DAS Departamento de Automação e Sistemas
CTC Centro Tecnológico
UFSC Universidade Federal de Santa Catarina

Geração Automática de *Features* para Modelagem Preditiva - Predição de Empresas Brasileiras de Alto Crescimento

Relatório submetido à Universidade Federal de Santa Catarina
como requisito para a aprovação da disciplina:
DAS 5511: Projeto de Fim de Curso

Luis Felipe Pelison

Florianópolis, Dezembro de 2018

Geração Automática de *Features* para Modelagem Preditiva - Predição de Empresas Brasileiras de Alto Crescimento

Luis Felipe Pelison

Esta monografia foi julgada no contexto da disciplina

DAS 5511: Projeto de Fim de Curso

e aprovada na sua forma final pelo

Curso de Engenharia de Controle e Automação

Prof. Jomi Fred Hübner

Banca Examinadora:

André Carvalho Bittencourt
Orientador na Empresa

Prof. Jomi Fred Hübner
Orientador no Curso

Prof. Julio Elias Normey-Rico
Responsável pela disciplina

Prof. Ricardo José Rabelo, Avaliador

Guilherme Hammes, Debatedor

Fernando Teures Redondo, Debatedor

Agradecimentos

Merecem agradecimentos especiais as pessoas e entidades que estiveram, de alguma forma, envolvidas com a execução e o acontecimento desse sonho: realizar uma graduação em Engenharia de Controle e Automação. Dentre as entidades, agradeço à Neoway, pela estrutura, conhecimento e por me receber sempre de braços abertos; à Universidade Federal de Santa Catarina (UFSC), que forneceu professores e estrutura para o aprendizado de diversos conceitos utilizados nesses 5 anos; ao LABMETRO, onde pude realizar um período de iniciação científica e ao Departamento de Automação e Sistemas, por manter e organizar um curso de alto padrão dentro do nosso país.

Com relação às pessoas, agradeço ao Prof. Jomi F. Hübner pelas orientações em estágio e PFC; ao Prof. Rodolfo C. C. Flesch, o primeiro a me dar oportunidades de pesquisa dentro da universidade; aos colegas de empresa André C. Bittencourt, Gabriel B. Alvim, Matheus dos Santos Frata, Ígor S. Cortez, Beatriz Gouveia, Claudinei C., André Boechat, Roger Estefani, Bento Borges, Paulo Jansen e Raul Deus pela coorientação e dicas no dia-a-dia que contribuíram, e muito, para o projeto final aqui citado, na minha vida pessoal e profissional; e aos colegas de graduação que formaram uma segunda família para mim em Florianópolis.

Em especial, agradeço à minha família, pelo suporte nas minhas escolhas de vida, e à minha namorada e companheira do dia a dia Bruna C. Appelt, pelas palavras de apoio e momentos de reflexão que trazem conforto e abrem horizontes.

"Querer vencer significa
já ter percorrido
metade do caminho.
(Ignacy Paderewski)

Resumo

A *Neoway Business Solutions*, empresa brasileira focada em inteligência de negócios a partir de um grande volume de dados, coletados em mais de 3.000 fontes públicas, atua em um mercado *Business to Business (B2B)* para prospectar e entender o mercado ou prevenir perdas dos clientes. Nesse contexto, a área interna de *Analytics* é responsável por gerar inteligência em cima dos mais variados dados. O projeto desenvolvido, dentro da área de *Analytics*, visa gerar inteligência a partir de dados relacionados à natureza e localidade das empresas, a fim de predizer se o crescimento médio dos próximos três anos será superior a 17%, para qualquer empresa ativa do Brasil que possua 10 ou mais funcionários. Aliado à inteligência, o projeto tem como objetivo validar uma abordagem muito recente no mundo de aprendizado de máquina e ciência de dados: a geração automática de *features*. Com a geração automática de *features*, os modelos inteligentes podem ser rapidamente implementados, com uma redução de até 5x no tempo de desenvolvimento, comparado ao *pipeline* de ciência de dados padrão. Técnicas de geração automática de *features* foram estudadas e uma foi escolhida para ser utilizada. Base de dados relacionais foram utilizadas para a geração automática de *features* a partir da teoria chamada *Deep Feature Synthesis*. Com essas *features*, alguns modelos foram criados e comparados entre si. Por fim, o melhor modelo preditivo (com *features* automáticas) foi comparado com outro, gerado por um cientista de dados especialista no domínio, e atingiu resultados muito semelhantes.

Palavras-chave: Geração de *Features* Automáticas. Engenharia de *Features*. Modelo Preditivo. Aprendizado de Máquina Automático. Inteligência Artificial. Empresas de Alto Crescimento.

Abstract

The Neoway Business Solutions is a brazilian company focused in business intelligence for big data. With more than 3.000 public data sources, the company operates in a Business-to-Business (B2B) marketplace to prospect and understand the customer's market or prevent customer losses. At this scenario, the Neoway's Analytics team is responsible for generating intelligence on the most varied data. The project developed aims to generate knowledge from data related to the firmographics, in order to predict wich brazilian company with 10 or more employees will be a High-Growth Firm. Also, the projects aims to validate a very recent approach in the Data Science world: the Automated Feature Generation. With this approach, smart models can be raplidly deployed, with up to a 5x reduction in development time compared to standard Data Science pipeline. Some techniques about Automated Feature Generation have been studied and one chosen to be used. Relational databases were used to generate the automated features from the Deep Feature Synthesis theory. With these new features, some models were created and compared to each other. Lastly, the best predictive model generated (with automated features) was compared to another, builded by a senior data scientist, and have gotten results very similar.

Keywords: Automated Feature Generation. Automated Machine Learning. Feature Engineering. Deep Feature Synthesis. Artificial Intelligence. High-Growth Firms.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API - *Application Programming Interface*

AUC - *Area Under the ROC Curve*

AutoML - *Auto Machile Learning*

B2B - *Business to Business*

CNAE - Classificação Nacional de Atividades Econômicas

CRISP-DM - *Cross Industry Standard Process for Data Mining*

DFS - *Deep Feature Synthesis*

DRF - *Distributed Random Forest*

EAC - Empresa de Alto Crescimento

EPP - Empresa de Pequeno Porte

GBM - *Gradient Boosting Machine*

GLM - *Gneralized Linear Model*

ME - Microempresa

OBM - *One Button Machine*

PCA - *Principal Component Analysis*

SaaS - *Software as a Service*

UFSC - Universidade Federal de Santa Catarina

Sumário

1 Introdução

Data Science é uma área que vem crescendo pelo fato de haver uma presença massiva de dados disponíveis no dia a dia. Enquanto as pessoas compram, se comunicam, leem notícias, navegam na internet, procuram informações, ouvem música e expressam suas opiniões, existe um banco de dados armazenando tudo isso [1]. Aliado a grande massa de dados, o poder computacional cresceu muito e ficou barato. Assim, a análise de dados e a criação de modelos preditivos estão em alta, principalmente na indústria [2].

Muitos sistemas hoje utilizam o poder dos dados para criar inteligência artificial embutida. No mundo virtual, os algoritmos inteligentes estão em grandes plataformas mundiais, como *Google* [3], *Amazon* [4], *Facebook* [5], *Netflix* [6], *Uber* [7] entre outros. Recomendações de passagens áreas, comida, filmes, hospedagens, algoritmos que identificam objetos, pessoas, carros, voz, texto e sentimentos, são exemplos de inteligência artificial aplicada no cotidiano. O responsável por desenvolver essas inteligências artificiais é chamado de Cientista de Dados.

O Cientista de dados é responsável por limpar, padronizar e pré-processar os dados. Para, então, estudar e entender cada variável e registro que terá disponível para criar um modelo ou extrair conhecimentos, que serão de grande utilidade para o assunto envolvido. Esse *pipeline* todo costuma levar tempo. Por exemplo, dentro da Neoway, um modelo de *Data Science* leva em média 2 meses para ficar pronto. Desse modo, ainda é um trabalho demorado de análises e processamentos manuais.

1.1 Motivação

De acordo com [8], as etapas que mais consomem tempo e conhecimento humano, dentro do *pipeline* padrão de um cientista de dados, são as etapas de *Feature Extraction* e *Feature Selection*, que em português podem ser definidas como Extração de Variáveis e Seleção de Variáveis, respectivamente. Em linhas gerais, essas tarefas são o levantamento das melhores variáveis possíveis da base de dados, juntamente com o entendimento específico do problema, utilizando também de agregações e outras operações em cima dos dados, a fim de gerar informações que possam conduzir o algoritmo a tomar melhores decisões depois do aprendizado. Geralmente, essas etapas consomem entre 70% a 80% do tempo de um cientista de dados [9].

Então, entendendo as tarefas que mais demandam tempo, a motivação do projeto é utilizar técnicas que automatizam esse trabalho para os cientistas de dados. Assim, com variáveis produzidas automaticamente, o trabalho de um cientista de dados ficará mais

ágil.

Espera-se que, além do tempo reduzido para a geração de um modelo de aprendizado, novos modelos possam ser gerados a partir de variáveis criadas automaticamente e as variáveis consigam ser interpretadas e possam gerar novos *insights* para o time de *Analytics* da Neoway.

1.2 Metodologia

Para abordar esse problema, diversas técnicas recentes para a geração automática de variáveis foram estudadas. Uma técnica foi escolhida para aplicar em um problema real de predição de empresas brasileiras de alto crescimento.

Existem técnicas que necessitam de dados relacionais e outras não, bastam seus tipos bem definidos. A partir disso, elas são capazes de gerar um número muito grande de variáveis automaticamente. Assim que escolhida a técnica, ela foi então aplicada e gerou as variáveis.

Após essa etapa, é necessário passar as variáveis por uma seleção automática que retira aquelas que não consigam representar ou informar algo útil ao modelo de aprendizado.

Ao final, foram configurados e executados diversos algoritmos de aprendizado de máquina para criar o melhor modelo preditivo.

1.3 Escopo

O escopo do projeto engloba apenas as etapas necessárias para validar as técnicas de geração automática de variáveis. Essas ocorrem em cima de um problema específico: predição de empresas de alto crescimento. A escolha do melhor algoritmo de aprendizado de máquina não foi focada nesse projeto, porém, alguns algoritmos foram utilizados para fins de comparação de resultados.

1.4 Objetivos

1.4.1 Gerais

O projeto possui dois principais objetivos:

- Validar a abordagem de geração automática de variáveis;
- Criar um modelo preditivo para as empresas brasileiras de alto crescimento.

1.4.2 Específicos

O projeto tem objetivos relacionados a cada objetivo macro e estes estão descritos no formato de *Research Questions*.

- Validar a abordagem de geração automática de variáveis, respondendo as questões:
  - A abordagem está madura o suficiente para ser implementada em problemas reais?
  - Qual o poder preditivo e a qualidade dessas variáveis?
  - Como se comporta a performance da geração automática conforme aumentam o número de registros e de combinações entre dados diferentes?
  - O que a ferramenta necessita como *Input*? Essa técnica é capaz de aprender os relacionamentos entre os dados?
  - Como cada ferramenta de geração automática de variáveis se encontra em relação a:
    - Estrutura;
    - Capacidades técnicas;
    - Restrições;
    - Desempenho;
    - Contribuidores;
- Criar um modelo preditivo para o crescimento de empresas brasileiras;
  - Como o modelo responde às métricas definidas em comparação a um modelo gerado sem técnicas automáticas de geração de variáveis quanto à:
    - Capacidade de predição das variáveis;
    - Memória e processamento computacional exigidos;
    - Tempo de desenvolvimento gasto.

1.5 Organização do Documento

O documento está organizado em 7 capítulos, onde será exposto, fundamentado e avaliado o trabalhado desenvolvido.

O capítulo 2 abordará a empresa onde o projeto foi realizado e como a solução proposta é importante para ela.

O capítulo 3 traz a fundamentação teórica por trás da solução proposta.

No capítulo 4 está a metodologia de desenvolvimento da solução proposta.

O capítulo 5 demonstra a implementação da solução proposta, com todos os passos requeridos para a resolução do problema.

No capítulo 6 estão os resultados obtidos, após a aplicação da solução e comparação com a solução manual.

Finalizando, o capítulo 7 traz conclusões, propostas de melhorias e trabalhos futuros.

2 Apresentação da Empresa

Figura 1 – Logo da empresa *Neoway*

A empresa onde o projeto foi realizado chama-se *Neoway* e está localizada em Florianópolis, Santa Catarina - Brasil. Sua logo está apresentada na Figura 1. Atualmente, a empresa tem mais de 500 clientes em diversos países e, até 2017, captou mais de 140 milhões de reais em investimentos. Além disso, tem sedes em outras cidades, como São Paulo e Nova York e conta com mais de 300 colaboradores.

A Neoway traz soluções de inteligência de mercado para outras empresas, trabalhando de uma forma B2B - *Business to Business*, utilizando-se de *Big Data*, com o processamento de mais de 50bi de dados e informações.

A empresa se divide em duas especialidades: Prospecção de Novos Clientes e Prevenção de Perdas. Para atigir essas duas especialidades, a empresa fornece soluções em formato de serviços (específicos para cada cliente, como pequenos estudos, consultorias e fornecimento de inteligência de negócios) e produtos, como uma plataforma online, no formato *SaaS - Software as a Service*, que provê diversos aplicativos, na nuvem, para os clientes adquirirem informações relevantes sobre empresas, pessoas e ligações entre ambos.

Dentre esses aplicativos, existe um para pesquisas de empresas e pessoas, que segmenta as buscas por filtros criados pela *Neoway*, a fim de proporcionar conhecimento aos potenciais clientes ou de qualquer outra empresa que o cliente da *Neoway* deseja ter. Outro aplicativo, presente na plataforma, é o de mapa. Esse traz informações georreferenciadas que possibilitam decisões estratégicas sobre ida ao mercado, regiões propícias para investimento, localização de concorrentes, infraestrutura entre outros.

A plataforma online utiliza dados capturados de fontes públicas disponíveis na internet e entrega os dados de uma forma fácil de serem analisados.

2.1 Estrutura Interna

Internamente, a empresa de Florianópolis se divide em áreas maiores de Produto, Aplicação, Dados, *Analytics*, Financeiro, Suporte e *Customer Success*.

2.2 Área de Analytics

Analytics é a área onde o projeto foi desenvolvido. O time em Florianópolis, pertencente à essa área, é composto por 9 membros e tem como objetivo desenvolver modelos estatísticos e de inteligência artificial em cima da base de dados da *Neoway*, para gerar inteligência na plataforma online. Dessa forma, são desenvolvidos desde sistemas de recomendação de novos clientes até filtros inteligentes, para serem utilizados como segmentações nas buscas, como por exemplo o faturamento estimado de uma empresa.

2.3 Importância do Projeto

A *Neoway* possui uma equipe de cientista de dados que lidam com diversos problemas de áreas distintas. Já lidaram com contextos muito diferentes, como jurídico, bens de consumo, negócios e saúde. A criação de variáveis a partir das 3.000 fontes públicas de dados é uma tarefa extremamente difícil de ser realizada manualmente, tanto pelo volume de dados, quanto pelas diferenças de contexto dos dados.

Existe uma necessidade de criar variáveis comuns e simples que podem ser a base para vários modelos de *Data Science*. Nessa linha, há um projeto interno que agrega mais de 400 variáveis criadas manualmente por todo o time. Essas variáveis são agregações simples a partir de fórmulas matemáticas como a soma e porcentagem, ou simplesmente os dados crus.

Assim, com a inserção de uma ferramenta automatizada capaz de gerar novas variáveis de maneira rápida, essa base de variáveis comuns pode crescer muito, realizando combinações entre todas as tabelas disponíveis e as variáveis já criadas. Além disso, a geração automática de variáveis pode gerar variáveis resposta para modelos antes não pensados. Assim, tanto o número de variáveis, quanto o número de modelos vai crescer em um tempo muito menor do que estava sendo feito.

3 Fundamentação Teórica

A área de ciência de dados é muito abrangente na teoria e na prática. Nesse trabalho algumas áreas foram mais focadas que outras, como as que envolvem *Feature Engineering* e geração automática de variáveis. Desse modo, apenas as necessárias para o entendimento do projeto em questão serão abordadas de modo mais profundo.

3.1 Ciência de Dados

De acordo com [1], a ciência de dados ainda não é bem definida na academia, mas herda características da área de estatística, matemática e computação. É uma área analítica que faz uso de algoritmos de inteligência artificial em alguns casos.

Um cientista de dados é responsável por extrair conhecimento e interpretar dados, que requerem ferramentas e métodos da estatística e de aprendizado de máquina [1].

Para clarificar as etapas de análise de dados dentro de uma empresa, [10] elaborou uma tabela relacionando as tarefas de *Analytics* com os responsáveis por elas. A Tabela 1 mostra, em ordem decrescente de utilidade de inteligência artificial e *Data Science*, as 4 etapas do uso de análise de dados dentro de uma empresa.

Tabela 1 – Tarefas de *Analytics* em ordem decrescente de utilidade de inteligência artificial e *data science*

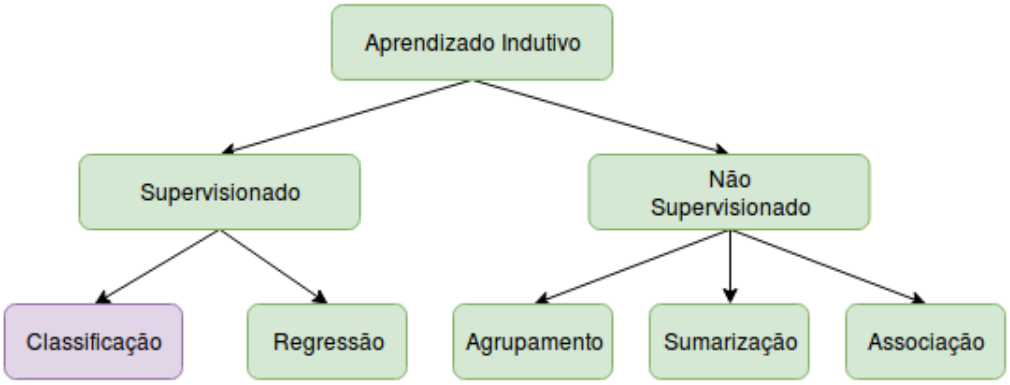
Nome da Etapa	Questão Descritiva	Técnicas Necessárias
Optimizing Analytics	"O que eu posso fazer?"	Suporte à Decisão, Otimização Multicritério
Predictive Analytics	"O que acontecerá?"	Modelagem
Diagnostic Analytics	"Por que isso ocorre?"	Business Intelligence, Modelagem
Descriptive Analytics	"O que aconteceu?	Business Intelligence

Fonte: [10]

Dentre esses 4 níveis apresentados por [10], *Data Science* se encaixa em *Predictive Analytics* e *Diagnostic Analytics*. E, para isso ser possível, algoritmos de inteligência

artificial são necessários. Esses algoritmos podem aprender com uma amostra de dados supervisionados, ou seja, com a resposta já definida a priori, ou com uma amostra de dados sem nenhuma identificação da resposta. Entrando mais a fundo nessas opções, se o caso for supervisionado (preditivo), podem haver problemas de classificação, onde a resposta é de forma discreta, ou problemas de regressão, onde a resposta é uma variável contínua. Já, se for um problema não supervisionado (descritivo), as tarefas podem ser de agrupamento por similaridades, sumarização ou associação. [11] Essa divisão pode ser melhor visualizada na Figura 2.

Figura 2 – Hierarquia do aprendizado. Método de classificação destacado por ser o problema abordado nesse projeto.



Fonte: Adaptado de [11].

Por fim, para abordar todas essas possibilidades de tarefas, existe um padrão para estabelecer o *pipeline* de ciência de dados. Esse padrão se chama *CRISP-DM* e será explicado na sequência.

3.1.1 *CRISP-DM*

A sigla em inglês significa *Cross Industry Standard Process for Data Mining* [12]. Em português, podemos chamar de Processo Padrão Inter-Indústrias para Mineração de Dados. Pelo nome fica fácil identificar que o *CRISP-DM* é um padrão a ser seguido para a tarefa de *Data Mining*.

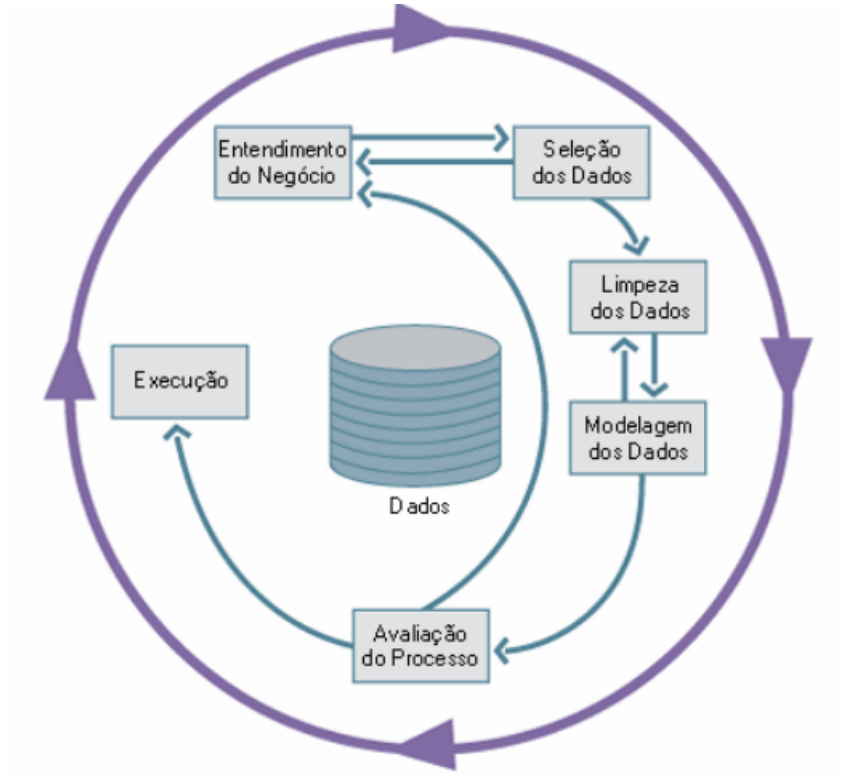
Esse padrão ou metodologia é definido por [12] como um processo que descreve as principais decisões que os experientes mineradores de dados utilizam em seus trabalhos. De acordo com [13], o processo de mineração de dados possui 6 fases:

- (a) Entendimento do negócio/problema;
- (b) Entendimento dos dados;

- (c) Preparação dos dados;
- (d) Aplicação do modelo (Aprendizado de Máquina);
- (e) Análise e avaliação dos resultados;
- (f) Produção.

Seu fluxo pode ser visualizado na Figura 3. Nesse trabalho, as etapas mais em evidências serão a (c) de preparação dos dados e (e) análise e avaliação dos resultados. A produção do modulo em larga escala não será abordada.

Figura 3 – Padrão de Atividades no Processo de Mineração de Dados



Fonte: 14

3.1.2 *Feature Engineering*

Em ciência de dados já é conhecido que o fator mais importante para o sucesso de um projeto com aprendizado de máquina são as variáveis utilizadas [15]. As variáveis são geradas depois de muito esforço humano, coletando, limpando, pré-processando, analisando e transformando dados.

De acordo com [16], *Feature Engineering* é a etapa que mais consome tempo e capacidade humana, dentre todas as etapas citadas pelo *CRISP-DM*. Essa etapa é composta por outras duas: *Feature Extraction* e *Feature Selection*.

A extração das variáveis, *Feature Extraction* em português, é a fase onde o cientista de dados passa seu tempo escolhendo aquelas variáveis que entende serem importantes para o modelo de aprendizado de máquina. A etapa de seleção de variáveis, *Feature Selection* em português, é compreendida como sendo a remoção de variáveis que não consigam trazer informações úteis para os algoritmos de aprendizado de máquina. Geralmente, essa etapa é automatizada e remove variáveis que têm pouca variância, contêm muitos valores nulos ou não têm correlação com a variável que se deseja predizer.

Ainda de acordo com [16], as maneiras de se realizar *Feature Engineering* podem ser classificadas em 4 maneiras:

- (a) Escolha manual das variáveis, guiadas por conhecimentos do domínio
- (b) Recomendação de variáveis automaticamente (método utilizado nesse projeto)
- (c) Transformação de variáveis, como o método *Principal Component Analysis* (*PCA*)
- (d) Aprendizado automático utilizando arquiteturas profundas, estilo *black box*.

Dentre as 4 maneiras citadas acima, apenas (a) e (b) geram variáveis interpretáveis aos humanos. Desse modo, essas são utilizadas em problemas que requerem prognósticos da predição feita. Algumas áreas que possuem esse requisito são: médica, de inteligência de negócios e análise de crédito.

A etapa de engenharia de variáveis é dependente do tipo de problema abordado. Porém, é possível encontrar certos padrões na realização da mesma. Em um estudo realizado por [17], algumas técnicas que geralmente ocorrem nesse processo foram listadas:

- 1. Transformação em uma variável
- 2. Taxa ou frequência das variáveis categóricas
- 3. Combinação das variáveis importantes
- 4. Relacionamento entre variáveis
- 5. Alteração do tipo de dados
- 6. Diferenças relativas entre variáveis
- 7. Transformações cartesianas
- 8. Transformações de variáveis contínuas em faixas
- 9. Dados de séries temporais por janelas
- 11. *One hot encoding* para variáveis sequenciais

3.2 Geração Automática de *Features*

A geração automática de *features* pode ser realizada de maneiras diferentes, dependendo dos requisitos do problema abordado. O principal requisito que diverge as maneiras é quanto a necessidade da interpretabilidade das *features*. Levando em conta que a natureza do problema abordado nesse projeto é da área de inteligência de negócios e requer variáveis interpretáveis aos tomadores de decisão, a maneira utilizada necessitará obedecer a esse requisito. Dessa forma, a utilização de algoritmos profundos (*Deep Learning*) ou transformadores de *features* (*PCA*) não servirão.

A geração automática de *features* abordada deverá ser capaz de gerar variáveis interpretáveis. Para cumprir esse requisito, existem algumas técnicas e ferramentas disponíveis atualmente:

- *Deep Feature Synthesis* (*DFS*)
- *Cognito*
- *ExploreKit*
- *One Button Machine* (*OBM*)

Essas técnicas são classificadas em 2 ramos, de acordo com [18]: (a) *Expansion-Reduction* e (b) *Evolution-Centric*.

A técnica de *Expansion-Reduction* engloba os algoritmos *OBM* e *DFS* e, basicamente, funciona gerando um número muito grande de variáveis, seguida por uma seleção das melhores.

O método *Evolution-Centric* engloba o algoritmo *ExploreKit* e trabalha adicionando uma variável de cada vez, seguido por uma avaliação da acurácia do modelo final.

Dessa forma, o método (b) é mais escalável que (a). Porém, é extremamente lento pelo fato de possuir em seu núcleo o treinamento e a avaliação do modelo a cada nova variável adicionada. Já, o modelo (a) não depende do treinamento de um modelo para avaliação, porém têm um desempenho muito afetado pelo grande tamanho do seu espaço de variáveis ($m \times k$), onde m é o numero de variáveis e k o número de transformações.

Cognito é a única técnica, dentre essas, que não faz parte dos 2 ramos mencionados. Essa técnica é caracterizada por realizar uma organização hierárquica de transformações. Essas transformações são realizadas em pequenos conjuntos de variáveis que depois são unidas por métodos exploratórios de busca, guiados por uma métrica de performance.

Quantos às ferramentas disponíveis, o algoritmo *DFS* possui uma implementação em *Python*, *Open Source*, chamada *Featuretools*. O *Featuretools* vai ser explicado mais

adiante, porém, alguns dados sobre ele são que possui 13 contribuidores ativos, está na versão 0.4 e já foi implementado em produção por uma grande empresa, como mostra [2].

O algoritmo *OBM* é licenciado pela empresa *IBM* e não possui implementação *Open Source*, logo não fui utilizado nesse estudo. Nessa linha, o algoritmo *Cognito* também não foi utilizado por não estar disponível.

O algoritmo *ExploreKit* tem uma versão de implementação publicada *Open Source*, mas está muito desatualizada e contém apenas 1 contribuidor.

Há mais uma técnica para geração automática de variáveis chamada *TransmogrifAI*, desenvolvida pela empresa *SalesForce*. Porém, essa ferramenta não foi publicada em nenhum artigo e seu funcionamento ainda é desconhecido. O pouco que se sabe é que a técnica é capaz de gerar novas variáveis a partir dos tipos de dados das variáveis. Esse projeto tem mais contribuidores, cerca de 19 membros, porém é realizada em uma linguagem que possui uma maior barreira de entrada: *Scala*.

3.2.1 Deep Feature Synthesis

Dentre os algoritmos citados anteriormente, para realizar esse projeto foi escolhido o algoritmo *Deep Feature Synthesis* (*DFS*). Esse algoritmo foi implementado e está disponível *Open Source* para utilização como um pacote da linguagem *Python*. Dessa forma, foi o algoritmo mais acessível encontrado que promete encontrar uma solução para o problema em questão, utilizando a engenheria automática de variáveis.

Como exposto em [19], o algoritmo gera automaticamente as variáveis para bases relacionais. Em sua essência, o algoritmo segue os relacionamentos dos dados e, sequencialmente, aplica funções matemáticas primitivas para criar novas variáveis. Além disso, ele é capaz de agrupar variáveis, gerando uma nova variável a partir de outras, de acordo com uma certa profundidade d.

O *DFS* recebe como entrada um conjunto de entidades relacionadas e as tabelas pertencentes a cada. Em notação, podem existir K entidades $E^{1...K}$, onde cada uma possui $1...J$ variáveis. Assim, uma entrada específica é denotada como $x^k_{i,j}$, sendo o valor da variável j para a $i^{ésima}$ instância da $k^{ésima}$ entidade.

Dessa forma, o algoritmo consegue aplicar funções matemáticas em 2 níveis: a nivel de entidade e de relação.

Para o primeiro nível, de entidade, são calculadas variáveis somente para a entidade em questão. Essas variáveis são chamadas de *Entity Features (efeat)* ou *transformações*. Alguns exemplos são funções que arredondam valores numéricos ou que transformam uma data em dia da semana, hora do dia, dia do mês, ano ou mês.

Para o segundo nível, de relação, são calculadas variáveis pelo relacionamento entre

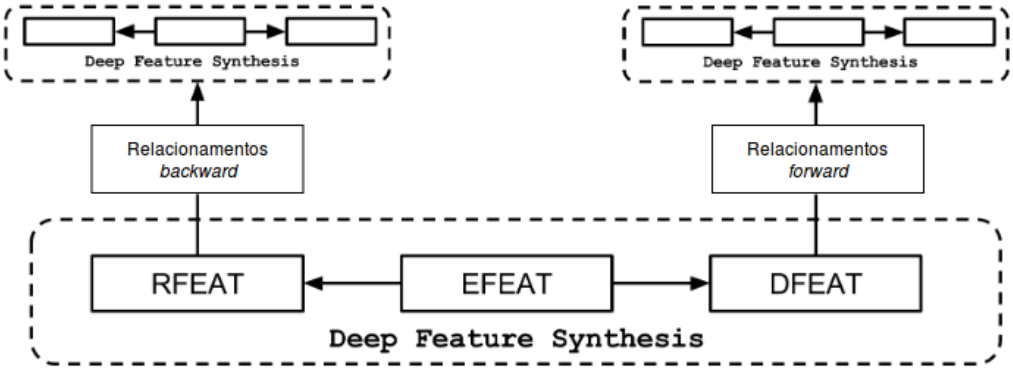
2 entidades. Essas 2 entidades podem se relacionar de 2 maneiras distintas: *forward* ou *backward*. Ambas são chamadas de variáveis de *agregação*.

Uma relação *forward* é entre uma instância m de uma entidade E^k com apenas uma única instância i de outra entidade E^i. Esse tipo de relacionamento ocorre quando i tem uma dependência explícita de m. É chamada de ($dfeat$) ou variável direta.

Uma relação *backward* é entre uma instância i em E^i com todas as instâncias $m = 1...M$ em E^k. Esse tipo de relacionamento ocorre em relacionamentos de um para muitos. É chamada de ($rfeat$) ou variável relacional. Alguns exemplos dessas variáveis são operações de agregação como $COUNT$, MIN e MAX.

Com esses conceitos, o algoritmo funciona de forma que todas essas variáveis ($efeat$, $dfeat$, $rfeat$) sejam extraídas para uma entidade alvo. Para isso, primeiro são calculadas as variáveis de relação, para que a variável de entidade seja aplicada depois. A Figura 4 traz uma explicação visual da ordem dos cálculos do algoritmo DFS.

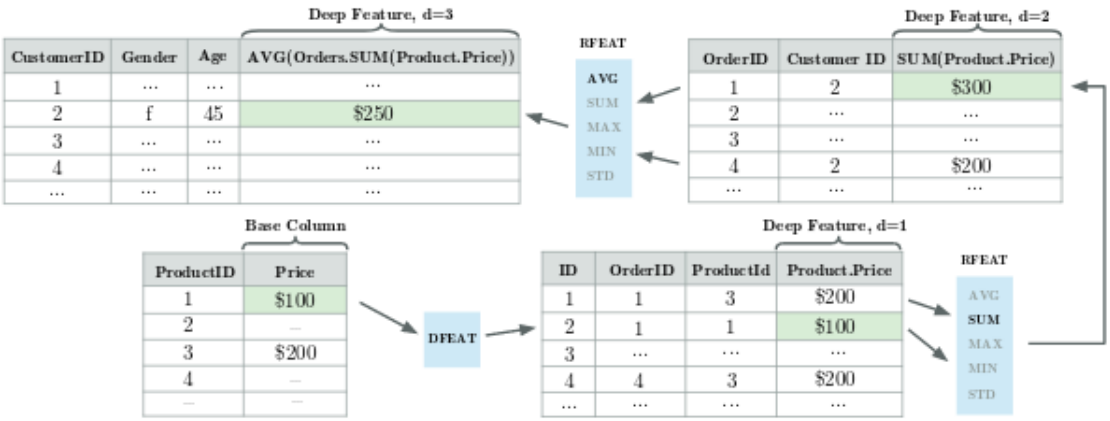
Figura 4 – Algoritmo DFS para criação de variáveis a partir de relacionamentos



Fonte: Adapdato de 19.

Para ter um exemplo mais claro de como o algoritmo funciona com suas funções primitivas, de relação e entidade, e como lida com a profundidade, um exemplo foi retirado de [19] e está na Figura 5. Nesse exemplo, há a retirada de uma $dfeat$ (preço do produto) da tabela de produtos e levada à tabela de pedidos. Na tabela de pedidos, é realizada uma variável $rfeat$ de soma em cima de todos os produtos de cada pedido. Após isso, é calculado outra $rfeat$ para verificar a média dos produtos comprados para cada usuário (entidade alvo do algoritmo). Dessa forma, o algoritmo conseguiu atingir 3 níveis de profundidade para produzir uma variável para a entidade alvo.

Figura 5 – Exemplo de cálculo do algoritmo DFS em inglês



Fonte: 19

3.3 Modelagem Preditiva

Uma modelagem preditiva, como citada no início desse capítulo, é um problema de aprendizado de máquina supervisionado. Dessa forma, são fornecidos dados sobre os acontecimentos e também o resultado de cada acontecimento. Assim, o algoritmo de aprendizado gera um modelo para prever os próximos acontecimentos.

A modelagem preditiva tenta antecipar um evento, baseado no modelo criado a partir dos dados. Essa predição pode ser realizada para variáveis contínuas, chamado assim de um problema de regressão, ou para variáveis discretas, sendo um problema de classificação.

3.3.1 Modelo Autorregressivo Defasado

Para modelos preditivos que lidam com dados temporais, os dados precisam estar estruturados de uma maneira um pouco diferente. Geralmente, ele inclui defasagens da variável dependente entre as variáveis explanatórias. Dessa forma, esse modelo de dados é chamado de autoregressivo [20].

Nesse projeto, a utilização desse modelo é requisitada não só pois variáveis defasadas dão melhores informações ao algoritmo de aprendizado, mas também pois algumas variáveis de crescimento são necessárias para o cálculo da variável resposta. Dessa forma, foram calculadas as variáveis no tempo t, $t-1$ e $t-2$ para compor o conjunto de variáveis explanatórias. E, para compor a variável resposta, foram calculadas as variáveis em $t+1$, $t+2$ e $t+3$.

3.3.2 Aprendizado de Máquina Automático

O aprendizado de máquina automático surgiu pela demanda da indústria em poder realizar aplicações com inteligência artificial sem contar com um time específico de cientista de dados [21].

Os sistemas que realizam o aprendizado automático são capazes de testar múltiplos algoritmos e otimizar seus parâmetros, para que apenas com os dados de entrada, crie-se um modelo. Dessa forma, o *AutoML* supre as necessidades de implementar vários algoritmos e depois verificar qual reagiu melhor ao problema, testando vários e até criando agrupamentos de algoritmos rapidamente, chegando em resultados ótimos.

Nesse trabalho, a ferramenta escolhida se chama *H*20. Ela é capaz de realizar validação cruzada, padronização dos dados, transformação de categorias em variáveis binárias, busca exaustiva por hiper-parâmetros e cálculo de diversas métricas, de acordo com o problema.

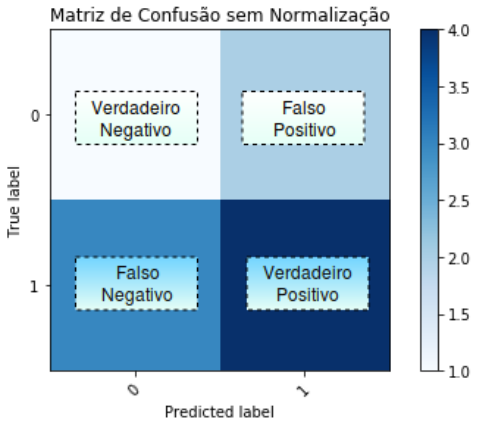
3.3.3 Métricas de Avaliação

Para avaliar qual modelo de aprendizado de máquina se ajustaria melhor e daria uma ideia de como ele prediz as classes, foram utilizadas várias medidas de avaliação. Elas diferem em fórmulas e aplicações.

A principal e mais simples métrica utilizada é a Acurácia. Ela compara a saída do modelo com as classes esperadas para cada entrada. Assim, uma relação entre acertos e o total de entradas submetidas é calculada.

Outras métricas são utilizadas para a avaliação de modelos, que estão relacionadas com o problema em questão. Elas se baseiam em conceitos da matriz de confusão. Uma matriz de confusão é exposta na Figura 6 para melhor clareza dos conceitos.

Figura 6 – Uma Matriz de Confusão com seus conceitos dentro.



Essas métricas diferentes da Acurácia são a Precisão, *Recall*, Especificidade, *F1-Score* e *Cohen-Kappa*.

A Precisão avalia a taxa de dados classificados como corretos e que na verdade eram corretos. Ou seja, é uma taxa entre verdadeiros positivos pela soma de verdadeiros e falsos positivos. A precisão será máxima se não existirem falsos positivos.

$$\text{Precisão} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Positivos}}$$

O *Recall* será máximo se não existirem falsos negativos. Assim, ele avalia a taxa entre os verdadeiros positivos pela soma de verdadeiros positivos com falsos negativos.

$$\textit{Recall} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Negativos}}$$

Já, o *F1-Score* realiza uma média harmônica entre as duas anteriores, trazendo uma visão que junta as duas métricas. Como se trata de uma média harmônica, ela acusa valores que são pequenos em alguma das métricas anteriores, decaindo muito seu valor se ou a Precisão, ou o *Recall* forem baixos.

$$\text{F1} = \frac{2 * \text{Precisão} * \textit{Recall}}{\text{Precisão} + \textit{Recall}}$$

A Especificidade é uma métrica de fácil cálculo, que mede a proporção de acerto nos casos negativos. Essa métrica tende a ser alta em problemas cujo desbalanceamento dos dados é grande em favor da classe negativa. O seu cálculo é o seguinte:

$$\text{Especificidade} = \frac{\text{Verdadeiros Negativos}}{\text{Verdadeiros Negativos} + \text{Falsos Positivos}}$$

Por último, há a métrica de *Cohen-Kappa* que visa comparar a assertividade de um modelo contra um algoritmo aleatório que vai tentar adivinhar o resultado a partir da distribuição da amostra de dados. Dessa forma, essa métrica compara a acurácia atual com uma acurácia esperada. Se a acurácia esperada for alta, é difícil ter um *Cohen-Kappa* alto, pois a sua fórmula é a seguinte:

$$\text{Cohen-Kappa} = \frac{\text{Acurácia} - \text{Acurácia Esperada}}{1 - \text{Acurácia Esperada}}$$

Onde a *Acurácia Esperada* é a probabilidade de concordância randômica, calculada da forma:

$$\text{Acurácia Esperada} = \frac{\text{Falso Positivo} + \text{Verdadeiro Positivo}}{\text{Total}} * \frac{\text{Falso Negativo} + \text{Verdadeiro Positivo}}{\text{Total}}$$

Todas essas métricas de avaliação variam entre 0 e 1.

3.4 Empresas de Alto Crescimento

Nesse projeto, a variável resposta está totalmente atrelada à empresas de alto crescimento. De acordo com [22], empresas de alto crescimento atraíram atenção ultimamente pois um pequeno número dessas empresas acabam gerando uma quantia muito grande de emprego. Logo, elas geram um grande impacto na economia.

A definição de uma empresa de alto crescimento não é muito clara e pode ser realizada de diversas formas. No estudo de [23], duas formas foram citadas: (a) Empresas de Alto Crescimento podem ser entendidas como as firmas que crescem a um determinado ritmo, ou acima dele, por um período intensivo e observável (por exemplo, empresas que crescem pelo menos 20% ao ano por 3 anos consecutivos) [22]; (b) Referem-se à parcela de empresas em uma população que experimenta o maior crescimento durante um dado período (por exemplo, as 10% empresas com maior crescimento no ano de 2016).

Além disso, os indicadores mais comuns para a identificação de firmas de alto crescimento são o nível de emprego ou de faturamento [24]. Embora tais indicadores sejam apenas discretamente correlacionados, os resultados dos estudos empíricos não variam muito em função da escolha do indicador [22].

4 Solução Proposta

O problema abordado nesse projeto se trata de uma predição para empresas brasileiras de alto crescimento. Sendo assim, é necessário que a solução proposta leve em consideração dados momentâneos e passados para conseguir criar um modelo que prediz o valor futuro do crescimento das empresas.

Para haver um modelo preditivo, o passo inicial é estruturar os dados em um formato que auxilie o aprendizado do mesmo. Após essa etapa, segue-se o fluxo para selecionar variáveis úteis, remover as variáveis não-explicativas, criar o modelo de aprendizado até chegar na parte da predição. Essas etapas utilizarão as técnicas descritas nesse capítulo.

Como o principal objetivo desse trabalho é validar a utilização de técnicas que possam automatizar parte ou totalmente o trabalho mais custoso de um cientista de dados, primeiro foram estudadas diversas técnicas de geração automática de variáveis. Então, uma técnica foi escolhida e os dados foram estruturados de forma que a ferramenta requisita.

Como técnica de geração automática de *features*, o algoritmo chamado *Deep Feature Synthesis*, aplicado com a ferramenta *Featuretools*, foi escolhido por ser *Open Source*, ter uma curva de aprendizado suave, utilizar dados relacionais como geradores de variáveis, ter a capacidade de criar inúmeras variáveis interpretáveis por humanos e facilitar na implementação de modelos preditivos, com a escolha de janelas e limites de tempo que se deseja levar em consideração.

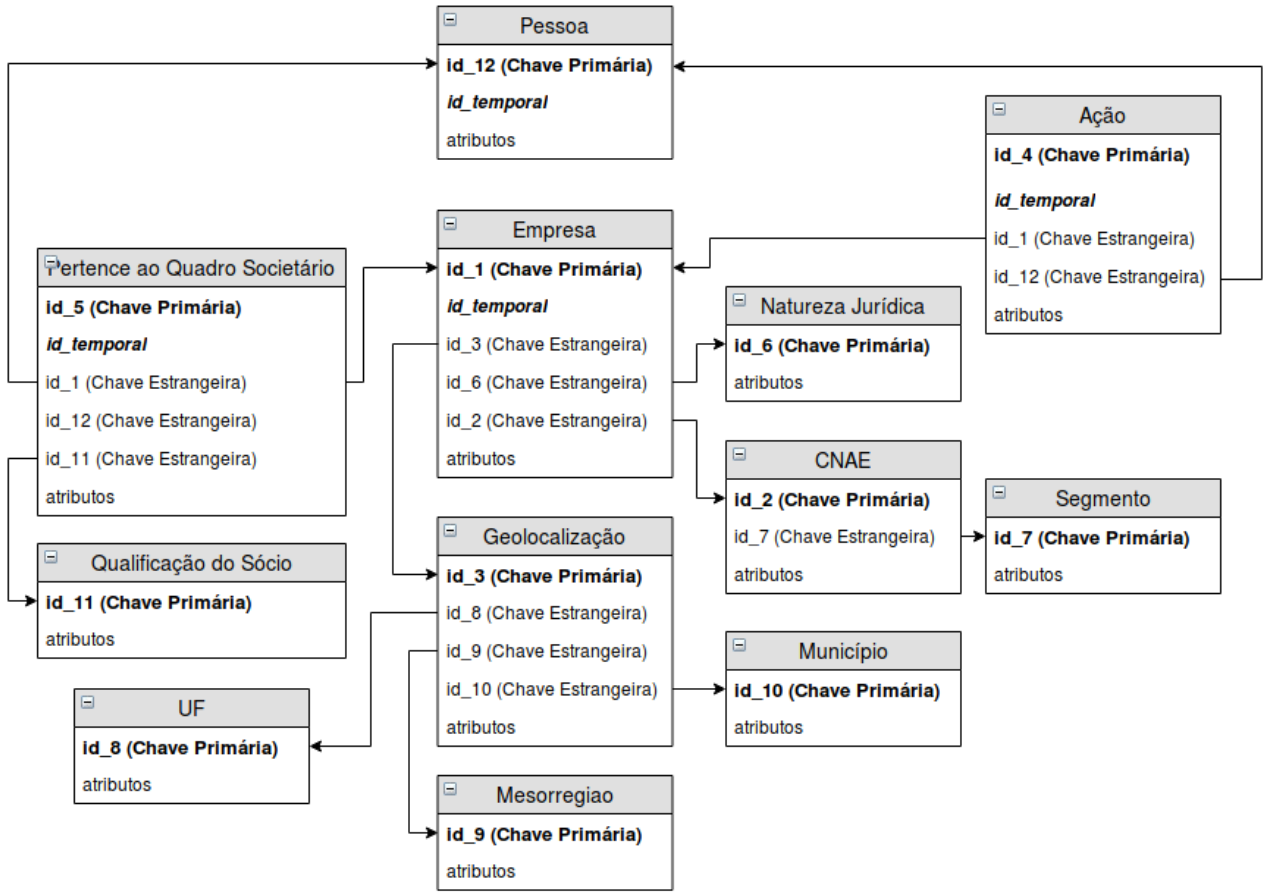
Após a escolha do *Featuretools*, viu-se a necessidade de estruturar os dados de uma forma relacional, com entidades e suas relações.

4.1 Dados Relacionais

Os dados precisam estar estruturados em formato relacional para o *Deep Feature Synthesis* poder entender as relações entre os dados e gerar mais variáveis informativas. Dessa forma, depois de coletados os dados crus, eles foram modelado em entidades, relacionamentos e chaves primárias e estrangeiras. No final, os dados estruturados que servirão de base para o *DFS* devem estar como na Figura 7.

Além das relações, os dados transacionais são muito importantes para a geração das variáveis. Assim, é necessário estruturar os dados para que cada registro seja uma nova transação na base de dados, contendo sua data de acontecimento. Duas entidades que são transacionais nesse esquema de dados são a *Ação* e *Pertence ao Quadro Societário*.

Figura 7 – Dados estruturados em formato relacional



4.2 Definição da Variável Resposta

Conforme explicado na seção 3.4, uma empresa de alto crescimento pode ser caracterizada de diversas formas. A definição utilizada nesse projeto será de acordo com a definição de [22], onde cita-se que uma EAC cresce a um determinado ritmo ou acima dele, por um período intensivo e observável. Com essa base e um estudo realizado na empresa, em conjunto com uma consultoria em economia, a variável resposta do crescimento da empresa será em cima da média geométrica dos próximo 3 anos consecutivos. Sendo calculada da seguinte forma: se essa média for superior a 0.17 (17%), pode-se afirmar que a empresa será de alto crescimento. Com isso, o problema de aprendizado de máquina será uma classificação binária.

A fórmula matemática (4.1) que define uma EAC adotada nesse projeto está

explicada abaixo.

$$EAC = \begin{cases} 1, & \text{se } \sqrt[3]{(crescimento_1 + 1)(crescimento_2 + 1)(crescimento_3 + 1)} - 1 > 0.17 \\ 0, & \text{caso contrário} \end{cases} \tag{4.1}$$

Onde o $crescimento_i$ com $i = [1, 2, 3]$ é dado por:

$$crescimento_i = \frac{qtd\_funcionarios_{t+i} - qtd\_funcionarios_{t+i-1}}{qtd\_funcionarios_{t+i-1}}$$

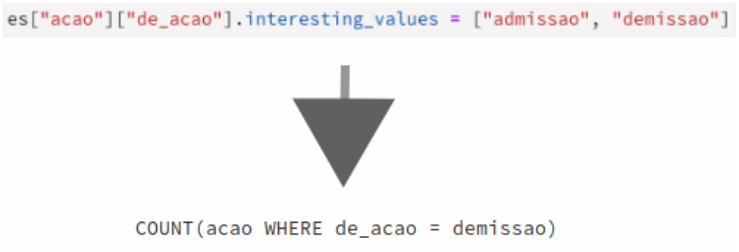
4.3 Geração Automática de Variáveis

Com as bases de dados em estrutura relacional, o *Featuretools* necessita apenas da criação de um conjunto principal de entidades, contendo todas as entidades e seus relacionamentos. Também, ele deixa claro que pode utilizar índices temporais, para a criação de variáveis respeitando um tempo limite definido.

A partir dessa definição, a ferramenta gera inúmeras variáveis, partindo sempre das suas primitivas, de transformação ou agregação, chegando em variáveis compostas, tanto por diversas primitivas, quanto pelo caminho entre relacionamentos de entidades.

Algumas definições complementares podem ser realizadas, como a indicação de valores importantes em variáveis categóricas pelo método *interesting_variables()*. Esse método é responsável por alertar o algortimo *DFS* que caso esses valores estejam presentes, uma nova variável deverá ser incrementada, como no caso de um *COUNT*. Um exemplo de definição e criação de uma variável a partir desse método está na Figura 8. Nela, há a definição de uma *variável interessante*: *de_ação* igual a *demissão* ou *admissão* na entidade *ação*. Na mesma figura, mas abaixo da seta, está uma das possíveis variáveis geradas com a definição de cima, utilizando a agregação *COUNT*.

Figura 8 – Exemplo de *feature* gerada a partir do método de variáveis interessantes

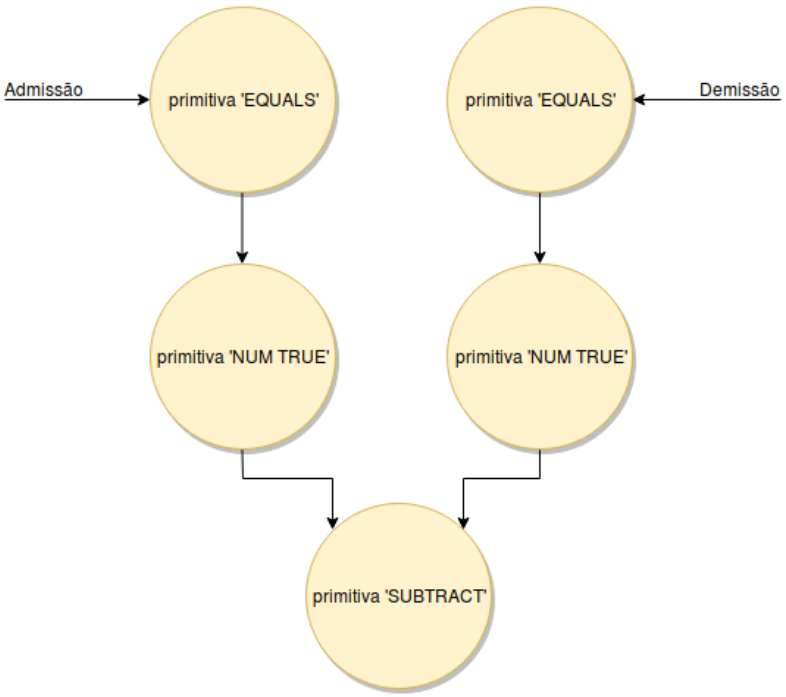


Além disso, variáveis *Seed* podem ser configuradas para atuar como uma primitiva, porém de forma totalmente personalizada. Para esse projeto, será necessário utilizar uma variável personalizada que calcula o número de funcionários das empresas, a partir de

dados transacionais. A Figura 9 traz o exemplo da variável personalizada *qtd_funcionarios* criada dentro da ferramenta.

Figura 9 – Definição de uma variável *seed*, capaz de ser utilizada para novas agregações do *Featuretools*

```
admissao = ft.primitives.Equals(es["acao"]["de_acao"], 'admissao')

demissao = ft.primitives.Equals(es["acao"]["de_acao"], 'demissao')

sum_adm_companies = ft.primitives.NumTrue(admissao, es['companies'])

sum_dem_companies = ft.primitives.NumTrue(demissao, es['companies'])

seed_qtd_func = sum_adm_companies - sum_dem_companies

seed_qtd_func = seed_qtd_func.rename('qtd_funcionarios')
```

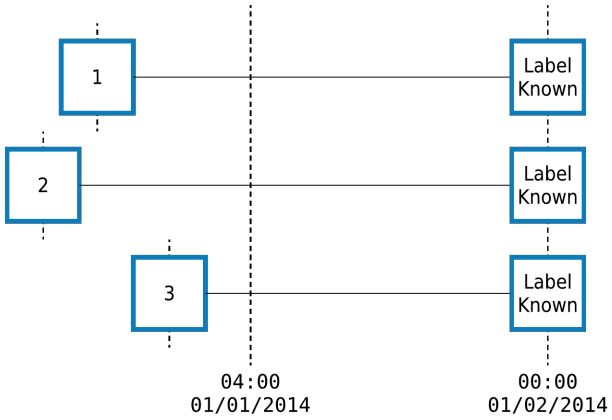


A ferramenta *Featuretools* possibilita a indicação de um *cutoff_time*, que limita o cálculo das variáveis até esse momento. Essa capacidade possibilita a criação de variáveis que calculam certa propriedade até o momento explicitado. Assim, é necessário definir um tempo limite em cada janela de tempo a ser calculada.

Por exemplo, na Figura 10, existem 3 instâncias de uma entidade que possuem informações temporais. Com o *cutoff_time* definido às 4h00, nenhuma informação após isso vai ser levada em consideração no modelo preditivo. O objetivo é evitar que a resposta esteja contida nos dados de treinamento.

Após a geração de variáveis automaticamente, cada empresa terá um conjunto de variáveis em diferentes tempos limites (*temporal_cutoffs*). Esses *temporal_cutoffs* são um conjunto de instâncias com seus respectivos *cutoff_time* separados por alguma janela de

Figura 10 – Exemplo de *cutoff_time* às 4:00.



Fonte: *Featuretools*

tempo definida. Por exemplo, na Figura 11 nota-se a definição de três janelas de tempo, com um período anual.

Figura 11 – Exemplo de um *temporal_cutoff* com 3 janelas anuais de tempo.

	time
120573	2012-12-31
120574	2013-12-31
120575	2014-12-31

Desse modo, é possível calcular novas variáveis para uma janela temporal, que representa um valor real até aquele momento definido.

Essas variáveis independentes em momentos distintos do tempo são essenciais para formar os modelos autorregressivos defasados. Depois dessa etapa, é necessário outro passo para criar a relação entre uma variável no tempo t com outra no tempo $t-1$. Assim, cria-se outra variável personalizada, que compara essas variáveis de janelas diferentes.

Após as duas gerações automáticas, basta isolar a variável resposta (variável dependente) das outras. Assim, restam o conjunto X de variáveis explanatórias e a variável resposta (dependente) y.

4.4 Seleção de Variáveis

Antes de aplicar um modelo de aprendizado de máquina nas inúmeras variáveis geradas, é recomendado que alguma técnica de seleção de variáveis seja utilizada. Essa seleção evita que variáveis sem informação útil possam confundir o modelo ou apenas ocupar processamento desnecessário.

Duas técnicas de seleção de variáveis foram escolhidas. A primeira retira variáveis idênticas ou que possuem muitos valores nulos. A segunda seleciona apenas aquelas que possuem alto coeficiente de variação nos dados.

O coeficiente de variação é calculado pelo desvio padrão dividido pela média:

$$CV_j = \frac{\sqrt{\frac{\sum_{i=1}^{N}(x_{i,j}-\overline{x_j})^2}{N-1}}}{\overline{x_j}} \tag{4.2}$$

onde $\{x_{1,j}, x_{2,j}, \ldots, x_{N,j}\}$ são os valores dos registros da variável j em questão, $\overline{x_j}$ é a média desses valores e N é o número de registros.

4.5 Aprendizado de Máquina

Com as variáveis todas geradas e selecionadas, basta aplicar um modelo de aprendizado de máquina capaz de elaborar um modelo preditivo para a variável resposta y, utilizando as variáveis de entrada X.

Para essa etapa, foram utilizados um *framework* capaz de testar vários algoritmos diferentes, com uma seleção automática de hiper-parâmetros, chamado *H2O - AutoML* e uma biblioteca de *Python* capaz de criar algoritmos de árvore chamado *Catboost*.

4.6 Métricas e Comparações

Para avaliar o resultado do modelo gerado, foram utilizadas as métricas de Acurácia, Precisão, *Recall*, *Especificidade*, *F1-Score* e *Cohen-Kappa*, explicadas na subseção 3.3.3.

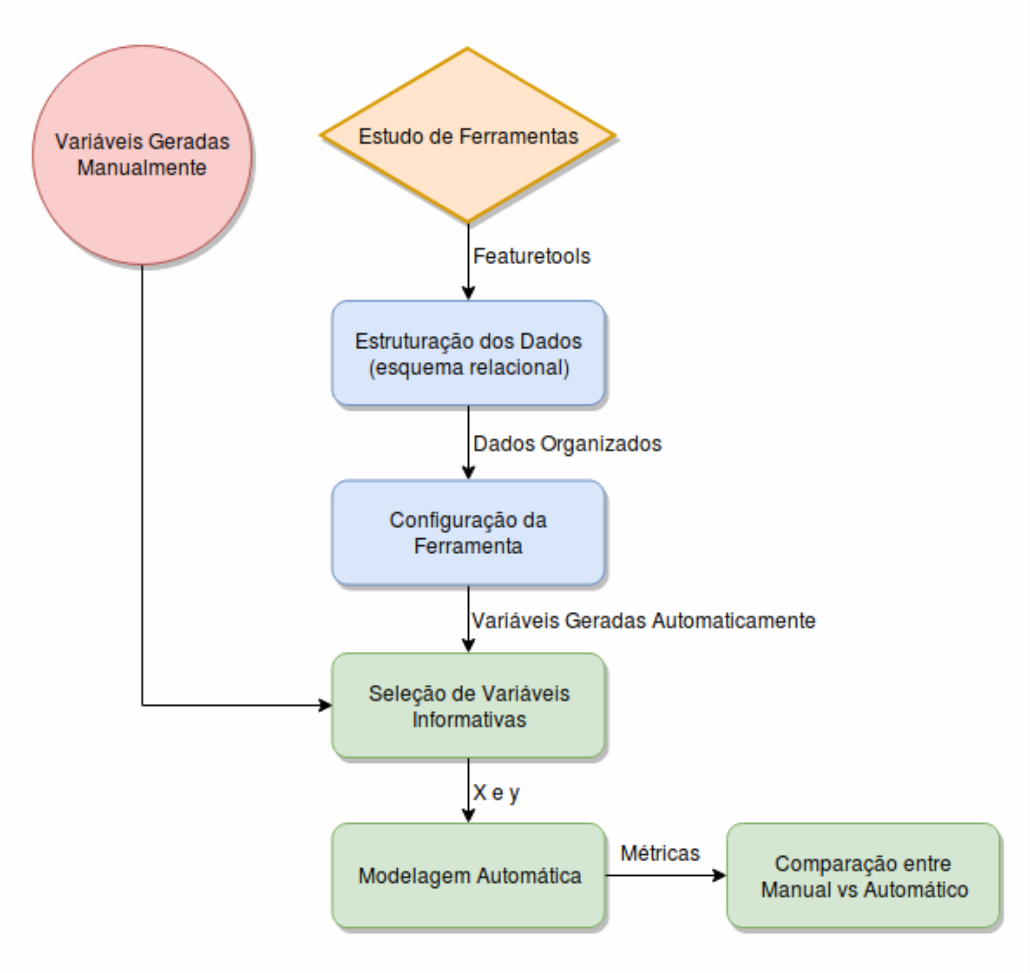
As métricas mais observadas nesse experimento foram a precisão e o *F1-Score*, pois deseja-se que o algoritmo consiga acertar aquelas predições que fizer. Mas, além disso, foi necessário também que ele não atuasse em poucos casos.

No fim, os modelos gerados a partir das variáveis automáticas foram comparados a um modelo manualmente criado por um cientista de dados da *Neoway*.

4.7 Visão Geral da Metodologia

Em resumo, a metodologia utilizada pode ser visualizada na Figura 12.

Figura 12 – Metodologia proposta para a solução do problema



5 Implementação

5.1 Ferramentas

Como ambiente de desenvolvimento foram utilizados o *Jupyter* com a linguagem de programação *Python*. Para manipulação dos dados foi escolhida a biblioteca *Pandas*. Na etapa da geração das variáveis automáticas, a ferramenta *Featuretools* foi utilizada. Logo após a geração de variáveis automaticamente, utilizou-se da biblioteca *pandas-profiling* para o *Feature Selection*. Finalmente, para a modelagem, as ferramentas *H2O* e *Catboost* foram utilizadas, pela capacidade de rápida implementação de vários algoritmos otimizados.

5.1.1 *Featuretools*

Figura 13 – Logo da ferramenta *Featuretools*.

Fonte: *Featuretools*

A biblioteca *Featuretools*, cuja logo está na Figura 13, foi desenvolvida pela *Featurelabs* e aplica o algoritmo *DFS*, citado no capítulo de revisão bibliográfica.

A biblioteca traz os conceitos utilizados no algoritmo de [19], como entidade, relacionamentos, variáveis interessantes, variáveis primitivas, tempo limite entre outras. É a aplicação da teoria na prática.

Para a geração automática de variáveis, basta criar um conjunto principal de entidades (*EntitySet*), que vai conter todas as entidades dos dados. Essas entidades podem ser importadas como *DataFrames* do *Pandas* ou normalizadas, a partir de uma entidade já criada. Os relacionamentos deverão ser adicionados posteriormente, ligando uma chave estrangeira de uma entidade a uma chave primária de outra. Com isso, já é possível gerar *features* automáticas chamando o método *dfs()*. Porém, ainda pode-se adicionar outras características que agreguem valor às variáveis geradas. Algumas dessas características são:

- (a) Variáveis interessantes;
- (b) Variáveis semente;

- (c) Variáveis personalizadas;
- (d) Tempo limite;
- (e) Conjunto de tempos limites (janelas temporais).

A característica (a) é útil para gerar mais variáveis quando uma condição é importante no contexto do problema. Ela gera um evento sempre que a condição for notada, assim, novas variáveis de agregação podem ser geradas em cima, como um *COUNT* desses eventos. Um exemplo de utilização desse método já foi explicado na Figura 8.

As outras características são essenciais para o problema em questão, pois possibilitaram a criação da variável resposta pela ferramenta. A característica (b) é responsável por criar uma nova variável que vai estar presente desde o início no conjunto de variáveis, assim, ela pode ser utilizada como base para operações ou relações profundas. Com ela, geralmente vem a característica (c) agregada, pois essa permite a criação de uma nova variável, totalmente diferente das outras primitivas que a ferramenta possui por padrão (inclusive, essas primitivas padrão estão no Apêndice A). Então, com a característica (c) foi possível criar a variável *qtd_funcionarios*, representando a quantidade de funcionários de uma empresa. O código da nova variável personalizada pode ser visualizado na Figura 9, do capítulo anterior. Ela é uma combinação de outras variáveis com operações aplicadas.

Após essas configurações, a matriz de variáveis pode ser gerada automaticamente chamando o método *dfs()*. Dentre os possíveis parâmetros desse método, os principais tratam-se das variáveis primitivas que pode-se adicionar e a profundidade da geração. As variáveis primitivas dividem-se em variáveis de transformação e agregação. Além dessas, existem as variáveis específicas para serem aplicadas em variáveis interessantes. Esse conjunto de variáveis são o coração do projeto, que cria automaticamente as *features* baseado-se em relações entre as entidades, em variáveis primitivas e na agregação dessas primitivas, conforme especificado na profundidade da geração.

As variáveis primitivas de transformação levam em consideração apenas o registro para criar características diferentes das existentes. Elas estão listadas no Apêndice A.

Já, as variáveis primitivas de agregação levam em consideração todos os registros da mesma entidade que se deseja calcular algo. Elas também estão listadas no Apêndice A.

No final da geração, inúmeras variáveis são criadas. Essa matriz possui colunas com uma sintaxe própria do *Featuretools*. Para explicá-la e exemplificá-la, a Figura 14 traz algumas possíveis variáveis geradas automaticamente.

Da Figura 14 podemos interpretar a sintaxe da seguinte maneira: Quantidade de empresas cujo porte é igual a *EPP* ou *ME* que pertencem à mesma natureza jurídica da empresa em questão.

Figura 14 – Exemplo de 2 *features* automáticas geradas.

natureza_jurid.COUNT(companies WHERE nm_porte = EPP)	natureza_jurid.COUNT(companies WHERE nm_porte = ME)
37.0	44.0
3651.0	16070.0

5.1.2 H2O

O *H2O* é um *Framework* específico para *AutoML*, sua logo está na Figura 15. Ele foi construído em *Java*, porém tem *API* para *Python*. Basicamente, o *H2O* consegue aplicar múltiplos algoritmos de aprendizado de máquina, com hiperparâmetros otimizados. Esse algoritmos são comparados com uma métrica configurada a priori. Alguns algoritmos utilizados para problemas de classificação são: *Distributed Random Forest (DRF)*, *Deep Learning*, *Gradient Boosting Machine (GBM)*; e para problemas de regressão são: *Generalized Linear Model (GLM)*, *Distributed Random Forest (DRF)*, *Deep Learning* e *Gradient Boosting Machine (GBM)*, além dos métodos *Stacked Ensemble* que são utilizados para ambos os problemas. Esses algoritmos são também otimizados para rodarem muito rapidamente e com os melhores parâmetros.

Figura 15 – Logo da ferramenta *H2O*.

Fonte: *H2o.ai*

Com essa ferramente, muito tempo desperdiçado na escolha do algoritmo é economizado. Ele já faz o trabalhado de utilizar vários algoritmos e calcular as métricas para todos.

Figura 16 – Logo da biblioteca *Catboost*.

Fonte: *Catboost*

5.1.3 Catboost

Além do *H2O*, a biblioteca *Catboost* foi utilizada para gerar modelos. O *Catboost* não é um *AutoML*, porém ele é prático e possui os melhores resultados em problemas de classificação utilizando árvores de decisão (é o estado-da-arte). Sendo assim, o *Catboost* será capaz de gerar modelos preditivos para comparações com os resultados do *H2O*. Sua logo está na Figura 16.

5.2 Metodologia

A metodologia utilizada foi descrita no capítulo anterior. Agora, mais especificamente será explicada como a solução foi desenvolvida passo a passo.

5.2.1 Escolha da Ferramenta

O primeiro passo do projeto final de curso foi realizar um estudo sobre as principais técnicas atuais de geração automática de variáveis. Nesse estudo, as técnicas que já possuíam ferramentas construídas e disponíveis no formato *Open Source* foram focadas. Então, as ferramentas foram avaliadas nos quesitos:

- Facilidade de aprendizado;
- Geração de variáveis interpretáveis;
- Escalabilidade;
- Contribuidores;
- Maturidade da solução.

Desse modo, obteve o melhor desempenho geral a ferramenta *Featuretools*, que se baseia na teria de *DFS*. Essa ferramenta é desenvolvida em *Python*, que possui uma pequena

curva de aprendizado, comparada com *Scala*, a linguagem da ferramenta *TransmogrifAI*. Essa ferramenta gera variáveis a partir de relacionamentos entre entidades, diferente do *TransmogrifAI*, que gera apenas variáveis a partir dos tipos de dados iniciais. Além disso, possui 12 contribuidores no projeto *Open Source*, contra 1 contribuidor do *ExploreKit* e 19 contribuidores do *TransmogrifAI*. Ainda não possui uma maturidade muito evoluida, por estar apenas na versão 0.4, porém já existem artigos sobre a ferramenta ter sido aplicada em produção de um grande banco para mais de 9 milhões de dados e ter obtido resultados muito satisfatórios para a redução de fraudes [2]. A escalabilidade ainda é um ponto fraco da ferramenta, que utiliza a biblioteca *Pandas* por baixo dos panos e acaba consumindo muito a memória. Em linhas gerais, a ferramenta se mostrou mais competitiva frente as demais por ter uma barreira de entrada pequena e já ter provas de estar funcionando em produção com uma quantidade massiva de dados.

5.2.2 Seleção e Organização dos Dados

Com a escolha da ferramenta *Featuretools*, os dados de entrada do algoritmo necessitam estar em um formato relacional bem estruturado. Dessa forma, os dados crus obtidos foram reestruturados em um formato relacional, onde cada entidade tornou-se uma tabela capaz de se relacionar através de chaves primárias e estrangeiras com as outras entidades.

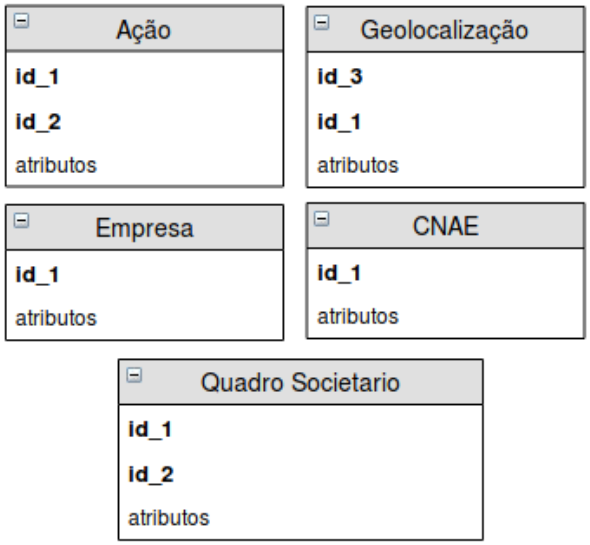
Então, partindo de 5 tabelas cruas, a reestruturação ocorreu e tornaram-se 13 tabelas, com suas novas chaves primárias e estrangeiras. Uma diagramação desses relacionamentos foi gerada a fim de auxiliar no processo de manipulação de dados. Para a comparação, as tabelas cruas são as representadas na Figura 17. Já, as tabelas reestruturadas foram citadas na Figura 7.

Com isso, houveram até tabelas que foram dividas em outras e algumas unidas para que alguns dados de transações ficassem caracterizados como tais e também para adição de índices temporais.

Nesse caso, como as tabelas cruas não tinham um formato totalmente relacional, boa parte do tempo despendido nesse projeto foi na etapa de reestruturação dos dados.

É interessante separar o máximo possível as tabelas em outras tabelas que representem unicamente uma entidade. Como exemplo, a tabela que continha informações do CNAE foi separada em 2, para uma possuir apenas a descrição do SEGMENTO. Dessa forma, o *Featuretools* é possibilitado a calcular novas variáveis de acordo com segmentos iguais, e não só números de CNAE iguais. Para clarificar, CNAE significa Classificação Nacional de Atividade Econômicas e as empresas são obrigadas a declarar quais atividades econômicas estarão exercendo. Um segmento pode englobar várias atividades econômicas. Por exemplo, o CNAE de número 5611203 aponta para o ramo de atividade *LANCHO-*

Figura 17 – Bases de dados antes de serem estruturadas em um formato relacional com as chaves primárias e estrangeiras.



NETES, CASAS DE CHÁ, DE SUCOS E SIMILARES que está dentro do segmento *ALOJAMENTO E ALIMENTAÇÃO*.

Além disso, é muito importante definir os índices das tabelas ou chaves primárias únicas que identifiquem apenas um registro, para que os relacionamentos sejam possíveis de serem declarados.

5.2.3 Configuração do *Featuretools*

Com os dados estruturados, basta configurar o *Featuretools* para gerar as variáveis automaticamente. O primeiro passo foi criar um conjunto principal de entidades e adicionar cada entidade nesse conjunto. Também foi possível normalizar algumas entidades a partir de colunas de outras entidades. Dessa forma, as 13 tabelas mostradas na Figura 7 tornaram-se 15.

A partir do passo que construiu as entidades, os relacionamentos foram definidos e, assim, restaram identificar as variáveis primitivas, semente e interessantes. Com eles, o *Featuretools* é capaz de aplicar a teoria explicada na seção 3.2.1 e gerar variáveis como, por exemplo, o número de empresas na mesma região.

As variáveis primitivas de agregação escolhidas foram: COUNT, MEAN, MODE e SKEW. As variáveis de transformação foram CUM_MEAN e PERCENTILE.

As variáveis interessantes escolhidas foram as expostas na Figura 18.

As variáveis personalizadas e semente foram a já explicada *qtd_funcionarios* (Figura 9) e também as de crescimento, essenciais para a variável resposta, que estarão

Figura 18 – Variáveis interessantes escolhidas.

```
es["companies"]["nm_porte"].interesting_values = ["DEMAIS", "ME", "EPP"]

es["acao"]["de_acao"].interesting_values = ["admissao", "demissao"]
```

melhor explicadas na seção seguinte.

Além disso, foi necessário construir um mapeamento para cada empresa com a data limite de cada janela. Foram escolhidas janelas desde 2011 até 2016, com um período de 1 ano entre elas.

5.2.3.1 Criação da Variável Resposta

Utilizando a nova variável personalizada *qtd_funcionarios* como uma variável *Seed*, foi possível utilizá-la como base para outras agregações e transformações. Assim, a variável resposta quista foi possível de ser realizada, pois ela é um cálculo em cima da variável *qtd_funcionarios*, como mostra a fórmula (4.1).

Passando essa fórmula para código, com métodos do *Featuretools*, pode-se calcular a variável resposta da forma que está na Figura 19. Essa criação requere que uma nova matriz de variáveis seja gerada, então um novo conjunto de entidades foi criado, contendo algumas variáveis numéricas importantes, que se desejava calcular o crescimento ao longo tempo, junto da variável *qtd_funcionarios*, que gera a variável resposta.

Para a geração dessa nova matriz, com o *Featuretools*, foram escolhidas e calculadas, para os anos de 2011, 2012 e 2013, um total de 23 variáveis numéricas (contendo a *qtd_funcionarios*). Dessa forma, o conjunto de entidades ficou com essa entidade macro contendo todas as 23 colunas e mais uma entidade, que foi normalizada a partir da coluna que identifica uma empresa, para representar as empresas (entidade alvo). Assim, aplicou-se as primitivas de agregação *COUNT*, *MODE*, STD e *TREND*, além da variável personalizada de crescimento (comparação entre uma variável no ano t com o a mesma variável no ano $t-1$. O resultado disso foram 95 novas variáveis mais a variável resposta, que se desejava calcular. É importante notar que a única variável que levou em consideração informações além de 2013, foi a variável resposta. As outras todas possuem um *cutoff* na data 31/12/2013.

Figura 19 – Criação da variável resposta através de variáveis do *Featuretools*.

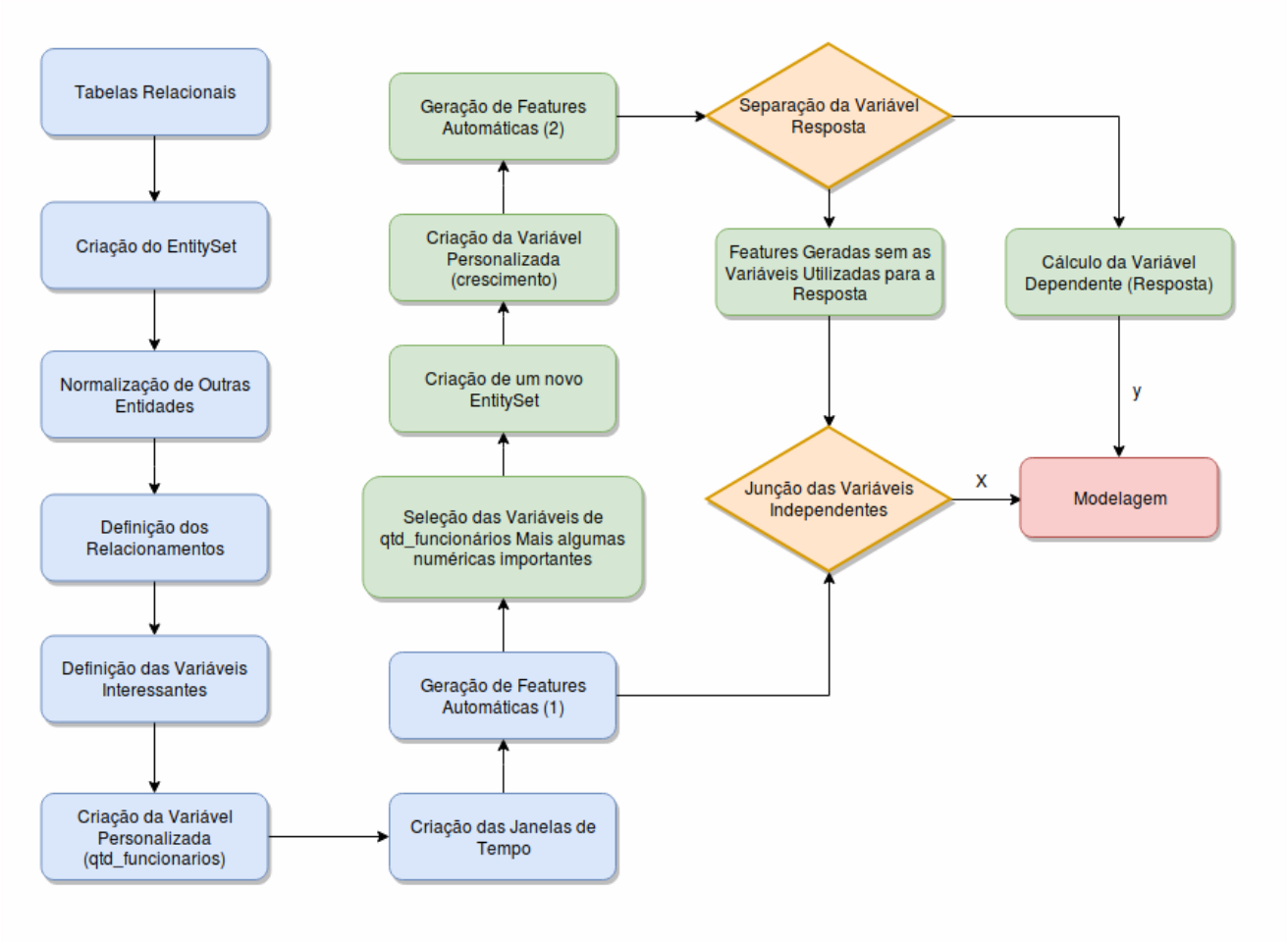
```
def g(dict_, list_, ref, growth):
    """Calculates the growth of two periods"""
    if dict_[list_[ref-growth]]==0: return 0
    return ((dict_[list_[ref-(growth+1)]])/(dict_[list_[ref-growth]]))-1

##########################

def eac_017(datetime, numeric, cutoff_date=None, diff=None, windows=None, ref=None):
    """Set 1 for High Growth Firms (geometric mean to next 3 years above 17%)"""
    list_ = []
    for i in range(1, windows+1):
        list_.append(pd.date_range(periods=i, freq=diff, end=cutoff_date)[0].date())

    dict_ = dict(zip(datetime.dt.date,
                     numeric))

    g1 = g(dict_, list_, ref, 1)
    g2 = g(dict_, list_, ref, 2)
    g3 = g(dict_, list_, ref, 3)

    avg = (np.power([(g1+1)*(g2+1)*(g3+1)],[1/3])-1)[0]

    return 1.0 if (avg > 0.17) else 0.0

##########################

growth_eac_017 = ft.primitives.make_agg_primitive(
    function=eac_017,
    input_types=[Datetime, Numeric],
    return_type=Numeric,
)

##########################

grw_13_eac_017 = growth_eac_017(base_features=[es_top_series['serie']['time'],
                                    es_top_series['serie']['qtd_funcionarios']
                                   ],
                     parent_entity=es_top_series['companies'],
                     cutoff_date='2016-12-31',
                     diff='Y',
                     windows=6,
                     ref=4
                )
```

5.2.3.2 Metodologia Featuretools

No fim, foram gerados 2 tabelas com variáveis automáticas. A primeira contém as variáveis *qtd_funcionarios* para todos os anos das janelas definidas e outras que foram gerados em cima do conjunto de entidades. A segunda contém a variável resposta junto, pois foi aplicada a variável personalizada do crescimento entre as janelas. Para clarificar esse processo, um fluxograma da configuração do *Featuretools* pode ser visualizado na Figura 20.

No total, foram geradas 190 variáveis de maneira automática. Dentre essas, 43 são categóricas.

Figura 20 – Configuração do *Featuretools* para o desenvolvimento da solução



5.2.4 Avaliação e Seleção das Variáveis Geradas

Foram geradas 190 variáveis no total. Para selecionar apenas um conjunto dessas, existem as técnicas de *Feature Selection*. Um conjunto menor de variáveis é recomendado não somente pelo fato de muitas poderem confundir os algoritmos, mas também pois o processamento dos modelos será mais ágil se houver menos variáveis a serem calculadas.

Para esse fim, foram utilizados dois métodos de seleção. No primeiro, foi utilizado a biblioteca *pandas-profiling* para gerar um relatório de todas as variáveis e poder retirar aquelas que seja muito correlacionadas entre si, que possuam muitos zeros ou que possuam um valor constante. Então, utilizando essa biblioteca, foram retiradas 52 variáveis que não auxiliariam os algoritmos de aprendizado de modelo.

Além desse primeiro método, foi testado um segundo método para comparação ao final do trabalho. O segundo método corresponde a uma verificação da correlação entre cada coluna com a variável contínua que gerou a variável resposta. Essa variável que foi comparada a cada uma do conjunto X é a média geométrica do crescimento dos 3 anos

consecutivos, a partir da data de corte. Dessa forma, foram selecionadas as 10 variáveis que possuiam maior correlação positiva e as 10 variáveis que possuiam maior correlação negativa.

Foram gerados modelos para os 3 conjuntos de X: com as 190 variáveis, com as 138 variáveis selecionadas e com apenas as 20 variáveis mais correlacionadas com a média geométrica do crescimento. Os resultados estão na seção 6.

5.2.5 Aprendizado

Para gerar o aprendizado do modelo preditivo, foram utilizadas as ferramentas *H2O - AutoML* e *Catboost*.

Os dados utilizados fazem parte de uma amostra de 37.880 empresas do Brasil que possuíam 10 ou mais funcionários no ano de 2013. Os dados foram divididos em subconjuntos para treino e teste segundo a relação de 80% para treino e 20% para teste. No *Catboost* essa relação foi respeitada estritamente, gerando 30.304 dados para treino e 7.576 para teste. No *H2O*, os dados foram separados com aproximadamente 80.27% para treino, resultando em 30.408 dados para treino e 7.472 para teste. Porém, o *H2O* tem duas funcionalidades bem úteis, que acabam não utilizando tanto esses números. A primeira é a capacidade de realizar validação cruzada. Foi configurado para a ferramenta realizar validação cruzada com 5 conjuntos. Essa validação é útil e passa uma maior confiança para o resultado, pois ela gera 5 subconjuntos de toda a base, sendo um subconjunto de cada vez a amostra de teste e os outros 4 de treino. A segunda capacidade do *H2O* é a de balancear as classes. Como existem apenas 6% dos dados da amostra sendo 1 (empresas de alto crescimento), é necessário realizar um balanceamento nos dados. Dessa forma, o próprio *H2O* se encarrega de igualar a proporção de 0 e 1 na variável resposta.

Em relação aos parâmetros utilizados, na ferramenta *H2O* foram escolhidos os seguintes:

- $stopping\_metric = logloss$;
- $balance\_classes = True$;
- $sort\_metric = AUC$.

Já, para a ferramenta *Catboost*, os seguintes parâmetros foram escolhidos:

- $loss\_function = logloss$;
- $eval\_metric = F1$;
- $iterations = 700$;

- $depth = 5$;
- $learning\_rate = 0.05$;

5.2.6 Métricas e Comparações

Existem várias métricas na hora de avaliar um modelo de aprendizado de máquina. Porém, depende do problema qual serão as métricas mais observadas. No caso do presente projeto, deseja-se que o modelo seja capaz de acertar a maior parte das empresas de alto crescimento que ele indicar. Ou seja, a precisão desse modelo deve ser a métrica mais observada. Mas, além disso, é necessário também que o modelo tente acertar o maior número de empresas e não diga apenas para aquelas que tem certeza absoluta, sendo o número de 1 preditos baixo.

Assim, em ordem decrescente de importância, serão analisadas as seguintes métricas:

- Precisão
- F1-Score
- Recall
- Acurácia
- Cohen-Kappa
- Especificidade

Todas essas métricas foram geradas para modelos utilizando 2 ferramentas: (a) *H2O* e (b) *Catboost* e 4 subconjuntos de variáveis explanatórias (X): (a) X contendo as 190 variáveis geradas automaticamente; (b) X contendo apenas as variáveis que passaram pelo *Feature Selection*: 138; (c) X contendo as 20 variáveis mais correlacionadas com a média geométrica do crescimento das empresas nos 3 anos seguintes a 2013; e (d) X contendo apenas as variáveis criadas manualmente por um cientista de dados na *Neoway*. Dessa forma, será possível comparar tanto os resultados das variáveis geradas automaticamente com as variáveis geradas manualmente, como também os resultados gerados pelas diferentes ferramentas.

6 Resultados

Os principais resultados obtidos podem ser separados em alguns tópicos, já que o objetivo do trabalho não foi somente elaborar um modelo preditivo para o crescimento das empresas, mas sim gerar valor para a equipe de *Analytics* através do uso de uma ferramenta nova, capaz de gerar variáveis automaticamente, diminuindo o tempo do processo mais custoso de um cientista de dados. Dessa forma, serão apresentados os resultados para o modelo de classificação de empresas de alto crescimento (6.1), as variáveis interessantes geradas automaticamente (6.2) e o desempenho da metodologia utilizada (6.3).

6.1 Modelo de Classificação

Como explicado na seção 5.2.6, foram gerados resultados com duas ferramentas distintas de modelagem e com 4 subconjuntos de dados explanatórios. Os resultados do subconjunto com as variáveis geradas manualmente será exposto na seção seguinte (6.1.4). Nessa seção estarão os resultados gerados para os modelos que utilizaram os 3 subconjuntos: (a) X contendo as 190 variáveis geradas automaticamente (seção 6.1.1); (b) X contendo apenas as variáveis que passaram pelo *Feature Selection*: 138 (seção 6.1.2); e (c) X contendo as 20 variáveis mais correlacionadas com a média geométrica do crescimento das empresas nos 3 anos seguintes a 2013 (seção 6.1.3).

6.1.1 Variáveis Automáticas

Primeiramente, com o intuito de verificar o resultado dos modelos com todas as variáveis automáticas geradas, foram gerados 5 modelos na ferramenta *H2O*, utilizando um conjunto X de variáveis explanatórias com todas as 190 variáveis. Os resultados desses modelos são mostrados na Figura 21. Para o primeiro modelo da figura, que obteve maior AUC, as métricas em cima dos dados de VALIDAÇÃO constam em duas tabelas (Tabela 2 e Tabela 3). A primeira delas traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 2). A segunda (Tabela 3) traz as métricas calculadas no *threshold* que maximiza a Precisão. Na Figura 22 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza a Precisão.

O *H2O* também gera um relatório para os modelos com métricas a partir da técnica de validação cruzada. Dessa forma, outras duas tabelas foram geradas. A primeira traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 4). E, a segunda (Tabela 5) traz as métricas calculadas no *threshold* que maximiza a Precisão. Na Figura 23 consta a matriz de confusão gerada para o melhor modelo do

Figura 21 – Modelos gerados pelo *H2O* com todas as 190 variáveis geradas automaticamente.

model_id	auc	logloss	mean_per_class_error
StackedEnsemble_BestOfFamily_0_AutoML_20181119_194803	0.716897	0.237889	0.380849
StackedEnsemble_AllModels_0_AutoML_20181119_194803	0.716897	0.237889	0.380849
GLM_grid_0_AutoML_20181119_194803_model_0	0.716412	0.238101	0.369988
DRF_0_AutoML_20181119_194803	0.650579	0.412226	0.411247
XRT_0_AutoML_20181119_194803	0.603339	0.371377	0.432088

Tabela 2 – Métricas máximas, nos dados de VALIDAÇÃO, para o modelo com 190 variáveis em X.

Métricas	Valor
Precision	0.3230769
Accuracy	0.9307294
Recall	1.0000000
F1-score	0.2262540

Tabela 3 – Métricas para o *threshold* onde a Precisão é máxima, nos dados de VALIDAÇÃO. Modelo com 190 variáveis em X.

Métricas	Valor
Precision	0.3230769
Specificity	0.9921104
Accuracy	0.9270572
Recall	0.0507246
F1-score	0.0876826
Kappa	0.0702454

H20 no *threshold* que maximiza a Precisão.

Tabela 4 – Métricas máximas, em VALIDAÇÃO CRUZADA, para o modelo com 190 variáveis em X.

Métricas	Valor
Precision	0.3333333
Accuracy	0.9332432
Recall	1.0000000
F1-score	0.2250292

Além desses modelos gerados pelo *H2O*, ainda foram gerados outros modelos com a biblioteca *CatBoost*. O resultado do melhor modelo gerado com todas as variáveis criadas

Figura 22 – Matriz de confusão nos dados de VALIDAÇÃO para o modelo gerado a partir de 190 variáveis.

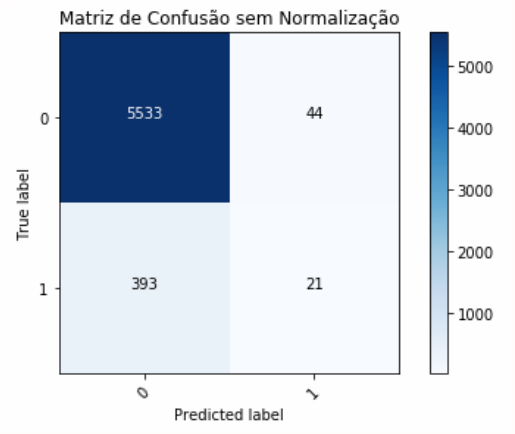
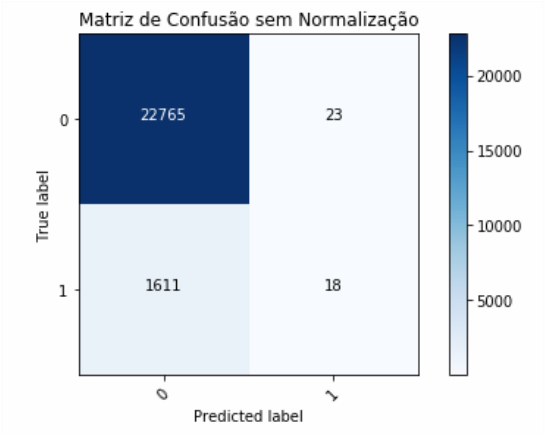


Tabela 5 – Métricas para o *threshold* onde a Precisão é máxima, em VALIDAÇÃO CRUZADA. Modelo com 190 variáveis em X.

Métricas	Valor
Precision	0.4390243
Specificity	0.9989906
Accuracy	0.9330794
Recall	0.0110497
F1-score	0.0215568
Kappa	0.0183411

Figura 23 – Matriz de confusão por VALIDAÇÃO CRUZADA para o modelo gerado a partir de 190 variáveis.



automaticamente está na Figura 24. As métricas desse modelo se encontram na Tabela 6.

Figura 24 – Resultado do modelo de 190 variáveis na ferramenta *Catboost*.

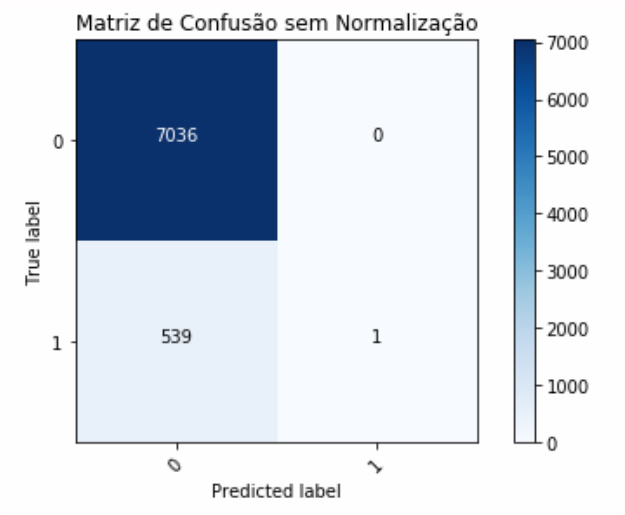


Tabela 6 – Métricas para o modelo com 190 variáveis explanatórias realizado no *Catboost*.

Métricas	Valor
Precision	1.00
Specificity	1.00
Accuracy	0.9288542
Recall	0.0018518
F1-score	0.0036968
Kappa	0.0034342

6.1.2 Variáveis Automáticas Selecionadas

Levando em consideração somente as variáveis que passaram pela etapa de *Feature Selection* (seção 5.2.4) e não foram rejeitadas (138 variáveis), foram gerados 7 modelos de classificação de empresas de alto crescimento com a ferramenta *H2O*. Os resultados desses modelos são mostrados na Figura 25.

Figura 25 – Modelos gerados pelo *H2O* apenas com 138 variáveis que passaram pelo *Feature Selection*.

model_id	auc	logloss	mean_per_class_error
StackedEnsemble_AllModels_0_AutoML_20181119_185227	0.715132	0.238504	0.373357
StackedEnsemble_BestOfFamily_0_AutoML_20181119_185227	0.715132	0.238504	0.373357
GLM_grid_0_AutoML_20181119_185227_model_0	0.713374	0.237977	0.359862
DRF_0_AutoML_20181119_185227	0.650881	0.405545	0.40037
GBM_grid_0_AutoML_20181119_185227_model_1	0.592498	0.258269	0.432131
GBM_grid_0_AutoML_20181119_185227_model_0	0.59166	0.293414	0.428235
XRT_0_AutoML_20181119_185227	0.576531	0.377838	0.439847

Para o melhor modelo da Figura 25, as métricas em cima dos dados de VALIDAÇÃO constam em duas tabelas (Tabela 7 e Tabela 8). A primeira delas traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 7). A segunda (Tabela 8) traz as métricas calculadas no *threshold* que maximiza a Precisão. Na Figura 26 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza a Precisão.

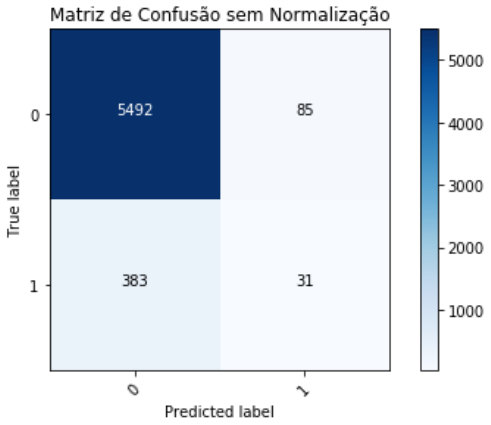
Tabela 7 – Métricas máximas, nos dados de VALIDAÇÃO, para o modelo com 138 variáveis em X.

Métricas	Valor
Precision	0.2672414
Accuracy	0.9307294
Recall	1.00
F1-score	0.2132937

Tabela 8 – Métricas para o *threshold* onde a Precisão é máxima, nos dados de VALIDAÇÃO. Modelo com 138 variáveis em X.

Métricas	Valor
Precision	0.2672413
Specificity	0.9847588
Accuracy	0.9218828
Recall	0.0748792
F1-score	0.1169811
Kappa	0.0894373

Figura 26 – Matriz de confusão nos dados de VALIDAÇÃO para o modelo gerado a partir de 138 variáveis.



O *H2O* também gera um relatório para os modelos com métricas a partir da técnica de validação cruzada. Dessa forma, outras duas tabelas foram geradas. A primeira traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 9). E, a segunda (Tabela 10) traz as métricas calculadas no *threshold* que maximiza a Precisão. Na Figura 27 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza a Precisão.

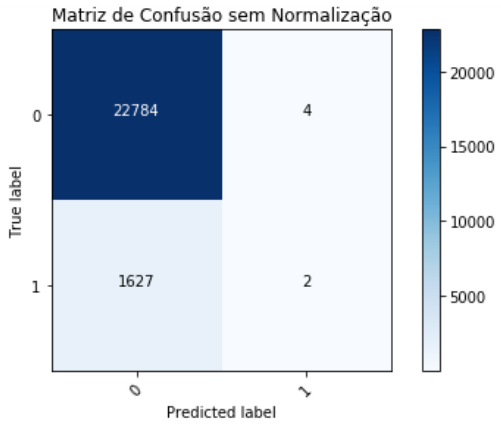
Tabela 9 – Métricas máximas, em VALIDAÇÃO CRUZADA, para o modelo com 138 variáveis em X.

Métricas	Valor
Precision	0.3333333
Accuracy	0.9332432
Recall	1.00
F1-score	0.2216826

Tabela 10 – Métricas para o *threshold* onde a Precisão é máxima, em VALIDAÇÃO CRUZADA. Modelo com 138 variáveis em X.

Métricas	Valor
Precision	0.3333333
Specificity	0.9998244
Accuracy	0.9332022
Recall	0.0012277
F1-score	0.0024464
Kappa	0.0019577

Figura 27 – Matriz de confusão por VALIDAÇÃO CRUZADA para o modelo gerado a partir de 138 variáveis.



Além desses modelos gerados pelo *H2O*, ainda foram gerados outros modelos com a biblioteca *CatBoost*. O resultado do melhor modelo gerado com as 138 variáveis criadas automaticamente está na Figura 28. As métricas desse modelo se encontram na Tabela 11.

Figura 28 – Resultado do modelo de 138 variáveis na ferramenta *Catboost*.

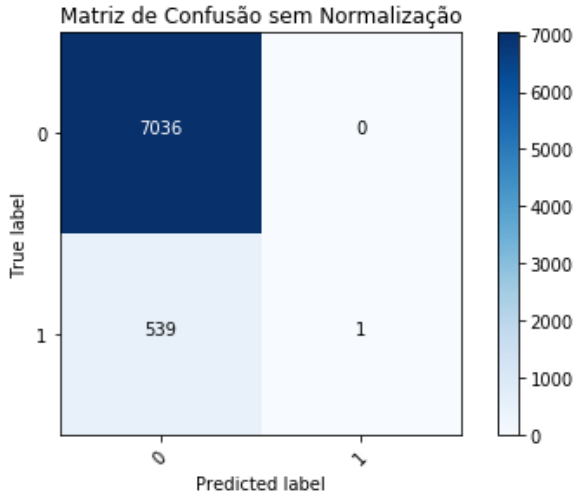


Tabela 11 – Métricas para o modelo com 138 variáveis explanatórias realizado no *Catboost*.

Métricas	Valor
Precision	1.00
Specificity	1.00
Accuracy	0.9288542
Recall	0.0018518
F1-score	0.0036968
Kappa	0.0034342

6.1.3 Variáveis Automáticas Correlacionadas

Para testar os modelos em um conjunto de variáveis menor ainda que 138, outra técnica de seleção foi utilizada. Foram selecionadas as 10 variáveis mais se correlacionavam positivamente com o crescimento médio e as 10 que mais se correlacionavam negativamente. Assim, com apenas 20 variáveis no conjunto X, os resultados desses modelos utilizando a ferramenta *H2O* são os mostrados na Figura 29.

Para o melhor modelo da Figura 29, as métricas em cima dos dados de VALIDAÇÃO constam em duas tabelas (Tabela 12 e Tabela 13). A primeira delas traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 12). A segunda (Tabela 13) traz as métricas calculadas no *threshold* que maximiza a Precisão. Na

Figura 29 – Modelos gerados pelo *H2O* apenas com 20 variáveis que se correlacionam com o crescimento médio das empresas.

model_id	auc	logloss	mean_per_class_error
DRF_0_AutoML_20181119_173327	0.651308	0.321024	0.408006
GLM_grid_0_AutoML_20181119_173327_model_0	0.653563	0.247329	0.410445
XRT_0_AutoML_20181119_173327	0.658983	0.3202	0.415606
GBM_grid_0_AutoML_20181119_173327_model_3	0.649911	0.26115	0.398851
GBM_grid_0_AutoML_20181119_173327_model_2	0.667648	0.247245	0.395882
GBM_grid_0_AutoML_20181119_173327_model_1	0.677133	0.24327	0.401725
StackedEnsemble_BestOfFamily_0_AutoML_20181119_173327	0.679669	0.240476	0.387747
GBM_grid_0_AutoML_20181119_173327_model_0	0.689249	0.239991	0.387068
GBM_grid_0_AutoML_20181119_173327_model_4	0.69037	0.247518	0.382854
StackedEnsemble_AllModels_0_AutoML_20181119_173327	0.700127	0.235276	0.377704

Figura 30 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza o a precisão.

Tabela 12 – Métricas máximas, nos dados de VALIDAÇÃO, para o modelo com 20 variáveis em X.

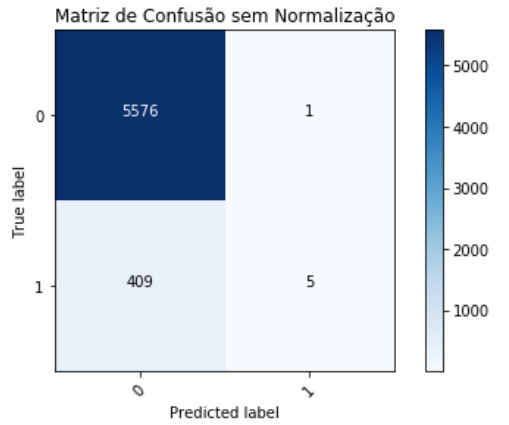
Métricas	Valor
Precision	0.8333333
Accuracy	0.9315640
Recall	1.00
F1-score	0.2448980

Tabela 13 – Métricas para o *threshold* onde a Precisão é máxima, nos dados de VALIDAÇÃO. Modelo com 20 variáveis em X.

Métricas	Valor
Precision	0.8333333
Specificity	0.9998206
Accuracy	0.9315640
Recall	0.0120772
F1-score	0.0238095
Kappa	0.0218783

O *H2O* também gera um relatório para os modelos com métricas a partir da técnica de validação cruzada. Dessa forma, outras duas tabelas foram geradas. A primeira traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 14). E, a segunda (Tabela 15) traz as métricas calculadas no *threshold* que maximiza

Figura 30 – Matriz de confusão nos dados de VALIDAÇÃO para o modelo gerado a partir de 20 variáveis.



a Precisão. Na Figura 31 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza a Precisão.

Tabela 14 – Métricas máximas, em VALIDAÇÃO CRUZADA, para o modelo com 20 variáveis em X.

Métricas	Valor
Precision	0.4390244
Accuracy	0.9332432
Recall	1.00
F1-score	0.2386282

Tabela 15 – Métricas para o *threshold* onde a Precisão é máxima, em VALIDAÇÃO CRUZADA. Modelo com 20 variáveis em X.

Métricas	Valor
Precision	0.4390243
Specificity	0.9989906
Accuracy	0.9330794
Recall	0.0110497
F1-score	0.0215568
Kappa	0.0183411

Além desses modelos gerados pelo *H2O*, ainda foram gerados outros modelos com a biblioteca *CatBoost*. O resultado do melhor modelo gerado com as 20 variáveis criadas automaticamente está na Figura 32. As métricas desse modelo se encontram na tabela Tabela 16.

Figura 31 – Matriz de confusão por VALIDAÇÃO CRUZADA para o modelo gerado a partir de 20 variáveis.

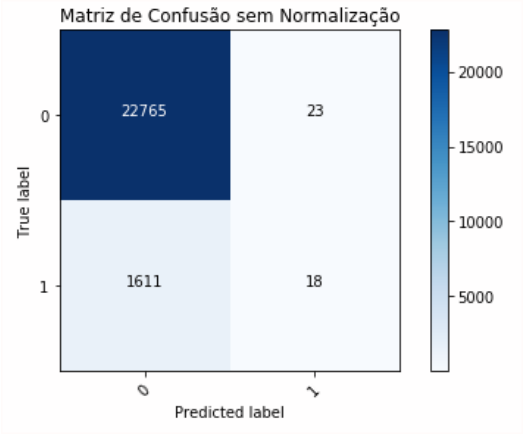


Figura 32 – Resultado do modelo de 20 variáveis na ferramenta *Catboost*.

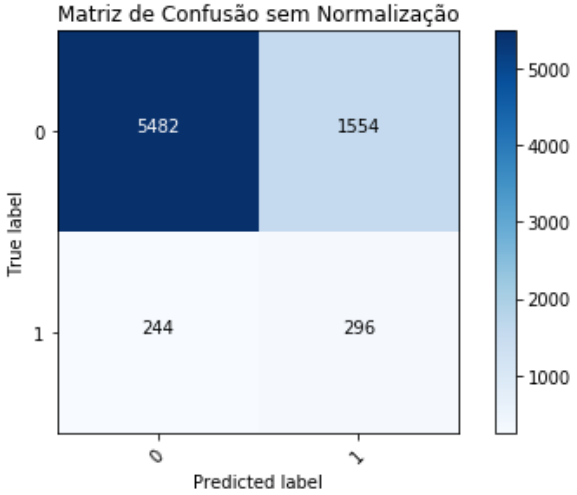


Tabela 16 – Métricas para o modelo com 20 variáveis explanatórias realizado no *Catboost*.

Métricas	Valor
Precision	0.16
Specificity	0.7791358
Accuracy	0.7626715
Recall	0.5481481
F1-score	0.2476987
Kappa	0.1543887

6.1.4 Variáveis Manuais

Para os resultados mostrados na seção acima terem uma base de comparação, foram gerados modelos utilizando somente as variáveis criadas manualmente por um cientista de

dados na *Neoway*. Dessa forma, a comparação entre ambos pode ser discutida na seção de Conclusão. Os modelos gerados com a mesma amostra de dados e mesma variável resposta, com a ferramenta *H20* estão na Figura 33.

Figura 33 – Modelos gerados pelo *H2O* com 12 variáveis geradas manualmente.

model_id	auc	logloss	mean_per_class_error
GBM_grid_0_AutoML_20181119_181319_model_0	0.852366	0.19629	0.285313
StackedEnsemble_BestOfFamily_0_AutoML_20181119_181319	0.850283	0.189409	0.283727
StackedEnsemble_AllModels_0_AutoML_20181119_181319	0.849191	0.187444	0.272005
GBM_grid_0_AutoML_20181119_181319_model_1	0.843286	0.204825	0.276219
GBM_grid_0_AutoML_20181119_181319_model_2	0.842365	0.21327	0.268942
GBM_grid_0_AutoML_20181119_181319_model_3	0.829194	0.23269	0.321844
GBM_grid_0_AutoML_20181119_181319_model_4	0.790075	0.224511	0.33197
XRT_0_AutoML_20181119_181319	0.73618	0.302115	0.350931
DRF_0_AutoML_20181119_181319	0.734428	0.328217	0.383155
GLM_grid_0_AutoML_20181119_181319_model_0	0.717228	0.239124	0.380471

Para o melhor modelo da Figura 33, as métricas em cima dos dados de VALIDAÇÃO constam em duas tabelas (Tabela 17 e Tabela 18). A primeira delas traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 17). A segunda (Tabela 18) traz as métricas calculadas no *threshold* que maximiza a Precisão. Na Figura 34 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza a Precisão.

Tabela 17 – Métricas máximas, nos dados de VALIDAÇÃO, para o modelo com 12 variáveis manuais em X.

Métricas	Valor
Precision	0.9090909
Accuracy	0.9399099
Recall	1.00
F1-score	0.4585492

O *H2O* também gera um relatório para os modelos com métricas a partir da técnica de validação cruzada. Dessa forma, outras duas tabelas foram geradas. A primeira traz as métricas máximas que podem ser obtidas nesse modelo, dependendo do *threshold* (Tabela 19). E, a segunda (Tabela 20) traz as métricas calculadas no *threshold* que maximiza o *F1-Score*. Na Figura 35 consta a matriz de confusão gerada para o melhor modelo do *H20* no *threshold* que maximiza a Precisão.

Além desses modelos gerados pelo *H2O*, ainda foram gerados outros modelos com a biblioteca *CatBoost*. O resultado do melhor modelo gerado com as 12 variáveis criadas

Tabela 18 – Métricas para o *threshold* onde a Precisão é máxima, nos dados de VALIDAÇÃO. Modelo com 12 variáveis manuais em X.

Métricas	Valor
Precision	0.9090909
Specificity	0.9998206
Accuracy	0.9323985
Recall	0.0241545
F1-score	0.0470588
Kappa	0.0436377

Figura 34 – Matriz de confusão nos dados de VALIDAÇÃO para o modelo gerado a partir de 12 variáveis manuais.

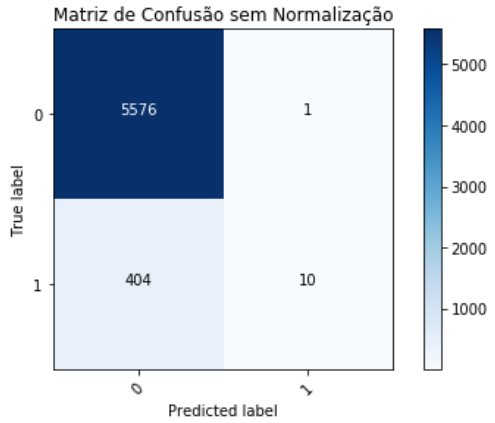


Tabela 19 – Métricas máximas, em VALIDAÇÃO CRUZADA, para o modelo com 12 variáveis manuais em X.

Métricas	Valor
Precision	1.00
Accuracy	0.9415981
Recall	1.00
F1-score	0.46875

manualmente está na Figura 36. As métricas desse modelo se encontram na Tabela 21.

Vale salientar que para essas 12 variáveis utilizadas, o cientista de dados criou cerca de 100 variáveis manualmente e depois aplicou técnicas de seleção das mais importantes e úteis. Então, o tempo gasto foi relativo a todo o processo.

Em relação aos custos da abordagem manual, podem ser levados em conta o tempo gasto na preparação das variáveis, o conhecimento sobre o domínio do ser humano que está desenvolvendo a engenharia de variáveis, o conhecimento sobre outras ferramentas

Tabela 20 – Métricas para o *threshold* onde a Precisão é máxima, em VALIDAÇÃO CRUZADA. Modelo com 12 variáveis manuais em X.

Métricas	Valor
Precision	1.00
Specificity	1.00
Accuracy	0.9343080
Recall	0.0153468
F1-score	0.0302297
Kappa	0.0282699

Figura 35 – Matriz de confusão por VALIDAÇÃO CRUZADA para o modelo gerado a partir de 12 variáveis manuais.

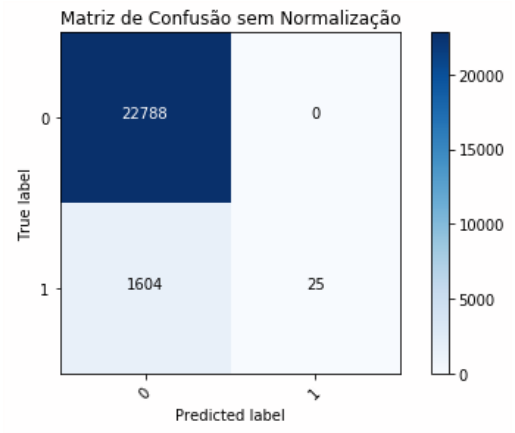
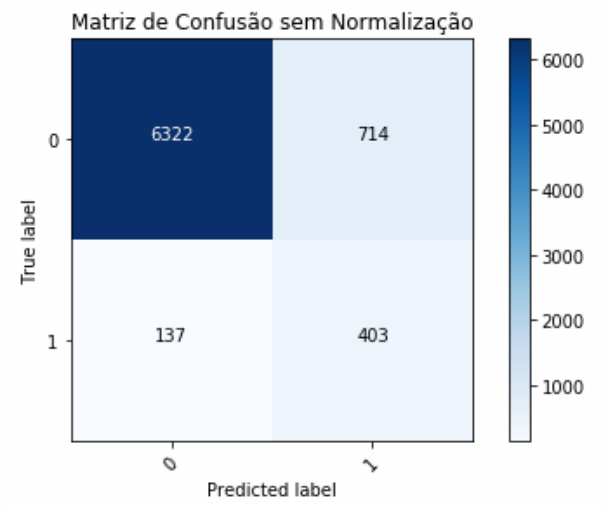


Figura 36 – Resultado do modelo de 12 variáveis manuais na ferramenta *Catboost*.



capazes de executar a geração dessas variáveis e o tempo gasto nessas ferramentas. Dessa forma, os custos estimados nesse processo manual podem ser visualizados nos itens abaixo:

Tabela 21 – Métricas para o modelo com 12 variáveis manuais explanatórias realizado no *Catboost*.

Métricas	Valor
Precision	0.3607878
Specificity	0.8985218
Accuracy	0.8876715
Recall	0.7462962
F1-score	0.4864212
Kappa	0.4318202

- **Conhecimento sobre o domínio necessário**: Altíssimo. Quem gerou as variáveis foi um cientista de dados mestre em Economia, com experiência de mais de 7 anos na indústria;
- **Tempo gasto com o aprendizado da nova ferramenta**: Não se aplica;
- **Tempo gasto com a estruturação dos dados**: Não se aplica especificamente, porém é incluído de forma geral no último item;
- **Tempo gasto na preparação e seleção das variáveis até chegar em um conjunto menor que performaram bem**: 3 meses;
- **Conhecimento necessário de outras ferramentas para manipulação de dados?** Sim. Nesse caso, foram utilizadas manipulações em *SQL* com *PostgreSQL* e as bibliotecas da ferramenta *R*.

6.1.5 Resumo Modelos

Em resumo, pode-se visualizar a Tabela 22 para as métricas máximas obtidas em todos os conjuntos de *features*, levando em consideração uma validação cruzada nos dados. Esses resultados serão analisados no capítulo de conclusão do projeto.

Tabela 22 – Métricas MÁXIMAS utilizando VALIDAÇÃO CRUZADA.

Conjunto	Precisão	*F1_Score*	Recall	Acurácia
A	0.333	0.225	1.000	0.933
B	0.333	0.222	1.000	0.933
C	0.439	0.239	1.000	0.933
Manual	1.000	0.469	1.000	0.942

Além das métricas máximas, podem ser visualizados os números para as métricas quando a precisão é máxima. Esses valores estão na Tabela 23. A análise será realizada no capitulo de conclusão do projeto.

Tabela 23 – Métricas para PRECISÃO MÁXIMA utilizando VALIDAÇÃO CRUZADA.

Conjunto	Precisão	*F1_Score*	Recall	Acurácia
A	0.333	0.022	0.011	0.933
B	0.333	0.002	0.001	0.933
C	0.439	0.022	0.011	0.933
Manual	1.000	0.030	0.015	0.934

6.2 *Features* Interessantes Geradas

Além dos resultados do modelo preditivo, outro resultado interessante para a empresa é acerca das variáveis que a ferramenta *Featuretools* conseguiu gerar em cima dos dados relacionais passados à ela. Dessa forma, dentre as 190 variáveis geradas automaticamente, pode-se citar que as variáveis da lista a seguir são de muita valia para a empresa, que já desenvolveu algumas dessas de forma manual e despendeu muito tempo.

- Quantidade de contratações + demissões realizadas até certo momento no tempo;
- Quantidade de pessoas no quadro societário da empresa;
- Qual o porte mais frequente das empresas da mesma região;
- Quantidade de contratações + demissões de todas as empresas de mesma natureza jurídica;
- Número de contratações + demissões na mesma região;
- Número de empresas na mesma região;
- Soma do número de sócios das empresas da mesma Classificação Nacional de Atividades Econômicas(CNAE);
- Quantidade de contratações + demissões das empresas pertencentes à mesma CNAE;
- Número de admissões e demissões;
- Quantidade de empresas na mesma CNAE;
- O porte mais frequente das empresas da mesma natureza jurídica;
- Crescimentos do número de funcionários e demissões e contratações por região, CNAE e natureza jurídica;
- Desvio padrão do número de contratações + admissões de uma empresa durante 3 anos.

6.3 Desempenho da Metodologia

A metodologia utilizada, que está totalmente atrelada à ferramenta *Featuretools*, possui alguns custos. Os custos visualizados nesse projeto são em relação ao tempo de aprendizado da ferramenta, ao tempo de processamento das variáveis e ao tempo gasto na estruturação dos dados.

A ferramenta, por estar em uma linguagem de pequena curva de aprendizado: *Python*, possui uma média de tempo baixa para o aprendizado. O autor do projeto pode aprender todas as funcionalidades da ferramenta em aproximadamente 10 dias. Esse tempo inclui a geração de algumas variáveis automaticamente. Levando em consideração que esse tempo inicial é despendido apenas uma vez, ele não volta a ser contabilizado quando a mesma pessoa for utilizar a ferramenta novamente.

O processamento das variáveis é um problema maior, pois conforme aumenta o número de variáveis primitivas e também a profundidade desejada, a ferramenta despende mais tempo processando. Para ter uma base, a ferramenta demorou cerca de 3 horas calculando 95 variáveis para 15 entidades, onde apenas 1 delas continha 13.3 milhões de registros, a tabela principal de empresas continha 37880 registros, a profundidade era igual a 2 e haviam 6 primitivas. Já, em outro experimento, com as mesmas 15 entidades, onde a maior delas continha apenas 2.5 milhões de registros, a entidade principal com 45.299 empresas e as mesmas 6 primitivas, a ferramenta demorou cerca de 33 minutos para calcular 165 variáveis. Ou seja, se há uma entidade muito requisitada que possui milhões de registros, a geração de variáveis pode ficar bem lenta.

Outro resultado acerca do tempo gasto tem relação com a estruturação dos dados. Se as tabelas que forem utilizadas como entrada do algoritmo não estiverem em um formato relacional, todas deverão se adequar de forma que possuam chaves primárias, estrangeiras, índices e relações. Também, algumas tabelas deverão ser quebradas em 2 ou mais, mas a ferramenta possibilita que essa quebra aconteça já dentro do *Featuretools*. Nesse projeto, essa etapa de estruturação e entendimento dos dados que seriam utilizados pela ferramenta demorou cerca de 5 dias.

Ao todo, foram despendidos aproximadamente 15 dias e algumas horas de trabalho para aprender a ferramenta, entender os dados, estruturar os dados, selecionar amostras, processar as variáveis, selecionar as variáveis, rodar modelos e avaliar os modelos gerados.

Com isso, para comparação com os itens da seção 6.1.4, podemos listar os mesmos pontos acerca da metodologia com a ferramenta *Featuretools*:

- **Conhecimento sobre o domínio necessário**: Baixo-Médio. Para gerar as variáveis, basta estruturar os dados e inserir na ferramenta. Os conhecimentos de negócio necessários são para escolher quais primitivas utilizar e como gerar a variável resposta,

se necessário um cálculo específico;

- **Tempo gasto com o aprendizado da nova ferramenta**: 10 dias;
- **Tempo gasto com a estruturação dos dados**: 5 dias;
- **Tempo gasto na preparação e seleção das variáveis até chegar em um conjunto menor que performaram bem**: Algumas horas, depende do número de dados de entrada e a quantidade de variáveis finais;
- **Conhecimento necessário de outras ferramentas para manipulação de dados?** Não. Apenas o *Python* e o *Featuretools* resolvem.

7 Conclusões e Perspectivas

Após os resultados obtidos no capítulo anterior e a experiência adquirida ao longo do projeto, algumas conclusões foram tiradas e expostas nesse capítulo. Estruturalmente, o capítulo de conclusão estará disposto com as respostas aos objetivos na seção 7.1, os problemas encontrados na seção 7.2, as possíveis extensões na seção 7.3 e as recomendações para trabalhos futuros na seção 7.4.

7.1 Respostas aos Objetivos

Como definido na seção 1.4, os objetivos gerais foram divididos em 2. Assim, cada objetivo terá uma seção específica da conclusão.

7.1.1 Geração Automática de Variáveis

Em relação ao primeiro objetivo geral, que diz respeito à validação da abordagem de geração automática de variáveis para economizar tempo humano, gasto no processo de engenharia de variáveis, pode ser relatado que muitas variáveis interessantes foram geradas, a partir de 15 entidades diferentes. Algumas dessas variáveis foram expostas na seção 6.2 e a partir delas, podem ser criados novos modelos. Esses modelos podem usufruir das variáveis geradas automaticamente tanto para explicar algo, quanto para servir de variável resposta. Dessa forma, a produção de um novo modelo de aprendizado de máquina pode ser realizado em um tempo muito inferior ao que vem sendo realizado hoje, já que muito tempo é gasto na etapa de geração de variáveis. Esse tempo é gasto pois depende de um conhecimento muito grande do domínio e também da geração de códigos para agregar e transformar bases de dados.

Ficou claro também que existem dois pontos fracos na geração de variáveis de maneira automática. O primeiro ponto vem a existir se a base de dados não estiver em um formato totalmente relacional. Dessa forma, os dados deverão ser reestruturados e isso pode levar tempo. O segundo ponto diz respeito à quantidade de dados utilizados para a geração automática. Como mostrado na seção 6.3, para gerar a mesma quantidade de variáveis, com os mesmos parâmetros, alterando somente uma entidade que passou de 2.5 milhões de registros para 13.3 milhões, foram despendidos 6x a mais de tempo no processamento. Nesse caso, uma alternativa seria gerar uma quantidade grande de variáveis para uma amostra inicial, executar uma fase de *Feature Selection* e rodar um modelo de aprendizado de máquina. Só então com base nas melhores variáveis, a execução de toda a base seria realizada.

Em relação ao trabalho manual, a abordagem automatizada se mostrou satisfatória quando comparada com o tempo gasto manualmente, ao número de variáveis geradas e ao conhecimento de domínio necessário. Com isso, a abordagem automatizada pode gerar impactos profundos dentro da empresa, que passará a criar inúmeras variáveis a partir da imensa base de dados disponível. Essas variáveis poderão ser utilizadas tanto para servir de explicação para novos modelos, quanto para serem a própria variável resposta. Se forem a variável resposta, inúmeros novos modelos poderão ser criados em questão de dias, não mais meses. No mínimo, a geração automática de variáveis pode reduzir o tempo da engenharia de *features* e auxiliar na quantidade de dados a serem utilizados para explicar outras variáveis dependentes.

As variáveis mostraram ter qualidade e poder preditivo igual às geradas manualmente. Isso será discutido mais a fundo na próxima seção, que trata do modelo preditivo. Porém, a escalabilidade da abordagem ainda é uma questão a ser estudada. Como informado anteriormente, quando o volume de dados cresce, o tempo de execução também cresce. Essa relação depende das primitivas que estarão sendo utilizadas também. Uma forma de combater esse gargalo é dividir os dados utilizados em mini conjuntos. Cada conjunto conteria os dados suficientes para certa quantidade de instâncias da entidade alvo, no caso desse projeto uma quantia de empresas. Dessa forma, seriam geradas matrizes de variáveis para cada número limitado de empresas, deixando o processamento ser aplicado paralelamente. Com a aplicação paralela ou distribuída, em ferramentas como *Spark* e *Dask*, pode-se escalar a solução, mas esse passo inicial de separação dos dados em conjuntos ainda é necessário.

Quanto às ferramentas de geração automática de variáveis, foi concluído que essa abordagem é recente e existem várias teorias promissoras sendo desenvolvidas. A maioria delas, que foram desenvolvidas até o momento desse estudo, não possuem ainda uma ferramenta capaz de traduzir a teoria em código, portanto não foram utilizadas. Dentre as ferramentas que possuem algoritmos implementados e *Open Source*, as que mais se destacam são o *Featuretools* e a *TransmogrifAI*. A ferramenta *TransmogrifAI* não foi escolhida por utilizar uma teoria não publicada, utilizar apenas os tipos de dados para gerar novas variáveis e estar desenvolvida em uma linguagem de curva de aprendizado maior: *Scala*. Em contraste, a ferramenta utilizada *Featuretools* se mostrou ser de muito fácil aprendizado, aplicar uma teoria muito bem definida e que leva em consideração o relacionamento entre os dados - atividade essa que acontece em um trabalho manual de engenharia de *features* - e ter muitos contribuidores ativos que auxiliam na aplicação da mesma.

A ferramenta *Featuretools* basicamente necessita de uma entrada de dados relacionais, sendo explicitamente dito quais entidades serão utilizadas, quais serão criadas a partir de uma normalização de outra coluna e quais os relacionamentos entre as chaves.

Logo, a ferramenta por si só não é capaz de aprender o relacionamento entre os dados.

7.1.2 Modelo Preditivo

Com relação ao modelo preditivo, foram criados alguns modelos a partir das variáveis automáticas para comparação com outro modelo criado a partir de variáveis manuais. Dos resultados apresentados na seção 6.1, pode-se dizer que os modelos gerados a partir de variáveis puramente automáticas tiveram resultados semelhantes ao modelo manual, porém não superiores.

Mais profundamente, em relação ao modelo de 190 variáveis, seu resultado não conseguiu superar o modelo manual, pois a acurácia ficou muito próxima, mas abaixo, e o máximo *F1-Score* é apenas a metade do *F1-Score* manual máximo. Essa diferença pode ser devido à quantidade de variáveis utilizadas em cima de apenas uma amostra de 37.880 empresas. Também, muitas dessas variáveis possuem correlações entre si e podem estar confundindo os algoritmos de aprendizado com informações sujas.

Comparando somente o modelo de 190 variáveis nas duas ferramentas (*H2O* e *Catboost*) nota-se que a segunda chegou em melhores resultados na base de validação, para a métrica de precisão. Porém, apenas 1 registro foi predito como 1, enquanto existiam 540 para serem classificados como 1. Isso é, o *Recall* foi baixíssimo.

O modelo que utilizou as variáveis selecionadas pela etapa de *Feature Selection* chegou em resultados muito parecidos ao anterior. Assim, o número elevado de variáveis pode ainda estar interferindo nesses resultados.

O último modelo, que contou com apenas 20 variáveis, obteve acurácia bem parecida com o manual, porém ainda deixou a desejar no *F1-Score*. Ele é até capaz de atingir uma precisão alta, mas quando isso ocorre, seu *Recall* cai.

Apesar do modelo manual ter uma alta acurácia e um *F1-Score* superior aos obtidos com os conjuntos automáticos, o seu *F1-Score* não foi tão alto assim (máximo de 0.46875). Então, é normal que para esses problemas de crescimento, o *F1-Score* esperado não seja alto. E, apesar de a precisão máxima ter sido alta, nota-se que poucos registros são classificados como 1, então o *Recall* continua baixo.

Em linhas gerais, as variáveis geradas automaticamente tiveram um poder preditivo muito satisfatório, pois os resultados do modelos com variáveis automáticas chegou muito próximo aos resultados do modelo manual. Em relação ao poder computacional exigido, quanto maior a profundidade definida e o número de registros utilizados, mais processamento necessitará para a ferramenta gerar suas variáveis. Porém, na criação manual de variáveis, também existirá um tempo relacionado ao processamento computacional, já que as variáveis serão geradas com código. A diferença é que o método automático certamente construirá muito mais variáveis no mesmo tempo. Então, no momento de decidir utilizar

uma ferramenta automática, tem de se pesar se é válido construir uma base com muitas variáveis de forma automática para auxiliar no aprendizado do modelo, ou apenas algumas variáveis criadas manualmente já supre a necessidade.

Quanto ao tempo de desenvolvimento propriamente dito, desconsiderando o processamento, a etapa manual consome um tempo consideravelmente superior ao método automático. Um conhecimento específico sobre o domínio será requisitado pelo humano, além de dias de trabalho processando dados e pensando em novas formas de traduzir seu conhecimento. Enquanto isso, a ferramenta automática apenas requer uma base relacional e a definição de algumas funções primitivas para gerar inúmeras variáveis.

7.2 Problemas Encontrados

O primeiro problema encontrado enquanto lidando com a ferramenta *Featuretools* foi em questão a criação de variáveis que comparassem a mesma coluna em diferentes momentos do tempo. Essa funcionalidade não existe até a versão utilizada no projeto (0.4). Então, para contornar esse problema foi necessário o estudo mais aprofundado da ferramenta para a criação de variáveis personalizadas que servissem de primitivas ao mesmo tempo. Com a criação dessa variável personalizada, foi possível criar variáveis de crescimento, por exemplo, em cima de outro conjunto de entidade. Ou seja, com essa abordagem, duas gerações de variáveis automáticas foram necessárias.

O segundo problema encontrado tem relação ao grande número de dados disponíveis para a execução do modelo. Essa quantidade pesou na hora da geração de variáveis pelo *Featuretools*. A ferramenta possui uma funcionalidade de paralelizar o trabalho, porém ainda muita memória é consumida, já que todas as entidades são replicadas para os processadores. Assim, com 13.3 milhões de registros em uma tabela, 3 horas foram gastas e apenas 95 das 165 variáveis previstas foram geradas. Esses 13.3 milhões foram apenas para uma amostra de 37.880 empresas. Então, ficou complicado para gerar um modelo para a base toda de empresas com mais de 10 funcionários. Para contornar esse problema, duas abordagens poderiam ser levadas em consideração: (a) aplicar apenas as variáveis mais importantes para a base toda; (b) dividir a base em pequenos subconjuntos que contenham dados relativos apenas a uma parcela de empresas. Porém, essas abordagens serão deixadas como sugestões de trabalhos futuros.

7.3 Extensões Possíveis

A abordagem de utilizar uma ferramenta que gere variáveis automaticamente tem muito potencial, tanto dentro da empresa que o projeto foi realizado, quanto em outros locais que utilizem ciência de dados para solucionar problemas. Isso é devido à capacidade

de economizar tempo de engenharia na criação de variáveis independentes e também à capacidade de gerar variáveis respostas de forma rápida. Com a geração de variáveis respostas, modelos podem ser criados em dias, provendo uma extensibilidade enorme da abordagem. Nesse caso de crescimentos de empresas, tanto modelos para crescimento de região, quanto crescimento do ramo das empresas seriam possíveis rapidamente.

Além disso, um motor de criação de variáveis poderia ser realizado utilizando abordagens automáticas como base. Dessa forma, o número de variáveis disponíveis para serem utilizadas em outros modelos poderia crescer indefinidamente, tanto com o crescimento das bases de dados, quanto sem isso.

Automatizando a geração de variáveis explanatórias e variáveis resposta, bastaria a seleção das melhores e a aplicação dos algoritmos de aprendizado de máquina. Como já dito nesse documento, a automatização de todo o *pipeline* está sendo muito estudado ultimamente e com essa etapa a mais, facilmente todo o *pipeline* será automatizado.

7.4 Trabalhos Futuros

Para dar continuidade a esse projeto, especificamente, alguns trabalhos poderiam ser realizados. Dentre eles podem ser citados:

- Aplicação de toda a base de empresas brasileiras com mais de 10 funcionários no *Featuretools*;
- Geração de variáveis com diferentes parâmetros no *Featuretools*, como por exemplo profundidades maiores e diferentes primitivas;
- Criação de um modelo que preveja o crescimento em formato numérico (modelo de regressão);
- Criação de um modelo que siga outra definição de empresa de alto crescimento, como por exemplo a definição que seleciona apenas aquelas empresas que crescem no mínimo 20% cada ano, por 3 anos consecutivos;
- Utilizar a abordagem de separação dos dados para conjuntos menores de empresas, de modo que o processo possa ser distribuído, utilizando *Dask* ou *Scala*;
- Utilizar a abordagem de gerar muitas variáveis para uma amostra pequena da população, selecionar as melhores com técnicas de *Feature Selection* e depois aplicar somente essas variáveis na população.

Além desses, a criação de um motor que gere variáveis continuamente e a criação de modelos a partir de outras variáveis respostas (criadas automaticamente) seriam projetos interessantes de serem elaborados.

Referências

1 SCHUTT, R.; O'NEIL, C. *Doing Data Science: Straight Talk from the Frontline*. [S.l.]: O'Reilly Media, Inc., 2013. ISBN 1449358659, 9781449358655. Citado 2 vezes nas páginas 19 e 25.

2 WEDGE, R. et al. Solving the "false positives"problem in fraud prediction. *CoRR*, abs/1710.07709, 2017. Disponível em: <http://arxiv.org/abs/1710.07709>. Citado 3 vezes nas páginas 19, 30 e 49.

3 FRANZ, A.; MILCH, B. Searching the web by voice. In: *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*. [s.n.], 2002. p. 1213–1217. Disponível em: <https://research.google.com/archive/webbyvoice.html>. Citado na página 19.

4 LINDEN, G.; SMITH, B.; YORK, J. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, v. 7, p. 76–80, 01 2003. ISSN 1089-7801. Disponível em: <doi.ieeecomputersociety.org/10.1109/MIC.2003.1167344>. Citado na página 19.

5 Gauci, J. et al. Horizon: Facebook's Open Source Applied Reinforcement Learning Platform. *ArXiv e-prints*, nov. 2018. Citado na página 19.

6 Sadekar, K.; Jiang, H. Time Travel based Feature Generation. 2018. Citado na página 19.

7 Zhu, L.; Laptev, N. Deep and Confident Prediction for Time Series at Uber. *ArXiv e-prints*, set. 2017. Citado na página 19.

8 Banerjee, S. et al. Towards Wide Learning: Experiments in Healthcare. *ArXiv e-prints*, dez. 2016. Citado na página 19.

9 SARKAR, D.; BALI, R.; SHARMA, T. *Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems*. Apress, 2017. ISBN 9781484232071. Disponível em: <https://books.google.com.br/books?id=9ClEDwAAQBAJ>. Citado na página 19.

10 Hofmann, M.; Neukart, F.; Bäck, T. Artificial Intelligence and Data Science in the Automotive Industry. *ArXiv e-prints*, set. 2017. Citado na página 25.

11 FACELI, K. et al. *Inteligência artificial: uma abordagem de aprendizado de máquina*. [S.l.]: LTC, 2011. Citado na página 26.

12 CHAPMAN, P.; CLINTON, J.; AL, R. K. et. The crisp-dm user guide. In: . [S.l.: s.n.], 1999. Citado na página 26.

13 SHEARER, C. The crisp-dm model: the new blueprint for data mining. In: *Journal of Data Warehousing*. [S.l.: s.n.], 2000. v. 5, p. 13–22. Citado na página 26.

14 AMORIM, T. *Conceitos, técnicas, ferramentas e aplicações de Mineração de Dados para gerar conhecimento a partir de bases de dados*. [S.l.]: Universidade Federal de Pernambuco, 2007. Citado na página 27.

15 DOMINGOS, P. A few useful things to know about machine learning. *Communications of the ACM*, p. 55(10):78–87, 2012. Citado na página 27.

16 BANERJEE, S. et al. Towards wide learning: Experiments in healthcare. *30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain*, 2016. Citado 2 vezes nas páginas 27 e 28.

17 ZHENG, A.; CASARI, A. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. [S.l.]: O'REILLY, 2018. Citado na página 28.

18 DONG, G.; LIU, H. *Feature Engineering for Machine Learning and Data Analytics*. 1st. ed. [S.l.]: Chapman and Hall/CRC Press, 2018. Citado na página 29.

19 KANTER, J. M.; VEERAMACHANENI, K. Deep feature synthesis: Towards automating data science endeavors. *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, p. 1–10, 2015. Citado 4 vezes nas páginas 30, 31, 32 e 45.

20 GUJARATI, D. *Econometria Básica*. [S.l.]: Pearson, 2006. 529-541 p. Citado na página 32.

21 Balaji, A.; Allen, A. Benchmarking Automatic Machine Learning Frameworks. *ArXiv e-prints*, ago. 2018. Citado na página 33.

22 COAD, A. et al. High-growth firms: introduction to the special section. *Industrial and Corporate Change*, v. 23, n. 1, p. 91–112, 2014. Disponível em: <http://dx.doi.org/10.1093/icc/dtt052>. Citado 2 vezes nas páginas 35 e 38.

23 MONTEIRO, G. F. Empresas de alto crescimento e o desafio de scale-up. *Endeavor/Insper*, 2015. Citado na página 35.

24 DELMAR, F.; DAVIDSSON, P.; GARTNER, W. Arriving at the high growth firm. *Journal of Business Venturing*, Elsevier, v. 18, n. 2, p. 189–216, 2003. ISSN 0883-9026. Citado na página 35.

Apêndices

APÊNDICE A – *Features* Primitivas do *Featuretools*

A.1 Primitivas de Agregação

As primitivas de agregação estão na Figura 37. Na coluna *Name* estão os nomes dos métodos das primitivas, como deverão ser utilizados dentro da ferramenta. A descrição de cada uma pode ser consultada na coluna *Description*.

Figura 37 – Todas as primitivas de agregação que a ferramenta possui até o momento.

	name	type	description
0	skew	aggregation	Computes the skewness of a data set.
1	trend	aggregation	Calculates the slope of the linear trend of variable overtime.
2	median	aggregation	Finds the median value of any feature with well-ordered values.
3	min	aggregation	Finds the minimum non-null value of a numeric feature.
4	time_since_last	aggregation	Time since last related instance.
5	avg_time_between	aggregation	Computes the average time between consecutive events.
6	num_true	aggregation	Finds the number of 'True' values in a boolean.
7	all	aggregation	Test if all values are 'True'.
8	mode	aggregation	Finds the most common element in a categorical feature.
9	std	aggregation	Finds the standard deviation of a numeric feature ignoring null values.
10	n_most_common	aggregation	Finds the N most common elements in a categorical feature.
11	num_unique	aggregation	Returns the number of unique categorical variables.
12	any	aggregation	Test if any value is 'True'.
13	mean	aggregation	Computes the average value of a numeric feature.
14	percent_true	aggregation	Finds the percent of 'True' values in a boolean feature.
15	max	aggregation	Finds the maximum non-null value of a numeric feature.
16	sum	aggregation	Sums elements of a numeric or boolean feature.
17	last	aggregation	Returns the last value.
18	count	aggregation	Counts the number of non null values.

Fonte: *Featuretools*

A.2 Primitivas de Transformação

Existem 43 primitivas de transformação na ferramenta até esse momento. Então, elas foram dispostas em duas figuras: as primeiras 23 primitivas estão na Figura 38 e as outras 20 na Figura 39. As colunas seguem o mesmo padrão explicado para as primitivas de agregação.

Figura 38 – As 23 primeiras primitivas de transformação que a ferramenta possui até o momento.

	name	type	description
19	is_null	transform	For each value of base feature, return 'True' if value is null.
20	cum_sum	transform	Calculates the sum of previous values of an instance for each value in a time-dependent entity.
21	day	transform	Transform a Datetime feature into the day.
22	multiply	transform	Creates a transform feature that multplies two features.
23	characters	transform	Return the characters in a given string.
24	and	transform	For two boolean values, determine if both values are 'True'.
25	add	transform	Creates a transform feature that adds two features.
26	weekend	transform	Transform Datetime feature into the boolean of Weekend.
27	longitude	transform	Returns the second value on the tuple base feature.
28	year	transform	Transform a Datetime feature into the year.
29	time_since	transform	Calculates time since the cutoff time.
30	percentile	transform	For each value of the base feature, determines the percentile in relation
31	mod	transform	Creates a transform feature that divides two features.
32	month	transform	Transform a Datetime feature into the month.
33	diff	transform	Compute the difference between the value of a base feature and the previous value.
34	week	transform	Transform a Datetime feature into the week.
35	cum_max	transform	Calculates the max of previous values of an instance for each value in a time-dependent entity.
36	hours	transform	Transform a Timedelta feature into the number of hours.
37	minute	transform	Transform a Datetime feature into the minute.
38	negate	transform	Creates a transform feature that negates a feature.
39	absolute	transform	Absolute value of base feature.
40	cum_mean	transform	Calculates the mean of previous values of an instance for each value in a time-dependent entity.
41	days	transform	Transform a Timedelta feature into the number of days.

Fonte: *Featuretools*

Figura 39 – As últimas 20 primitivas de transformação que a ferramenta possui até o momento.

	name	type	description
42	divide	transform	Creates a transform feature that divides two features.
43	numwords	transform	Returns the words in a given string by counting the spaces.
44	or	transform	For two boolean values, determine if one value is 'True'.
45	subtract	transform	Creates a transform feature that subtracts two features.
46	weekday	transform	Transform Datetime feature into the boolean of Weekday.
47	haversine	transform	Calculate the approximate haversine distance in miles between two LatLong variable types.
48	time_since_previous	transform	Compute the time since the previous instance.
49	years	transform	Transform a Timedelta feature into the number of years.
50	days_since	transform	For each value of the base feature, compute the number of days between it
51	seconds	transform	Transform a Timedelta feature into the number of seconds.
52	latitude	transform	Returns the first value of the tuple base feature.
53	months	transform	Transform a Timedelta feature into the number of months.
54	not	transform	For each value of the base feature, negates the boolean value.
55	weeks	transform	Transform a Timedelta feature into the number of weeks.
56	cum_min	transform	Calculates the min of previous values of an instance for each value in a time-dependent entity.
57	isin	transform	For each value of the base feature, checks whether it is in a provided list.
58	minutes	transform	Transform a Timedelta feature into the number of minutes.
59	second	transform	Transform a Datetime feature into the second.
60	cum_count	transform	Calculates the number of previous values of an instance for each value in a time-dependent entity.
61	hour	transform	Transform a Datetime feature into the hour.

Fonte: *Featuretools*