

Para os experimentos a seguir foi utilizado os mesmos passos de pré-processamento já citados, com adição da utilização da técnica de *stemming* do conteúdo textual. Para o *stemming* foi utilizado o modelo *SnowballStemer* da biblioteca NLTK. Os resultados estão apresentados na Tabela 22.

Tabela 22. Resultados dos modelos só com substantivos e com *stemming*

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5439	macro avg: 0.68 weighted avg: 0.69	macro avg: 0.47 weighted avg: 0.54	macro avg: 0.50 weighted avg: 0.56
Random Forest	Nº árvores: 100 Critério: entropia	0.6809	macro avg: 0.68 weighted avg: 0.69	macro avg: 0.58 weighted avg: 0.68	macro avg: 0.59 weighted avg: 0.66
Regressão Logística	Solucionador: Lbfgs	0.6788	**macro avg: 0.69** weighted avg: 0.69	macro avg: 0.57 weighted avg: 0.68	macro avg: 0.59 weighted avg: 0.66
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.7088**	macro avg: 0.67 **weighted**	**macro avg: 0.62** **weighted**	**macro avg: 0.63** **weighted**

			avg: 0.71	**avg: 0.71**	**avg: 0.70**
Gaussian Naive Bayes	Prévia: None	0.4325	macro avg: 0.56 weighted avg: 0.65	macro avg: 0.39 weighted avg: 0.43	macro avg: 0.42 weighted avg: 0.48
KNN	Nº vizinhos: 10 Função distância: minkowski	0.5182	macro avg: 0.51 weighted avg: 0.55	macro avg: 0.39 weighted avg: 0.52	macro avg: 0.41 weighted avg: 0.49
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6681	**macro avg: 0.69** weighted avg: 0.69	macro avg: 0.56 weighted avg: 0.67	macro avg: 0.58 weighted avg: 0.65
BERT		0.5513	macro avg: 0.45 weighted avg: 0.60	macro avg: 0.40 weighted avg: 0.55	macro avg: 0.38 weighted avg: 0.52

Ao comparar com os resultados na Tabela 19, os modelos em geral apresentaram uma pequena queda de desempenho, com exceção dos modelos de regressão logística, que manteve sua sensibilidade e pontuação F1 e aumentou levemente sua acurácia e precisão. O KNN teve um aumento em todas as métricas. Esse aumento de desempenho do KNN pode se dar ao fato do *stemming* reduzir as palavras ao seu *stem,* o que pode vir a separar melhor as vizinhanças e diminuir o valor de distância entre os vizinhos, auxiliando na classificação de novos dados.

O que se destaca é a queda de desempenho do modelo BERT, que apresentou um desempenho muito pior utilizando *stemming*. Essa queda brusca de desempenho pode se dar ao fato do *stemming,* por vezes, acabar transformando palavras gramaticalmente corretas em palavras incorretas ou inexistentes. Como BERT trabalha com o contexto em que as palavras estão inseridas ao se reduzir as palavras ao seu *stem* pode se perder algumas informações sobre o contexto

O modelo que apresentou melhor desempenho com *stemming* foi o *multinomial naive bayes*, que apesar de uma queda de acurácia, continuou sendo uma das maiores, além de conseguir manter sua precisão, sensibilidade e pontuação F1.

As Tabelas 23 e 24 mostram os resultados obtidos executando os modelos com texto bruto e com *stemming*, utilizando 10.000 e 12.943 *features* respectivamente.

Tabela 23. Resultados dos modelos com texto bruto e com *stemming*- 10.000 *features*

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5359	macro avg: 0.67 weighted avg: 0.75	macro avg: 0.49 weighted avg: 0.54	macro avg: 0.51 weighted avg: 0.58
Random Forest	Nº árvores: 100 Critério: entropia	0.6813	macro avg: 0.66 weighted avg: 0.71	macro avg: 0.58 weighted avg: 0.68	macro avg: 0.58 weighted avg: 0.66
Regressão Logística	Solucionador: Lbfgs	0.6793	macro avg: 0.65 weighted avg: 0.69	**macro avg: 0.59** weighted avg: 0.68	**macro avg: 0.59** weighted avg: 0.67
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.7012**	**macro avg: 0.68** weighted avg: 0.73	macro avg: 0.55 **weighted avg: 0.70**	macro avg: 0.56 **weighted avg: 0.68**
Gaussian Naive Bayes	Prévia: None	0.3825	macro avg: 0.61 **weighted avg: 0.74**	macro avg: 0.31 weighted avg: 0.38	macro avg: 0.36 weighted avg: 0.45
KNN	Nº vizinhos: 10 Função distância: minkowski	0.4283	macro avg: 0.37 weighted avg: 0.48	macro avg: 0.33 weighted avg: 0.43	macro avg: 0.31 weighted avg: 0.42
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6574	macro avg: 0.66 weighted avg: 0.67	macro avg: 0.53 weighted avg: 0.66	macro avg: 0.54 weighted avg: 0.63
BERT	--	--	--	--	--

Em geral, com 10.000 *features* os modelos apresentaram uma leve piora de desempenho em relação ao experimento anterior. Em relação a acurácia, praticamente todos os modelos obtiveram um valor menor, com exceção de *random forest* e regressão logística que tiveram uma melhora ínfima. Para a métrica de precisão os modelos de *naive bayes*, *multinomial e gaussian*, *decision tree e random forest* tiveram uma melhora, os modelos restantes tiveram uma queda de

desempenho. Já para as métricas de sensibilidade e pontuação F1, os modelos em geral ou mantiveram desempenho parecido ou pioraram em relação ao experimento anterior.

O modelo que melhor desempenhou nesse cenário foi o *multinomial naive bayes*, tendo a maior acurácia, a melhor média macro em precisão e segunda melhor média ponderada em precisão, além das melhores médias ponderadas em sensibilidade e pontuação F1.

Como explicado anteriormente, a biblioteca utilizada para rodar o modelo BERT não permite a especificação de um número arbitrário de *features*, impossibilitando a execução do BERT nesse experimento.

Tabela 24. Resultados dos modelos com texto bruto e com *stemming*- 12.943 *features*

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5359	macro avg: 0.67 weighted avg: 0.74	macro avg: 0.48 weighted avg: 0.54	macro avg: 0.49 weighted avg: 0.58
Random Forest	Nº árvores: 100 Critério: entropia	0.6813	macro avg: 0.62 weighted avg: 0.69	macro avg: 0.57 weighted avg: 0.68	macro avg: 0.56 weighted avg: 0.65
Regressão Logística	Solucionador: Lbfgs	0.6813	macro avg: 0.65 weighted avg: 0.70	**macro avg: 0.59** weighted avg: 0.68	**macro avg: 0.59** **weighted avg: 0.67**
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.6932**	**macro avg: 0.72** **weighted avg: 0.74**	macro avg: 0.54 **weighted avg: 0.69**	macro avg: 0.55 weighted avg: 0.66
Gaussian Naive Bayes	Prévia: None	0.3825	macro avg: 0.61 **weighted avg: 0.74**	macro avg: 0.31 weighted avg: 0.38	macro avg: 0.36 weighted avg: 0.45
KNN	Nº vizinhos: 10 Função distância: minkowski	0.4203	macro avg: 0.36 weighted avg: 0.47	macro avg: 0.32 weighted avg: 0.42	macro avg: 0.31 weighted avg: 0.41
Support Vector Machine	Kernel: RBF	0.6574	macro avg:	macro avg:	macro avg:

	tamanho de cache: 20MB		0.66 weighted avg: 0.67	0.53 weighted avg: 0.66	0.54 weighted avg: 0.63
BERT		0.5238	macro avg: 0.30 weighted avg: 0.51	macro avg: 0.34 weighted avg: 0.52	macro avg: 0.29 weighted avg: 0.50

Mesmo com o aumento de *features* o desempenho dos modelos não melhorou significativamente. Os únicos modelos, que tiveram melhora em pontos específicos, foram regressão logística e *multinomial naive bayes*, incluindo acurácia para regressão logística e precisão para *multinomial naive bayes*. O restante dos modelos, bem como as demais métricas para os modelos já citados, apresentaram uma queda de desempenho ou no máximo mantiveram o mesmo desempenho anterior, como é o caso de *gaussian naive bayes* e *support vector machine.* Novamente, o BERT foi o modelo que apresentou a maior queda de desempenho, primeiramente por estar trabalhando com um maior número de palavras, sendo um grande número delas com pouco valor para classificação. E também com *stemming,* que pode gerar palavras gramaticalmente incorretas, atrapalhando a interpretabilidade do modelo.

Para o próximo experimento foi utilizada a técnica de *lemmatization*, que consiste na redução das palavras ao seu *lemma*. A diferença para o *stemming* é que na lematização do texto as palavras sempre são reduzidas para uma palavra gramaticalmente correta. A biblioteca utilizada foi a Spacy carregando a gramática do português. A Tabela 25 apresenta os resultados obtidos com a utilização do *lemmatizer* executado sobre os textos só com substantivos.

Tabela 25. Resultados dos modelos só com substantivos e com *lemmatization*

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5096	macro avg: 0.49 weighted avg: 0.56	macro avg: 0.39 weighted avg: 0.51	macro avg: 0.41 weighted avg: 0.49
Random Forest	Nº árvores: 100 Critério: entropia	0.5803	macro avg: 0.58 weighted avg: 0.61	macro avg: 0.45 weighted avg: 0.58	macro avg: 0.47 weighted avg: 0.56

Regressão Logística	Solucionador: Lbfgs	0.5931	**macro avg: 0.65** **weighted avg: 0.66**	macro avg: 0.46 weighted avg: 0.59	macro avg: 0.50 weighted avg: 0.58
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.6445**	macro avg: 0.61 weighted avg: 0.64	**macro avg: 0.53** **weighted avg: 0.64**	**macro avg: 0.54** **weighted avg: 0.62**
Gaussian Naive Bayes	Prévia: None	0.3919	macro avg: 0.50 weighted avg: 0.58	macro avg: 0.36 weighted avg: 0.39	macro avg: 0.38 weighted avg: 0.43
KNN	Nº vizinhos: 10 Função distância: minkowski	0.4540	macro avg: 0.53 weighted avg: 0.57	macro avg: 0.32 weighted avg: 0.45	macro avg: 0.35 weighted avg: 0.44
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6317	macro avg: 0.59 weighted avg: 0.63	macro avg: 0.51 weighted avg: 0.63	macro avg: 0.52 weighted avg: 0.60
BERT		0.5000	macro avg: 0.28 weighted avg: 0.42	macro avg: 0.33 weighted avg: 0.50	macro avg: 0.27 weighted avg: 0.42

Aplicando a *lemmatization*, e comparando com os resultados obtidos com *stemming*, todos os modelos tiveram uma piora no desempenho, com exceção do modelo de KNN, que obteve um aumento apenas na precisão.

Essa queda de desempenho dos modelos é interessante de ser observada, uma vez que a redução das palavras ao seu *lemma* sempre resulta em palavras existentes dentro da gramática da língua analisada, o que deveria auxiliar na classificação. Já com o *stemming* essa redução pode resultar em palavras incorretas e inexistentes, ou até mesmo em palavras existentes mas sem valor para a classificação, como artigos.

Nesse experimento o modelo com o melhor desempenho foi novamente o *multinomial naive bayes*, apresentando a melhor acurácia, sensibilidade e pontuação F1. Já a melhor precisão foi obtida pelo método de regressão logística.

A Tabela 26 apresenta os resultados dos modelos utilizando o texto bruto com *lemmatization*, o que gerou um vocabulário com 7.354 *features.*

Tabela 26. Resultados dos modelos com texto bruto e com *lemmatization* - 7.354 *features*

Modelo	Parâmetros	Métricas de Desempenho			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.5259	macro avg: 0.58 weighted avg: 0.65	macro avg: 0.47 weighted avg: 0.53	macro avg: 0.47 weighted avg: 0.54
Random Forest	Nº árvores: 100 Critério: entropia	0.6175	macro avg: 0.63 weighted avg: 0.67	macro avg: 0.51 weighted avg: 0.62	macro avg: 0.51 weighted avg: 0.59
Regressão Logística	Solucionador: Lbfgs	0.6454	macro avg: 0.64 weighted avg: 0.69	macro avg: 0.53 weighted avg: 0.65	macro avg: 0.54 weighted avg: 0.63
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	**0.6713**	macro avg: 0.64 weighted avg: 0.69	**macro avg: 0.56** **weighted avg: 0.67**	**macro avg: 0.57** **weighted avg: 0.66**
Gaussian Naive Bayes	Prévia: None	0.4243	macro avg: 0.60 **weighted avg: 0.72**	macro avg: 0.36 weighted avg: 0.42	macro avg: 0.40 weighted avg: 0.49
KNN	Nº vizinhos: 10 Função distância: minkowski	0.4661	macro avg: 0.46 weighted avg: 0.56	macro avg: 0.37 weighted avg: 0.47	macro avg: 0.36 weighted avg: 0.45
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6434	**macro avg: 0.67** weighted avg: 0.69	macro avg: 0.52 weighted avg: 0.64	macro avg: 0.54 weighted avg: 0.63
BERT		0.5238	macro avg: 0.39 weighted avg: 0.50	macro avg: 0.36 weighted avg: 0.52	macro avg: 0.34 weighted avg: 0.47

O aumento no número de *features* se converteu em um maior ganho de informação para os modelos, que conseguiram desempenhar melhor quando comparados ao experimento anterior utilizando apenas substantivos. KNN o único modelo que apresentou uma queda de desempenho, mas apenas na métrica de precisão.Assim como no experimento anterior, *multinomial naive bayes* foi o modelo

que apresentou o melhor desempenho, tendo a melhor acurácia, sensibilidade e pontuação F1, e apresentando uma das melhores precisões também.

O próximo experimento consistiu na aplicação do método de decomposição de matrizes LSA, ou *Latent Semantic Analysis.* Esse método recebe como entrada uma matriz, no formato documento x palavras, preenchida com os valores de frequência das palavras em cada texto e retorna uma matriz com K dimensões, sendo K arbitrário, com os valores singulares ordenados do maior para o menor. Para a tarefa de LSA, segundo Wiermer-Hastings (2004) o número de dimensões, ou *features* nesse contexto, necessárias para capturar o significado dos textos resume-se a 300.

Para a implementação e execução do LSA foi utilizada a classe *Truncated SVD* da biblioteca Scikit-learn, sendo 300 o número de dimensões da matriz resultante. Os resultados obtidos utilizando somente substantivos e LSA podem ser vistos na Tabela 27.

Tabela 27. Resultados dos modelos só com substantivos e LSA - 300 dimensões

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.4261	macro avg: 0.42 weighted avg: 0.51	macro avg: 0.34 weighted avg: 0.43	macro avg: 0.35 weighted avg: 0.43
Random Forest	Nº árvores: 100 Critério: entropia	0.6531	macro avg: 0.62 weighted avg: 0.66	macro avg: 0.55 weighted avg: 0.65	macro avg: 0.55 weighted avg: 0.63
Regressão Logística	Solucionador: Lbfgs	**0.6617**	**macro avg: 0.66** **weighted avg: 0.68**	**macro avg: 0.56** **weighted avg: 0.66**	**macro avg: 0.58** **weighted avg: 0.64**
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	--	--	--	--
Gaussian Naive Bayes	Prévia: None	0.2077	macro avg: 0.32 weighted avg: 0.46	macro avg: 0.18 weighted avg: 0.21	macro avg: 0.17 weighted avg: 0.22
KNN	Nº vizinhos: 10 Função distância: minkowski	0.5653	macro avg: 0.48 weighted avg:	macro avg: 0.46 weighted avg:	macro avg: 0.44 weighted avg:

			0.56	0.57	0.54
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.6467	macro avg: 0.64 weighted avg: 0.66	macro avg: 0.53 weighted avg: 0.65	macro avg: 0.55 weighted avg: 0.62
BERT	--	--	--	--	--

Para esse experimento não foi possível executar os modelos de *multinomial naive bayes* e BERT. Como *multinomial naive bayes* trabalha com valores de frequência de palavras, não é possível executá-lo com LSA pois a matriz resultante pode contar valores negativos, o que não faz sentido num contexto de frequência de palavras, portanto o modelo não aceita valores negativos de frequência. Já a execução do BERT com LSA não é possível, pois o BERT aceita como entrada apenas textos, uma vez que possui um tokentizador interno, e portanto não é possível passar como entrada a matriz resultante do processo de LSA.

Analisando os resultados e comparando-os com os resultados da Tabela 19 nota-se uma queda de desempenho dos modelos, sendo KNN o único modelo que obteve uma melhora de performance em acurácia, sensibilidade e pontuação F1. Uma possível causa para esse efeito pode ser a dificuldade que alguns métodos podem ter em trabalhar com os valores singulares gerados pelo processo de LSA. O modelo com o melhor desempenho foi o de regressão logística, liderando os modelos em todas as métricas analisadas.

As Tabelas 28 e 29 apresentam os resultados obtidos ao executar os modelos com texto bruto e com LSA, com 10.000 e 17.260 *features*, respectivamente.

Tabela 28. Resultados dos modelos com texto bruto 10.000 *features* e LSA - 300 dimensões

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.3347	macro avg: 0.33 weighted avg: 0.44	macro avg: 0.28 weighted avg: 0.33	macro avg: 0.27 weighted avg: 0.35
Random Forest	Nº árvores: 100 Critério: entropia	0.5657	macro avg: 0.49 weighted avg: 0.56	macro avg: 0.44 weighted avg: 0.57	macro avg: 0.44 weighted avg: 0.53

Regressão Logística	Solucionador: Lbfgs	**0.6295**	**macro avg: 0.55** **weighted avg: 0.64**	**macro avg: 0.50** **weighted avg: 0.63**	**macro avg: 0.50** **weighted avg: 0.62**
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	--	--	--	--
Gaussian Naive Bayes	Prévia: None	0.0857	macro avg: 0.20 weighted avg: 0.26	macro avg: 0.10 weighted avg: 0.09	macro avg: 0.07 weighted avg: 0.08
KNN	Nº vizinhos: 10 Função distância: minkowski	0.3805	macro avg: 0.28 weighted avg: 0.38	macro avg: 0.28 weighted avg: 0.38	macro avg: 0.26 weighted avg: 0.36
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.5837	macro avg: 0.52 weighted avg: 0.58	macro avg: 0.45 weighted avg: 0.58	macro avg: 0.44 weighted avg: 0.55
BERT	--	--	--	--	--

Com 10.000 *features* todos os modelos tiveram uma queda de desempenho em todas as métricas analisadas. Uma possível causa desse efeito é o fato de que com o aumento do número de *features* muitas palavras sem valor para a classificação foram adicionadas e, por algumas dessas palavras possuírem alta frequência nos textos, após a execução do LSA elas foram colocadas entre as 300 *features* mais importantes. Com a adição dessas *features "*ruins", a tarefa de classificação fica prejudicada.

Novamente o modelo com melhor desempenho é o de regressão logística, apesar da queda no valor de suas métricas elas ainda são as melhores em comparação com os outros modelos.

Tabela 29. Resultados dos modelos com texto bruto 17.260 *features* e LSA - 300 dimensões

Modelo	**Parâmetros**	**Métricas de Desempenho**			
		acurácia	**precisão**	**sensibilidade**	**pontuação F1**
Decision Tree	Critério: entropia Divisor: melhor	0.3327	macro avg: 0.33 weighted avg: 0.44	macro avg: 0.26 weighted avg: 0.33	macro avg: 0.26 weighted avg: 0.35

Random Forest	Nº árvores: 100 Critério: entropia	0.5737	macro avg: 0.50 weighted avg: 0.58	macro avg: 0.44 weighted avg: 0.57	macro avg: 0.43 weighted avg: 0.54
Regressão Logística	Solucionador: Lbfgs	**0.6335**	**macro avg: 0.56** **weighted avg: 0.65**	**macro avg: 0.50** **weighted avg: 0.63**	**macro avg: 0.51** **weighted avg: 0.63**
Multinomial Naive Bayes	Parâmetro de suavização: 1.0	--	--	--	--
Gaussian Naive Bayes	Prévia: None	0.0916	macro avg: 0.20 weighted avg: 0.25	macro avg: 0.11 weighted avg: 0.09	macro avg: 0.07 weighted avg: 0.08
KNN	Nº vizinhos: 10 Função distância: minkowski	0.3725	macro avg: 0.28 weighted avg: 0.39	macro avg: 0.27 weighted avg: 0.37	macro avg: 0.25 weighted avg: 0.35
Support Vector Machine	Kernel: RBF tamanho de cache: 20MB	0.5837	macro avg: 0.52 weighted avg: 0.58	macro avg: 0.45 weighted avg: 0.58	macro avg: 0.44 weighted avg: 0.55
BERT	--	--	--	--	--

Com 17.260 *features* o desempenho dos modelos não se alterou muito, alguns modelos tiveram uma pequena melhora em algumas métricas, como *random forest* e regressão logística, e outros uma leve piora, como *decision tree* e KNN. Essa variação mínima no desempenho dos modelos pode se dar ao fato das *features* adicionadas não serem tão impactantes para a classificação, tendo pouca frequência nos textos e portanto sendo filtradas na execução do LSA.

Assim como nos experimentos anteriores com LSA, o modelo de regressão logística obteve os melhores resultados, tendo um leve aumento na sua acurácia, precisão e pontuação F1, e mantendo o seu valor de sensibilidade. Esse desempenho superior se dá ao fato da regressão logística trabalhar com probabilidades de um evento ocorrer de acordo com o valor de variáveis independentes. Como após o LSA as *features* restantes são as que possuem maior peso para classificação em alguma categoria a regressão logística se beneficia desse fator para calcular melhor a probabilidade de um texto pertencer a uma certa categoria.

4.5 Discussão

Os experimentos de modelagem de tópicos realizados com LDA demonstraram uma grande dificuldade do modelo em gerar tópicos cujo tema seja facilmente distinguível um dos outros. Diversos tópicos poderiam ser associados a um mesmo tema, o que não é interessante para o contexto deste trabalho, portanto tanto a modelagem de tópicos quanto o modelo de LDA foram descartados como opção para a resolução da tarefa de classificação de aplicativos desenvolvidos no App Inventor.

Em relação aos experimentos de classificação utilizando modelos de *machine learning* e *deep learning* os melhores resultados obtidos em cada um dos experimentos realizados são resumidos na Tabela 30. Para esta comparação foi utilizada a métrica de acurácia como principal critério de desempenho, uma vez que ela mede a porcentagem de acertos na classificação, o que no contexto de classificação de aplicativos é muito importante.

Tabela 30. Resultados do melhor modelo em cada experimento

Modelo	**Métricas**				**Experimento**
	Acurácia	Precisão	Sensibilidade	Pontuação F1	
Multinomial Naive Bayes	**0.7109**	**0.71**	**0.71**	**0.70**	**Só substantivos**
Random Forest	0.7052	0.73	0.71	0.68	Texto Bruto 10.000 features
Multinomial Naive Bayes	0.6932	0.73	0.69	0.67	Texto Bruto 17.260 features
Multinomial Naive Bayes	0.7088	0.71	0.71	0.70	Só substantivos + stemming
Multinomial Naive Bayes	0.7012	0.73	0.70	0.68	Texto Bruto 10.000 features + stemming
Multinomial Naive Bayes	0.6932	0.74	0.69	0.66	Texto Bruto 12.943 features + stemming
Multinomial Naive Bayes	0.6445	0.64	0.64	0.62	Só substantivos + lemmatization
Multinomial Naive Bayes	0.6713	0.69	0.67	0.66	Texto Bruto 7.354 features + lemmatization
Regressão Logística	0.6617	0.68	0.66	0.64	Só substantivos + LSA
Regressão Logística	0.6295	0.64	0.63	0.62	Texto Bruto 10.000 features + LSA
Regressão Logística	0.6335	0.65	0.63	0.63	Texto Bruto 17.260 features + LSA

Analisando, de forma geral, os resultados, o modelo que mais se destaca é o *multinomial naive bayes*, apresentando o melhor desempenho na maioria dos experimentos realizados. Comparando os experimentos é possível notar um maior desempenho dos modelos ao utilizar o texto apenas com substantivos, com exceção dos experimentos utilizando *lemmatization*.

Dentre os resultados obtidos, destaca-se, o *multinomial naive bayes* no experimento utilizando o texto apenas com substantivos, obtendo a maior acurácia (0.7109), a maior sensibilidade (0.71) e maior pontuação F1 (0.70) dentre todos os modelos em qualquer um dos experimentos. A Figura 10 apresenta a matriz de confusão obtida pelo modelo *Multinomial Naive Bayes* no experimento utilizando os textos só com substantivos.

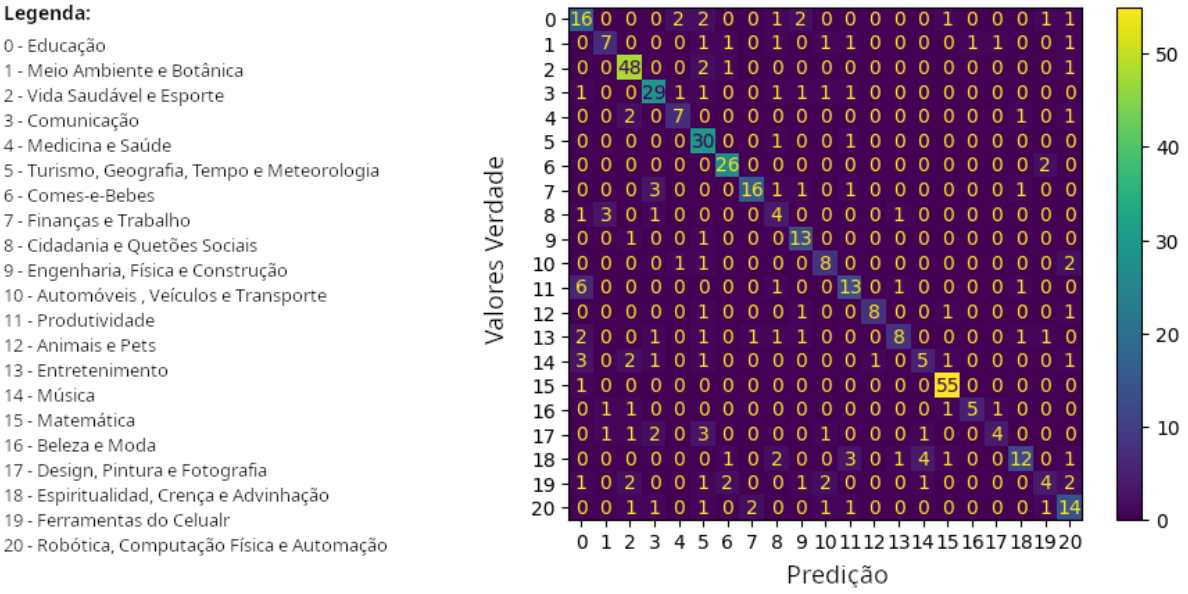

Figura 10. Matriz de confusão do modelo *Multinomial Naive Bayes* só com substantivos

Analisando a matriz de confusão é possível visualizar que o modelo apresenta mais facilidade ao classificar aplicativos de, por exemplo, Comes-e-Bebes; Animais e Pets; Matemática, enquanto apresenta certa dificuldade ao classificar aplicativos de Espiritualidade, crença e adivinhação; Música; Cidadania e questões sociais. Algumas estratégias podem ser usadas para melhorar o desempenho do modelo ao classificar aplicativos dessas categorias, como o aumento do número de aplicativos pertencentes a essas categorias dentro do conjunto de dados.

Destaca-se também o desempenho do *multinomial naive bayes* no experimento utilizando textos só com substantivos e *stemming*, alcançando a segunda melhor acurácia (0.7088) e empatando nas demais métricas. O terceiro melhor desempenho foi alcançado pelo modelo de *random forest* no experimento utilizando texto bruto com 10.000 *features*, obtendo a segunda melhor acurácia, precisão e pontuação F1, e empatando com a melhor sensibilidade.

Devido a natureza dos textos utilizados, modelos como o BERT não apresentam um desempenho satisfatório, uma vez que o BERT trabalha analisando o contexto em que as palavras estão e os textos extraídos dos aplicativos, em geral tratam-se de palavras soltas ou frases muito curtas sem um contexto para ser analisado, o que dificulta o desempenho desse modelo.

Com base nos resultados alcançados é possível destacar o modelo *multinomial naive bayes* como melhor opção para a classificação de textos curtos, como os encontrados em aplicativos móveis. Quanto à etapa de pré-processamento, observa-se que a utilização de textos só com substantivos tende a levar o modelo utilizado a um melhor desempenho na tarefa de classificação.

Ameaças a validade. Devido a característica empírica deste estudo existem diversas ameaças a sua validade. Em relação ao tamanho da amostra, considera-se o conjunto de dados, com 1.672 apps, suficiente para obter os primeiros resultados válidos. Em relação ao viés de seleção, foram utilizados aplicativos que abordam diversos temas, para que assim, a amostra fosse o mais abrangente possível. Porém devido a limitação do idioma, e ao fato de alguns temas serem mais recorrentemente abordados fez com que, apesar de conter exemplos de apps para todas as categorias utilizadas, a quantidade de aplicativos para cada categoria não foi balanceada, o que pode ter afetado o aprendizado dos modelos.

Para evitar o viés de pré-processamento, foram aplicadas etapas padronizadas de pré-processamento dos dados em cada experimento, inclusive testando várias alternativas. Quanto ao treinamento dos modelos escolhidos, foram utilizados os mesmo parâmetros de construção de conjunto de treinamento e validação para os modelos de *machine learning. Já* para o modelo de *deep learning* os parâmetros foram um pouco alterados devido a natureza do modelo. Devido a especificidade de alguns experimentos, alguns modelos não puderam ser executados, como foi o caso do modelo BERT em experimentos com um número específico de *features,* ou do

multinomial naive bayes com LSA. Já para evitar viés de métricas de avaliação, foram utilizadas diversas métricas que avaliam vários aspectos de desempenho dos modelos, essas métricas foram calculadas para todos os modelos utilizados e em cada experimento realizado.

5. CONCLUSÃO

Como resultado do presente trabalho foram analisados conceitos de originalidade pertinentes em um contexto de avaliação de originalidade em aplicativos criados no App Inventor. Foram também sintetizados como conceitos de *machine learning* e *deep learning*, bem como conceitos de pré-processamento de dados, necessários para a realização dos experimentos propostos (OE1). Foi realizado um levantamento do estado da arte em volta da utilização de modelos de *machine learning* e *deep learning* na classificação de textos (OE2) mostrando que atualmente não existem abordagens que utilizam os textos presentes nos aplicativos para a classificação desses apps. Com base nesse levantamento foram selecionados diversos modelos de classificação que foram treinados, avaliados e comparados utilizando um conjunto de textos extraídos de apps App Inventor. Os resultados obtidos indicam o modelo de *multinomial naive bayes* como a melhor opção para a classificação de aplicativos por meio dos textos extraídos destes aplicativos (OE3).

Assim, o principal impacto científico do presente trabalho é o conhecimento gerado indicando a(s) melhor(es) opções para a classificação de aplicativos com base nos textos presentes nele. Isto fornece uma base para o desenvolvimento de ferramentas de automatização de classificação de aplicativos móveis, como por exemplo, os criados em App Inventor. Assim, contribuindo com a avaliação automatizada da aprendizagem de criatividade, o que também representa uma contribuição social no contexto do ensino de computação na Educação Básica.

Contudo ainda existem possibilidades a serem exploradas dentro do contexto de classificação de apps por meio de seus textos. Um possível trabalho futuro seria a realização dos experimentos com base em um conjunto de dados maior e assim averiguar melhor o desempenho dos modelos abordados, principalmente BERT, que demanda uma grande quantidade de dados. Outra opção é a exploração de diferentes técnicas de pré-processamento do texto extraído, o que pode influenciar no desempenho dos modelos. Outra possibilidade é a associação dos termos mais descritivos dentro dos textos à ontologias que representam cada classe. Assim como a utilização de outros modelos de classificação de textos, como BERTimbau, que é uma versão do BERT para português, ou até mesmo outras estratégias de aprendizado como *Few-Shot Learning* ou *Zero-Shot Learning*.

REFERÊNCIAS

AL KILANI, Nadeem et al. Automatic Classification of Apps Reviews for Requirement Engineering: Exploring the Customers Need from Healthcare Applications. In: Anais da Sixth international Conference on Social Networks Analysis, Management and Security (SNAMS). IEEE, p. 541-548, 2019.

ALMEIDA, Alex Marino Gonçalves; LEITE, Natália Aparecida Beirão; RAMOS, Ricardo Fabrício. Recuperação da Informação e a Importância do Pré-Processamento. Retec: Revista de Tecnologias, Fatec, Ourinhos - SP, 2019.

ALVES, Nathalia da Cruz et al. Uma Proposta de Avaliação da Originalidade do Produto no Ensino de Algoritmos e Programação na Educação Básica. In: Anais do Simpósio Brasileiro de Informática na Educação, Natal, Brasil, 2020.

ALVES, Nathalia da Cruz; VON WANGENHEIM, Christiane Gresse; MARTINS-PACHECO, Lúcia Helena. Assessing Product Creativity in Computing Education: A Systematic Mapping Study. Informatics in Education, vol. 20 no. 1, 2021.

ANCHIÊTA, Rafael et al. PLN: Das Técnicas Tradicionais aos Modelos de Deep Learning. Sociedade Brasileira de Computação, 2021.

APPLE. Choosing a Category. Disponível em:<https://developer.apple.com/app-store/categories/>. 2022.

BERRAR, Daniel. Bayes' Theorem and Naive Bayes Classifier. Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics, v. 403, 2018.

BLEI, David M; NG, Andrew Y; JORDAN, Michael I. Latent Dirichlet Allocation. Journal of Machine Learning Research 3. p. 993-1022, 2003.

BREIMAN, Leo. Random Forests. Machine learning, v. 45, n. 1, p. 5-32, 2001.

CARDOSO, Beatriz, 9 em cada 10 brasileiros usam celular Android, diz relatório do Google. Techtudo. 2020. Disponível em: https://www.techtudo.com.br/noticias/2020/09/9-em-cada-10-brasileiros-usam-celular-android-diz-relatorio-do-google.ghtml.

CARVALHO, Deborah Ribeiro. Árvore de Decisão/ Algoritmo Genético para Tratar o Problema de Pequenos Disjuntos em Classificação de Dados. 173 f. Tese (Doutorado) - Curso de Engenharia Civil, Universidade Federal do Rio de Janeiro, Rio de Janeiro - RJ, 2005.

CHEN, Xi; ISHWARAN, Hemant. Random Forests for Genomic Data Analysis. Genomics, v. 99, n. 6, p. 323-329, 2012.

COVER, Thomas; HART, Peter. Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, v. 13, n. 1, p. 21-27, 1967.

CRIATIVIDADE. *In:* DICIO, Dicionário Online de Português. 7Graus. 2020. Disponível em:<https://www.dicio.com.br/criatividade/>.

CUNNINGHAM, Padraig; DELANY, Sarah Jane. K - Nearest Neighbour Classifiers - A Tutorial. ACM Computing Surveys (CSUR), v. 54, n. 6, p. 1-25, 2021.

DASARI, Durga Bhavani; RAO, Venu Gopala. Text Categorization and Machine Learning Methods: Current State Of The Art. Global Journal of Computer Science and Technology, *[S. l.]*, v. 12, n. C11, p. 37–46, 2012.

DENG, Xuelian et al. Feature Selection for Text Classification: A review. Multimedia Tools and Applications, v. 78, n. 3, p. 3797-3816, 2019.

DEVLIN, Jacob; CHANG, Ming-Wei. Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing. Google AI Blog, 2018. Disponível em: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html.

DEVLIN, Jacob et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Cornell University, arXiv preprint arXiv:1810.04805, 2018.

DONG, Feng et al. ClassifyDroid: Large Scale Android Applications Classification Using Semi-Supervised Multinomial Naive Bayes. In: 2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS). IEEE, p. 77-81, 2016.

DUMAIS, Susan T. Latent Semantic Analysis. Annu. Rev. Inf. Sci. Technol., v. 38, n. 1, p. 188-230, 2004.

FALEIROS, Thiago de Paulo; LOPES, Alneu de Andrade. Modelos Probabilísticos de Tópicos: Desvendando o Latent Dirichlet Allocation. Relatório Técnico. Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP, 2016.

FIGUEIRA, Cleonis Viater. Modelos de Regressão Logística. 149 f. Dissertação (Mestrado) - Curso de Matemática, Universidade Federal do Rio Grande do Sul, Porto Alegre - RS, 2006.

FRANCESCHI, Pietro Reinheimer de. Modelagens Preditivas de *CHURN*: O caso do Banco do Brasil. 121 f. Dissertação (Mestrado) - Curso de Programa de Pós-Graduação em Gestão de Negócios, Universidade do Vale do Rio dos Sinos - Unisinos, Porto Alegre - RS, 2019.

GAL, Lilach et al. Suggesting a Log-Based Creativity Measurement for Online Programming Learning Environment. In Proc. of the 4th Conference on Learning at Scale, 273-277. ACM, 2017.

GARCÊS, Soraia Fernandes. A Multidimensionalidade da Criatividade A Pessoa, o Processo, o Produto e o Ambiente Criativo no Ensino Superior. 212 f. Tese (Doutorado) - Curso de Psicologia, Universidade da Madeira, Madeira, 2013.

GHOSAL, Tirthanka et al. Novelty Goes Deep. A Deep Neural Solution To Document Level Novelty Detection. Proc. of the 27th Int. Conference on Computational Linguistics, Santa Fe, NM, USA, 2018.

GOOGLE. Choose a Category and Tags for your App or Game. Disponível em:<https://support.google.com/googleplay/android-developer/answer/9859673?hl=en#zippy=%2Capps%2Cjogos>. 2022

GONZALEZ, Leandro de Azevedo. Regressão Logística e suas Aplicações. 46 f. TCC (Graduação) - Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís - MA, 2018.

GONZÁLEZ-CARVAJAL, Santiago; GARRIDO-MERCHÁN, Eduardo C. Comparing BERT Against Traditional Machine Learning Text Classification. Cornell University, arXiv preprint arXiv:2005.13012, 2020.

GUZMAN, Emitza et al. Ensemble methods for app review classification: An approach for software evolution (n). In: 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, p. 771-776, 2015.

JUAN, Alfons; NEY, Hermann. Reversing and Smoothing the Multinomial Naive Bayes Text Classifier. In: Pris. p. 200-212, 2002.

KATARIA, Aman; SINGH, M. D. A Review of Data Classification Using K-Nearest Neighbour Algorithm. International Journal of Emerging Technology and Advanced Engineering, v. 3, n. 6, p. 354-360, 2013.

KASZUBOWSKI, Erikson. Modelo de Tópicos para Associações Livres. 213 f. Tese (Doutorado) - Curso de Psicologia, Universidade Federal de Santa Catarina, Florianópolis - SC, 2016.

KHYANI, Divya et al. An Interpretation of Lemmatization and Stemming in Natural Language Processing. Journal of University of Shanghai for Science and Technology, vol 22, p. 350-357, 2021.

KLEINBAUM, David G; KLEIN, Mitchel. Logistic Regression. New York: Springer-Verlag, 2002.

KOTSIANTIS, Sotiris B. et al. Supervised Machine Learning: A review of Classification Techniques. Emerging Artificial Intelligence Applications in Computer Engineering. v. 160, n. 1, p. 3-24, 2007.

KULTZAK, Adriano Francisco. Categorização de Textos Utilizando Algoritmos de Aprendizagem de Máquina com Weka. 74 f. TCC (Graduação) - Curso de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Ponta Grossa - PR, 2016.

LORENA, Ana Carolina; CARVALHO, André C. P. L. F. de. Estratégias para a Combinação de Classificadores Binários em Soluções Multiclasses. Revista de Informática Teórica e Aplicada, v. 25, n. 2, p.65-86, 2008.

LORENA, Ana Carolina; CARVALHO, André C. P. L. F. de. Introdução às Máquinas de Vetores Suporte. Relatório Técnico. Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos - SP, 2003.

LORENA, Ana Carolina; CARVALHO, André C. P. L. F. de. Uma Introdução às Support Vector Machines. Revista de Informática Teórica e Aplicada, v. 14, n. 2, p. 43-67, 2007.

MAMMONE, Alessia; TURCHI, Marco; CRISTIANINI, Nello. Support Vector Machines. Willey Interdisciplinary Reviews: Computational Statistics, v. 1, n. 3, p. 283-289, 2009.

MEC. Base Nacional Comum Curricular (BNCC). Disponível em: http://basenacionalcomum.mec.gov.br/.

MINAEE, Shervin et al. Deep Learning-Based Text Classification: A Comprehensive Review. ACM Computing Surveys (CSUR), v. 54, n. 3, p. 1-40, 2021.

MUSTAFARAJ, Eni; TURBAK, Franklyn; SVANBERG, Maja. Identifying Original Projects in App Inventor. In Proc. Florida Artificial Intelligence Research Society Conference, 2017.

OECD. PISA 2021 Creative Thinking Framework, 2019. Disponível em: https://www.oecd.org/pisa/publications/PISA2021-Creative-Thinking-Framework.pdf.

OLIVEIRA, Ewerton Lopes Silva de. Uma Investigação de Aspectos da Classificação de Tópicos para Textos Curtos. 82 f. Dissertação (Doutorado) - Curso de Ciência da Computação, Universidade Federal da Paraíba, João Pessoa - PB, 2015.

ONAN, Aytuğ et al. Ensemble of Keyword Extraction Methods and Classifiers in Text Classification. Expert Systems with Applications, v. 57, p. 232-247, 2016.

P21. 21st Century Skills, 2020. Disponível em: www.p21.org.

PATTON, Evan W. et al. MIT App Inventor: Objectives, Design, and Development. Computational thinking education, p. 31-49, 2019.

PETERSEN, Kai; VAKKALANKA, Sairam; KUZNIARZ, Ludwik. Guidelines for Conducting Systematic Mapping Studies in Software Engineering: An Update. Information and Software Technology, 64, 1–18, 2015.

POWERS, David M. W. Evaluation: From Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation. arXiv preprint arXiv:2010.16061, 2020.

QIU, Haoyu et al. Exploring Multiple Genres Text Classification: Classifying 61 Genres of Mobile App Description Based on Naïve Bayes and Count Vectorizer. In: 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI). IEEE, p. 156-162, 2022.

RIBEIRO, Barata; BIANCHI, Marcelo. Modelagem de Tópicos e Interpretabilidade: Uma Proposta de Visualização de Resultados Implementada em D3.js. 87 f. Dissertação (Mestrado) - Curso de Matemática, Fundação Getúlio Vargas- Fgv, Rio de Janeiro - RJ, 2020.

RIBEIRO, Juliana P. et al. Dinâmicas com App Inventor no Apoio ao Aprendizado e no Ensino de Programação. In: Anais do XXII Workshop de Informática na Escola. SBC. p. 271-280, 2016.

RIQUETI, Giovanna et al. Classificando Perfis de Longevidade de Bases de Dados Longitudinais Usando Floresta Aleatória. VI Symposium on Knowledge Discovery, Mining and Learning, 2018.

RISH, Irina et al. An Empirical Study of the Naive Bayes Classifier. In: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence. p. 41-46, 2001.

RIZZI, Claudia Brandelero; VALIATI, João Francisco; ENGEL, Paulo Martins. Uma Proposta para Categorização de Textos por uma Rede Neural. V Congresso Brasileiro de Redes Neurais - VI Escola de Redes Neurais, Rio de Janeiro - RJ, 2001.

RHODES, Mel. An Analysis of Creativity. The Phi Delta Kappan, v. 42, n. 7, p. 305-310, 1961.

SANTOS, Keila Barbosa Costa dos. Categorização de Textos por Aprendizagem de Máquina. 86 f. Dissertação (Mestrado) - Curso de Modelagem Computacional de Conhecimento, Universidade Federal de Alagoas, Maceió - AL, 2019.

SACRAMENTO, Daniel. Árvore de Decisão: Entenda esse Algoritmo de Machine Learning. Tera, 2021. Disponível em: https://blog.somostera.com/data-science/arvores-de-decisao.

SCAICO, Pasqueline Dantas; et al. Ensino de Programação no Ensino Médio: Uma Abordagem Orientada ao Design com a Linguagem Scratch. Revista Brasileira de Informática na Educação, 21(2), 2013.

SCARPA, Alice Duarte. Técnicas de Processamento de Linguagem Natural Aplicadas às Ciências Sociais. 68 f. Dissertação (Mestrado) - Curso de Matemática, Fundação Getúlio Vargas- Fgv, Rio de Janeiro - RJ, 2017.

SCHMID, Helmut. Part-Of-Speech Tagging with Neural Networks. Cornell University, arXiv preprint cmp-lg/9410018, 1994.

SHAH, Foram P.; PATEL, Vibah. A Review on Feature Selection and Feature Extraction for Text Classification. In: 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET). Chennai, India, 2016, p. 2264-2268, doi: 10.1109/WiSPNET.2016.7566545.

SHIBA, Marcelo Hiroshi et al. Classificação de Imagens de Sensoriamento Remoto pela Aprendizagem por Árvore de Decisão: Uma Avaliação de Desempenho. Anais do XII Simpósio Brasileiro de Sensoriamento Remoto, Goiânia - GO, Brasil.p. 4319-4326, 2005.

SILVA, Cassiana Fagundes da. Uso de Informações Linguísticas na Etapa de Pré-Processamento em Mineração de Textos. 109 f. Dissertação (Mestrado) - Curso de Ciências Exatas e Tecnológicas, Universidade do Vale do Rio dos Sinos, São Leopoldo - RS, 2004.

SILVA, Cassiana Fagundes da; VIEIRA, Renata; OSÓRIO, Fernando Santos. Uso de Informações Lingüísticas em Categorização de Textos Utilizando Redes Neurais Artificiais. VIII Simpósio Brasileiro de Redes Neurais,(2004a), p. 1-6, 2004.

SILVA, Daniel Filipe Baptista Ferreira da. Pré-Processamento de Dados e Comparação entre Algoritmos de Machine Learning para Análise Preditiva de Falhas em Linhas de Produção para o Controlo. 78 f. Dissertação (Mestrado) - Curso de Engenharia Informática, Instituto Superior de Engenharia do Porto, Porto - Portugal, 2021.

SOKOLOVA, Marina; LAPALME, Guy. A Systematic Analysis of Performance Measures for Classification Tasks. Information Processing & Management, v. 45, n. 4, p. 427-437, 2009.

SOUMIK, Md Muhtasim Jawad et al. Employing Machine Learning Techniques on Sentiment Analysis of Google Play Store Bangla Reviews. In: 2019 22nd International Conference on Computer and Information Technology (ICCIT). IEEE, p. 1-5, 2019.

SOUSA, Luana da Silva. Avaliação do Efeito de Normalização de Corpus na Coerência de Tópicos Extraídos Usando Latent Dirichlet Allocation. 77 f. Dissertação (Doutorado) - Curso de Ciência da Informação, Universidade Federal de Santa Catarina, Florianópolis - SC, 2022.

SOUZA , Marcos de; SOUZA , Renato Rocha. Modelagem de Tópicos: Resumir e Organizar Corpus de Dados por Meio de Algoritmos de Aprendizagem de Máquina. Múltiplos Olhares em Ciência da Informação, *[S. l.]*, v. 9, n. 2, 2020.

TRIANTAFYLLOU, Ioannis et al. How to Utilize my App Reviews? A Novel Topics Extraction Machine Learning Schema for Strategic Business Purposes. Entropy, v. 22, n. 11, p. 1310, 2020.

TURBAK, Franklyn et al. Work in progress: Identifying and Analyzing Original Projects in an Open-Ended Blocks Programming Environment. In International Conference on Distributed Multimedia Systems, Visual Languages and Sentient Systems, Wyndham Pittsburgh/EUA, 2017.

WIEMER-HASTINGS, Peter; WIEMER-HASTINGS, K.; GRAESSER, A. Latent Semantic Analysis. Proceedings of the 16th International Joint Conference on Artificial intelligence. p. 1-14, 2004.

YANG, Wenchuan et al. Design and Implementation of Application Classification Based on Deep Learning. In: 2019 Chinese Automation Congress (CAC). IEEE, p. 4821-4826, 2019.

ZEBARI, Rizgar R. et al. A comprehensive Review of Dimensionality Reduction Techniques for Feature Selection and Feature Extraction. Journal of Applied Science andTechnology Trends, v. 01, n. 02, p. 56-70, 2020.

ZHANG, Hua et al. Research on Android Multi-classification Based on Text. In: Journal of Physics: Conference Series. IOP Publishing, p. 012049, 2021.

Apêndice 1: Artigo científico originado deste trabalho

Análise e Comparação de Técnicas de Classificação de Tópicos de Aplicativos Móveis Criados com App Inventor

Pablo Daniel Riveros Strapasson
Dep. Informática e Estatística
Universidade Federal de Santa Catarina
Florianópolis, SC - Brasil
pablo.daniel.riveros.strapasson@grad.ufsc.br

Abstract: *Creativity is considered one of the main skills of the 21st century. While primarily associated with the arts, creativity can also be developed in computing education with mobile app development with App Inventor. Originality is one of the characteristics that define a product as creative, and it can be analyzed through the topics covered by the apps. This topic classification can be automated from texts present in applications by adopting machine learning algorithms. When comparing the performance of several of these text classification techniques, it was found that the best model is the Multinomial Naive Bayes, with a success rate of 71.09%. These results can be used in research aimed at automating the assessment of creativity in the context of teaching computing.*

Resumo: *A criatividade é considerada uma das principais habilidades do século XXI. Embora seja associada principalmente a artes, a criatividade também pode ser desenvolvida no ensino de computação com o desenvolvimento de aplicativos móveis com App Inventor. A originalidade é uma das características que definem um produto como criativo, e ela pode ser analisada por meio dos tópicos abordados pelos apps. Essa classificação de tópicos pode ser automatizada a partir de textos presentes nos aplicativos adotando algoritmos de machine learning. Ao comparar o desempenho de várias destas técnicas de classificação de textos obteve-se que o melhor modelo é o Multinomial Naive Bayes, com uma taxa de acerto de 71,09%. Estes resultados podem ser utilizados em pesquisas voltadas à automação da avaliação da criatividade no contexto do ensino de computação.*

1 Introdução

Nos dias atuais a criatividade é considerada uma das principais habilidades e ela é essencial para estimular o desenvolvimento da criatividade também na Educação Básica, uma vez que facilita e colabora para um aprendizado mais contextualizado e significativo [P21, 2019]. A criatividade está presente em diversos currículos escolares

ao redor do mundo, incluindo o Brasil. Ela também está sendo incluída em avaliações padronizadas como o exame PISA desde 2022 [OECD, 2019].

Por vezes a criatividade é associada ao ramo das artes, tais como música, pintura, cinema, fotografia e literatura. Porém, criatividade é a qualidade de quem tem capacidade, talento e inteligência para criar, inventar ou fazer inovações na área em que atua, podendo essa área ser qualquer uma [CRIATIVIDADE, 2020]. Portanto ela também está presente e pode ser desenvolvida como parte da computação como diz a Base Nacional Comum Curricular (BNCC):

> "Exercitar a curiosidade intelectual e recorrer à abordagem própria das ciências, incluindo a investigação, a reflexão, a análise crítica, a imaginação e a criatividade, para investigar causas, elaborar e testar hipóteses, formular e resolver problemas e criar soluções (inclusive tecnológicas) com base nos conhecimentos das diferentes áreas."[MEC, 2018].

Uma das formas para o desenvolvimento da criatividade por meio da computação é o desenvolvimento, por parte dos alunos, de aplicativos, utilizando plataformas como o App Inventor, como parte do ensino. Porém, para guiar o desenvolvimento da criatividade se faz necessário a utilização de alguma forma de avaliação do artefato criado e de *feedback* dos resultados dessa avaliação [MISHRA; HENRIKSEN, 2013]. Contudo, avaliar um aspecto complexo como a criatividade não é uma tarefa simples.

A definição de criatividade de um artefato normalmente é definida em termos de novidade, adequação e condensação [ALVES et al., 2020]. Dentro do aspecto de novidade a originalidade é uma das principais características, sendo, um artefato, considerado original quando apresenta algo inédito ou surpreendente.

Embora a criatividade seja considerada uma das principais competências do século XXI, e a originalidade uma das suas principais características, ainda existem poucas pesquisas sobre como avaliar o aspecto de originalidade de um artefato dentro de um contexto educacional, em específico, dentro do ensino de computação. As abordagens existentes [SCAICO et al., 2013][GAL et al., 2017][MUSTAFARAJ et al., 2017][TURBAK et al., 2017] focam na avaliação da originalidade do código fonte. Contudo, a originalidade de um app pode ser analisada de diversos pontos de vista , como, por exemplo, tópicos abordados, design de interface, funcionalidades, dentre outros [ALVES et al., 2020].

Assim, um dos aspectos se refere a avaliar o grau de originalidade de um artefato por meio da classificação dos tópicos abordados pelo aplicativo. Nesse contexto um aplicativo é considerado original se aborda tópicos novos (ou menos comumente abordados) comparado a aplicativos já existentes.

Essa classificação de tópico(s) do app pode ser automatizada utilizando processamento de linguagem natural (*natural language processing - NLP*). Nesse contexto, os textos utilizados para a classificação dos tópicos abordados por um

aplicativo são extraídos de todos os elementos textuais presentes no aplicativo, seja em componentes de interface como em outros tipos de componentes.

Uma solução para a classificação dos tópicos, abordados por um aplicativo, por meio dos textos presentes neste artefato é o uso de técnicas de classificação de tópicos. Existem diversas técnicas de modelagem e de classificação de tópicos, como, por exemplo, *Latent Dirichlet Allocation (LDA), Decision Trees, Random Forest, Logistic Regression, Naive Bayes e K Nearest Neighbour (KNN)*, entre outras. Cada uma destas técnicas apresenta diferentes abordagens para a descoberta/classificação de tópicos, como o uso de técnicas Bayesianas, no caso do LDA [SCARPA, 2017], e técnicas de *Machine Learning* e análise estatística para a classificação de tópicos, como no caso de *Decision Trees* e *Random Forest*. Também existem métodos como *Bidirectional Encoder Representations from Transformers (BERT)* [DEVLIN, 2018] que utilizam técnicas de *transformers* para a descoberta de relações contextuais entre as palavras de um texto.

Assim, o objetivo deste trabalho é, sistematicamente, analisar e comparar o desempenho de diferentes técnicas aplicadas aos elementos textuais de apps criados com App Inventor. E com base nos resultados definir qual das técnicas apresenta um melhor desempenho na classificação de tópicos dos aplicativos com base nos textos apresentados nas telas e em elementos de texto de componentes presentes nos aplicativos analisados para possibilitar a automação da avaliação da originalidade de aplicativos em termos de tópicos

2 Estado da Arte

O objetivo deste mapeamento é responder a seguinte pergunta de pesquisa: Quais abordagens existem para analisar e classificar, de forma automática, o tópico de um aplicativo móvel (criado com App Inventor) a partir de textos extraídos desse aplicativo? Esta pergunta é refinada nas seguintes questões de análise:

> **AQ1.** Quais abordagens existem para a classificação de tópicos de aplicativos móveis a partir dos textos extraídos do app e quais as suas características?
> **AQ2.** Qual o desempenho dessas técnicas?

Critérios de inclusão e exclusão de artigos. Conforme o foco da pesquisa são definidos os seguintes critérios de inclusão e exclusão de artigos científicos.

- São incluídos somente artigos que apresentam abordagens de classificação de tópicos com base unicamente no texto do aplicativo.
- São excluídos artigos que apresentem abordagens de classificação de tópicos que sejam baseados na descrição, revisões, resenhas, etc., uma vez que no contexto do App Inventor essas informações não existem ou são escassas.
- São excluídos artigos com abordagens voltadas para a detecção de plágio, análise de sentimentos, aplicações de *chatbots*, etc.
- São considerados apenas artigos publicados nos últimos dez anos (desde 2012), levando em consideração o avanço recente especificamente em relação a aplicativos móveis.

- São incluídos artigos que apresentam uma avaliação de ou comparativos de desempenho das abordagens propostas.

Fonte. Para o levantamento dos artigos utilizados nesta pesquisa foram utilizados os principais bancos de dados e bibliotecas virtuais dentro da área da computação, tais como as bibliotecas virtuais ACM, IEEE Xplore e Scopus, com acesso por meio do Portal Capes, além do motor de busca de artigos científicos Google Scholar.

Critérios de qualidade. São considerados apenas artigos que apresentam informações substanciais e relevantes para a resolução das questões de análises. Portanto artigos cujo conteúdo se resumem a uma única página são excluídos.

Termos de busca. Com base na questão de pesquisa, várias pesquisas adicionais foram utilizadas com o intuito de calibrar a *string* de busca, identificando os termos mais relevantes para a busca bem como seus sinônimos (Tabela 6) para assim diminuir ao máximo o risco de não incluir algum artigo relevante.
Após a definição dos termos relevantes e seus sinônimos foi definida a *string* de busca padrão a ser aplicada nas bases de dados e bibliotecas virtuais:

```
("topic classification" OR "topic categorization" OR "text classification" OR "text categorization" OR "keyword extraction") AND ("mobile application" OR "app" OR "ios" OR "android" OR "app inventor")
```

A busca foi realizada em outubro de 2022. A Tabela 1 apresenta os resultados das buscas nas fontes.

Tabela 1. Número de artigos identificados por repositório e por fase de seleção.

Fonte	**Número de resultados de busca**	**Número de resultados analisados**	**Número de documentos potencialmente relevantes**	**Número de documentos relevantes**
ACM	0	0	0	0
IEEE Xplore	44	44	5	0
SCOPUS	140	140	8	0
Google Scholar	81900	200	13	0
Total				**0**

Em relação a classificação de aplicativos móveis foram encontradas apenas pesquisas referentes a classificação com base nas descrições dos apps e em alguns casos com base nas API's. Analisando essas pesquisas, mesmo que não usem como entrada termos e textos extraídos do próprio aplicativo, observa-se a utilização de diversas técnicas de classificação. Entre elas, por exemplo, *Naive Bayes* e suas variantes [DONG et al., 2016][QIU et al., 2022], *Support Vector Machine* (SVM), *Random Forest*, *K Nearest Neighbors* (KNN) [QIU et al., 2022]; *TextCNN, LSTM, RCNN, CRNN* [ZHANG et al., 2021][YANG et al., 2019] e *BERT, BERT+Highway+GRU* [YANG et al., 2019].

Quanto às medidas de desempenho utilizadas, Dong et al. (2016) utiliza a acurácia como um comparativo de desempenho, assim como Qiu et al. (2022). Já Zhang et al. (2021) faz uso da pontuação F1 como métrica de comparação de desempenho. Yang et al. (2019) utiliza além da pontuação F1 e a acurácia métricas como precisão e sensibilidade, para comparar os métodos estudados.

Como resultado da revisão na literatura observa-se que não foi encontrado nenhum trabalho de pesquisa que tenha como foco a classificação de aplicativos por meio de textos e termos extraídos do próprio aplicativo.

3 Estudo Experimental

3.1 Definição do Estudo

O objetivo do estudo neste trabalho é comparar o desempenho de diferentes técnicas de classificação de textos com base em termos e textos extraídos diretamente de aplicativos móveis criados na plataforma App Inventor. Os textos utilizados para a classificação dos apps são extraídos diretamente de aplicativos em português desenvolvidos no App Inventor a partir do arquivo .aia.

Para a etapa de pré processamento primeiramente é feita a limpeza dos textos brutos extraídos, bem como eliminação de sinais de pontuação, números e caracteres especiais, além da conversão de letras maiúsculas em minúsculas. A partir desse ponto são eliminados os termos de pouca importância para classificação, como artigos, preposições, pronomes e advérbios, chamados também de *stopwords*.

Com o texto limpo são testadas várias alternativas de filtragens a mais, tais como, eliminação de termos que não sejam substantivos, ou utilização do texto bruto para a classificação, assim como *stemming* e *lemmatization* do conteúdo textual. Também será utilizada a técnica de redução de dimensionalidade LSA.

Após a etapa de pré processamento e de acordo com a revisão da literatura, foram escolhidas várias técnicas de classificação de texto que serão aplicadas:

- técnicas de *machine learning*: *Decision Tree, Random Forest, Logistic Regression, Multinomial Naives Bayes, K Nearest Neighbour, Support Vector Machine* (SVM).
- técnicas de *deep learning*: BERT.
- técnica de modelagem de tópicos: LDA

O desempenho destas técnicas é medido e comparado por meio de acurácia, precisão, sensibilidade e pontuação F1.

3.2 Definição de tópicos de apps criados com App Inventor

Para a classificação dos aplicativos se faz necessário a definição das categorias. A Tabela 2 apresenta as categorias que serão utilizadas para a classificação dos aplicativos App Inventor. Elas foram definidas com base no mapeamento das categorias usadas nas principais lojas online de aplicativos, Google Play e Apple App Store.

Tabela 2. Categorias criadas para App Inventor e suas definições

CATEGORIA	DEFINIÇÃO

Animais e pets	Aplicativos de animais domesticados ou não, identificação de animais, cuidados com animais.
Automóveis, veículos e transporte	Aplicativos relacionados a automóveis, bicicletas ou outros tipos de veículos e transporte.
Beleza e moda	Aplicativos que fornecem informações, história, agendamento ou tutoriais de beleza.
Cidadania e questões sociais	Aplicativo para tratar de questões cívicas, denúncias, transparência de recursos públicos e similares.
Comes-e-bebes	Aplicativos que fornecem recomendações, instruções, receitas ou críticas relacionadas à preparação, consumo ou revisão de alimentos ou líquidos/sucos.
Comunicação	Aplicativos que fornecem suporte para comunicação como acesso a chats ou aplicativos de chats, contatos, telefones.
Design, pintura e fotografia	Aplicativos para esboços, ferramentas de pintura e design, livros para colorir ou auxiliam na captura, edição, gerenciamento, armazenamento ou compartilhamento de fotos.
Educação	Aplicativos que abordam o ensino de uma habilidade, assunto específico ou são voltados à questões de estudo, como notas de provas, sala de aula, instituições de ensino.
Engenharia, física e construção	Aplicativo relacionado a construções, cálculo de materiais para obra, planejamento de obras, cálculo de medidas, motores, máquinas.
Entretenimento	Aplicativos que informam o usuário sobre algum evento ou apresentam conteúdo de entretenimento visual ou outro conteúdo de entretenimento.
Espiritualidade, crença e adivinhação	Aplicativos de adivinhação, horóscopo, questões religiosas, filosóficas e existenciais.
Ferramentas do celular	Aplicativos de lanterna, leitor de códigos e similares.
Finanças e trabalho	Aplicativos que realizam transações financeiras ou auxiliam o usuário em questões financeiras comerciais, profissionais ou pessoais, ou divulgam e encontram empregos, trabalhos, vagas, portfólios e similares.
Medicina e saúde	Aplicativos focados em educação médica, gerenciamento de informações ou informações de saúde para pacientes ou profissionais de saúde, cuidados do bebê e infantis.
Meio ambiente e botânica	Aplicativo de reciclagem, descarte ou cuidado e informações sobre o meio ambiente ou plantas, identificação plantas, cuidados com plantas.
Matemática	Aplicativos para fornecer informações ou realizar cálculos algébricos, trigonométricos e geométricos.
Música	Aplicativo para ouvir rádio, músicas e similares.
Produtividade	Aplicativos que tornam um processo ou tarefa específica mais organizado ou eficiente.
Robótica, computação física e automação	Aplicativos para controlar robô, comunicar-se com placas física (ex.: Arduino) e automação residencial/predial.
Vida saudável e esporte	Aplicativos relacionados a uma vida saudável ou prática de um esporte, treino, dieta, nutrição, gerenciamento de estresse, condicionamento físico, indicadores de peso (ex.: IMC).
Turismo e geografia, tempo e meteorologia	Aplicativos que fornecem informações ou mapas sobre um local, pontos de interesse ou fornecem previsões, alertas e informações relacionadas às condições meteorológicas de um local.

3.3 Preparação do conjunto de dados

Para o conjunto de dados, usado neste trabalho, são utilizados dados de um total de 1.672 aplicativos, em português brasileiro, criados e publicados na *gallery* do App Inventor, além de apps desenvolvidos no âmbito da iniciativa Computação na Escola/INCoD/INE/UFSC. O uso do App Inventor bem como os apps produzidos na plataforma são licenciados sob a licença *Creative Commons* o que o torna livres para a utilização, modificação e compartilhamento.

3.4 Treinamentos

Para a modelagem de tópicos com o método *Latent Dirichlet Allocation* (LDA) foi utilizado os dados textuais dos aplicativos presentes no conjunto de dados. Foram feitos

três experimentos de modelagem: o primeiro utilizando apenas os substantivos dos texto, o segundo e terceiro experimentos utilizaram os textos bruto com diferença apenas no número de *features*, ou palavras, utilizadas, sendo 10.000 *features* no segundo experimento e 17.260 *features* no terceiro. Em todos os experimentos os dados textuais tiveram seus caracteres especiais e números removidos, além da conversão para letras minúsculas, e o parâmetro de número de tópicos a serem gerados foi configurado para 21, conforme a quantidade dos tópicos definidos no contexto de apps com App Inventor. Para analisar e comparar técnicas de *machine learning* no contexto de classificação de aplicativos, são realizados treinamentos de métodos clássicos de *machine learning*, incluindo:

- *Decision Tree*
- *Random Forest*
- Regressão Logística
- *Multinomial Naive Bayes*
- *Gaussian Naive Bayes*
- *K Nearest Neighbour*
- *Support Vector Machine*

Para o treinamento dos métodos citados o conjunto de dados foi dividido em duas partes: conjunto de treinamento, com 70% do conjunto de dados, e conjunto de validação, composto pelos 30% restantes.

Além dos métodos de classificação clássicos é também realizado experimentos com o método de *deep learning* BERT. Para o treinamento do BERT o conjunto de treinamento é composto por 90% do conjunto de dados enquanto o conjunto de validação é composto pelos 10% restantes.

Os experimentos foram realizados dentro da plataforma Colaboratory do Google, um serviço de nuvem que oferece recursos computacionais para a execução de códigos em Python pelo navegador, muito utilizado para pesquisas em aprendizado de máquina, análise de dados, dentre outros. Para a importação e manipulação dos dados foi utilizado a biblioteca *open-source* Pandas.

Foram realizados diversos experimentos, alguns utilizando o texto pré-processado e limpo, e outros o texto bruto, apenas com remoção de números e caracteres especiais e conversão para letra minúscula. Em alguns testes foi utilizado métodos de *stemming* e *lemmatization* dos textos, e foi utilizado também o método LSA para a redução de dimensão.

3.5 Discussão

Os experimentos de modelagem de tópicos realizados com LDA demonstraram uma grande dificuldade do modelo em gerar tópicos cujo tema seja facilmente distinguível um dos outros. Diversos tópicos poderiam ser associados a um mesmo tema, o que não é interessante para o contexto deste trabalho, portanto tanto a modelagem de tópicos quanto o modelo de LDA foram descartados como opção para a resolução da tarefa de classificação de aplicativos desenvolvidos no App Inventor.

Em relação aos experimentos de classificação utilizando modelos de *machine learning* e *deep learning* os 5 melhores resultados obtidos nos experimentos realizados são resumidos na Tabela 3. Para esta comparação foi utilizada a métrica de acurácia como principal critério de desempenho, uma vez que ela mede a porcentagem de acertos na classificação, o que no contexto de classificação de aplicativos é muito importante.

Tabela 3. Top 5 melhores resultados

Modelo	**Métricas**				**Experimento**
	Acurácia	Precisão	Sensibilidade	Pontuação F1	
Multinomial Naive Bayes	**0.7109**	**0.71**	**0.71**	**0.70**	**Só substantivos**
Multinomial Naive Bayes	0.7088	0.71	0.71	0.70	Só substantivos + stemming
Random Forest	0.7052	0.73	0.71	0.68	Texto Bruto 10.000 features
Multinomial Naive Bayes	0.7012	0.73	0.70	0.68	Texto Bruto 10.000 features + stemming
Multinomial Naive Bayes	0.6932	0.73	0.69	0.67	Texto Bruto 17.260 features

Analisando, de forma geral, os resultados, o modelo que mais se destaca é o *multinomial naive bayes*, apresentando o melhor desempenho na maioria dos experimentos realizados. Comparando os experimentos é possível notar um maior desempenho dos modelos ao utilizar o texto apenas com substantivos, com exceção dos experimentos utilizando *lemmatization*.

Dentre os resultados obtidos, destaca-se, o *multinomial naive bayes* no experimento utilizando o texto apenas com substantivos, obtendo a maior acurácia (0.7109), a maior sensibilidade (0.71) e maior pontuação F1 (0.70) dentre todos os modelos em qualquer um dos experimentos. Com base nos resultados alcançados, o modelo *multinomial naive bayes* destaca-se como melhor opção para a classificação de textos curtos, como os encontrados em aplicativos móveis. Quanto ao pré-processamento, observa-se que a utilização de textos só com substantivos tende a levar o modelo utilizado a um melhor desempenho na tarefa de classificação.

4 Conclusão

Como resultado do presente trabalho foram analisados conceitos de originalidade pertinentes em um contexto de avaliação de originalidade em aplicativos criados no App Inventor. Foram também sintetizados como conceitos de *machine learning* e *deep learning*, bem como conceitos de pré-processamento de dados, necessários para a realização dos experimentos propostos. Foi realizado um levantamento do estado da arte em volta da utilização de modelos de *machine learning* e *deep learning* na classificação de textos mostrando que atualmente não existem abordagens que utilizam os textos

presentes nos aplicativos para a classificação desses apps. Com base nesse levantamento foram selecionados diversos modelos de classificação que foram treinados, avaliados e comparados utilizando um conjunto de textos extraídos de apps App Inventor. Os resultados obtidos indicam o modelo de *multinomial naive bayes* como a melhor opção para a classificação de aplicativos por meio dos textos extraídos destes aplicativos.

Assim, o principal impacto científico do presente trabalho é o conhecimento gerado indicando a(s) melhor(es) opções para a classificação de aplicativos com base nos textos presentes nele. Isto fornece uma base para o desenvolvimento de ferramentas de automatização de classificação de aplicativos móveis, como por exemplo, os criados em App Inventor. Assim, contribuindo com a avaliação automatizada da aprendizagem de criatividade, o que também representa uma contribuição social no contexto do ensino de computação na Educação Básica.

Contudo ainda existem possibilidades a serem exploradas dentro do contexto de classificação de apps por meio de seus textos. Um possível trabalho futuro seria a realização dos experimentos com base em um conjunto de dados maior e assim averiguar melhor o desempenho dos modelos abordados, principalmente BERT, que demanda uma grande quantidade de dados. Outra opção é a exploração de diferentes técnicas de pré-processamento do texto extraído, o que pode influenciar no desempenho dos modelos. Assim como a utilização de outros modelos de classificação de textos.

Referências

ALVES, Nathalia da Cruz et al. Uma Proposta de Avaliação da Originalidade do Produto no Ensino de Algoritmos e Programação na Educação Básica. In: Anais do Simpósio Brasileiro de Informática na Educação, Natal, Brasil, 2020.

CRIATIVIDADE. *In:* DICIO, Dicionário Online de Português. 7Graus. 2020. Disponível em:<https://www.dicio.com.br/criatividade/>.

DEVLIN, Jacob; CHANG, Ming-Wei. Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing. Google AI Blog, 2018. Disponível em: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html.

DEVLIN, Jacob et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Cornell University, arXiv preprint arXiv:1810.04805, 2018.

DONG, Feng et al. ClassifyDroid: Large Scale Android Applications Classification Using Semi-Supervised Multinomial Naive Bayes. In: 2016 4th International Conference on Cloud Computing and Intelligence Systems. IEEE, p. 77-81, 2016.

GAL, Lilach et al. Suggesting a Log-Based Creativity Measurement for Online Programming Learning Environment. In Proc. of the 4th Conference on Learning at Scale, 273-277. ACM, 2017.

MEC. Base Nacional Comum Curricular (BNCC). Disponível em: http://basenacionalcomum.mec.gov.br/.

MUSTAFARAJ, Eni; TURBAK, Franklyn; SVANBERG, Maja. Identifying Original Projects in App Inventor. In Proc. Florida Artificial Intelligence Research Society Conference, 2017.

OECD. PISA 2021 Creative Thinking Framework, 2019. Disponível em: https://www.oecd.org/pisa/publications/PISA2021-Creative-Thinking-Framework.pdf.

P21. 21st Century Skills, 2020. Disponível em: www.p21.org.

QIU, Haoyu et al. Exploring Multiple Genres Text Classification: Classifying 61 Genres of Mobile App Description Based on Naïve Bayes and Count Vectorizer. In: 2022 3rd International Conference on Electronic Communication and Artificial Intelligence. IEEE, p. 156-162, 2022.

SCAICO, Pasqueline Dantas; et al. Ensino de Programação no Ensino Médio: Uma Abordagem Orientada ao Design com a Linguagem Scratch. Revista Brasileira de Informática na Educação, 21(2), 2013.

SCARPA, Alice Duarte. Técnicas de Processamento de Linguagem Natural Aplicadas às Ciências Sociais. 68 f. Dissertação (Mestrado) - Curso de Matemática, Fundação Getúlio Vargas- Fgv, Rio de Janeiro - RJ, 2017.

TURBAK, Franklyn et al. Work in progress: Identifying and Analyzing Original Projects in an Open-Ended Blocks Programming Environment. In International Conference on Distributed Multimedia Systems, Visual Languages and Sentient Systems, Wyndham Pittsburgh/EUA, 2017.

YANG, Wenchuan et al. Design and Implementation of Application Classification Based on Deep Learning. In: 2019 Chinese Automation Congress. IEEE, p. 4821-4826, 2019.

ZHANG, Hua et al. Research on Android Multi-classification Based on Text. In: Journal of Physics: Conference Series. IOP Publishing, p. 012049, 2021.