UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ESPECIALIZAÇÃO EM ENGENHARIA DE SOFTWARE E INOVAÇÃO

PAULO HENRIQUE KLEIN

Aplicação de Aprendizado de Máquina para automatização de soluções no cenário de Licenciamento Urbanístico

Monografia de Conclusão de Curso apresentada como requisito parcial para a obtenção do grau de Especialista em Engenharia de Software e Inovação.

Orientador: Prof. Dr. Flávio Moreira de Oliveira

Porto Alegre
2021

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Klein, Paulo Henrique

Aplicação de Aprendizado de Máquina para automatização de soluções no cenário de Licenciamento Urbanístico / Paulo Henrique Klein. – Porto Alegre: PPGC da UFRGS, 2021.

65 f.: il.

Monografia (especialização) – Universidade Federal do Rio Grande do Sul. Curso de Especialização em Engenharia de Software e Inovação, Porto Alegre, BR–RS, 2021. Orientador: Flávio Moreira de Oliveira.

1. Ciência de Dados. 2. Aprendizado de Máquina. 3. Automatização de soluções. 4. Licenciamento Urbanístico. I. Moreira de Oliveira, Flávio. II. Título.

RESUMO

A aplicação de Ciência de Dados juntamente com Aprendizado de Máquina trazem opções de grande potencial ao se lidar com dados. Enquanto a Ciência de Dados é capaz de identificar descobertas valiosas sobre os dados, o Aprendizado de Máquina está em constante evolução e é capaz de aprender com os dados, gerando um processo inteligente. Os sistemas de Licenciamento Urbanístico de Porto Alegre também estão em constante evolução e, por isso, exigem melhorias através de diversas oportunidades. O propósito do trabalho está na criação de um projeto de aplicação de Ciência de Dados e Aprendizado de Máquina, como forma de automatizar soluções possíveis aos sistemas. O resultado do trabalho realizado é uma aplicação satisfatória para contemplar a evolução do sistema no caminho de automatizações interessantes.

Palavras-chave: Ciência de Dados. Aprendizado de Máquina. Automatização de soluções. Licenciamento Urbanístico.

Machine Learning application for automation of solutions in the Urban Licensing context

ABSTRACT

The application of Data Science and Machine Learning brings great potential options when dealing with data. While Data Science is able to identify valuable discoveries about data, Machine Learning is constantly evolving and it is able to learn from data, generating an intelligent process. Urban Licensing systems in Porto Alegre are also in constant evolution and, therefore, require improvements through several opportunities. The purpose of this work is to create a Data Science and Machine Learning application project, as a way to automate possible solutions to the systems. The result of the work done is a satisfactory application to contemplate the evolution of the system in the way of interesting automations.

Keywords: Data Science, Machine Learning, Automation of solutions, Urban Licensing.

LISTA DE ABREVIATURAS E SIGLAS

AEIS	Áreas Especiais de Interesse Social
API	Application Programming Interface
AUC	Area Under the Curve
BI	Business Intelligence
BPM	Business Process Management
CART	Classification and Regression Tree
CCCE	Comissão Consultiva do Código de Edificações
CNN	Convolutional Neural Network
CRISP-DM	Cross-Industry Standard Process for Data Mining
DAM	Documento de Arrecadação Municipal
DMI	Declaração Municipal Informativa
EPAHC	Equipe do Patrimônio Histórico e Cultural
EVU	Estudo de Viabilidade Urbanística
Procempa	Companhia de Processamento de Dados do Município de Porto Alegre
ROC	Receiver Operating Characteristic
RPA	Robotic Process Automation
STD	Standard Deviation

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O trabalho desenvolvido consiste em um estudo com caráter de experimentação, na área de Ciência de Dados, através da aplicação de Aprendizado de Máquina sobre um conjunto de dados, sendo demonstrada como uma inteligência capaz de fazer predições, com o objetivo de automatizar processos em sistemas e trazer otimizações ao processo como um todo.

Ciência de dados é a arte e a ciência para adquirir conhecimento por meio de dados, de modo a obter *insights* que, de outra forma, seriam perdidos (OZDEMIR, 2016).

O Aprendizado de Máquina (*Machine Learning*) consiste no processo de construção de sistemas de computador, implementando um processo de aprendizado. Dessa forma, os sistemas melhoram automaticamente com a experiência (GACOVSKI, 2019).

Com a crescente demanda de utilização de serviços de forma digital e integrada na prefeitura de Porto Alegre, um dos projetos que mais se destacam se encontra na área de Licenciamento Urbanístico, que corresponde, basicamente, ao processo de concessão de licenças nos âmbitos urbanístico e ambiental, para execução de processos de edificações, por parte de profissionais legalmente habilitados.

Há um foco em trazer maior automatização a esse processo de Licenciamento, com integrações de sistemas internos da prefeitura, e externos através de parceria com outros órgãos. Nesse sentido, atualmente existe uma suíte de sistemas desenvolvidos na Procempa, reunindo em torno de 200 diferentes tipos de processos. Os sistemas estão disponíveis para o público externo, e os procedimentos para emissão das licenças funcionam totalmente de forma digital, onde o requerente informa diversos dados necessários, e em pouco tempo consegue ter uma licença emitida para consequentemente poder trabalhar. Ainda assim, algumas licenças exigem maiores trâmites dentro da prefeitura. Assim, existe um controle de etapas dentro dos sistemas, até que se possa efetivamente gerar a licença.

Com essa transformação, se observa uma oportunidade de trazer ainda mais automatização ao processo, aplicando-se uma inteligência baseada nos dados, com o objetivo de qualificar e automatizar a geração de novas licenças, e também na questão de segurança, no enfrentamento de possíveis fraudes no sistema.

Para Ethem (2016), um sistema que está em um ambiente em mudança precisa ter a capacidade de aprender, caso contrário, dificilmente seria considerado inteligente. Com um sistema que pode aprender, se adaptando a tais mudanças, o projetista não precisa

prever e fornecer soluções para todas as situações possíveis.

O processo de Licenciamento Urbanístico apresenta várias oportunidades de melhoria, tais como:

- Prevenção de erros: É muito comum um requerente errar algum dado ou documento, e esse processo então passa pela análise dos revisores, que devolvem ao requerente para complementar algo inválido. Isso acaba se tornando um processo repetitivo, e por isso seria interessante prevenir que erros aconteçam por parte dos requerentes.
- Detecção de fraudes: Pelo fato do processo ser automatizado, os usuários acabam passando a confiar mais no sistema na medida em que fazem maior uso dele. Assim, é importante que haja uma forma de aumentar a segurança com relação a possíveis fraudes por parte dos requerentes e demais usuários, que poderiam acabar passando despercebidas. Encontra-se relação na detecção de fraudes com a prevenção de erros, no sentido em que o erro causado pelo usuário pode ou não ter por trás uma intenção de fraude, assim a detecção de padrões de fraudes poderiam ajudar na identificação correta.
- Recomendação de ações dentro do processo: Pela quantidade de etapas envolvidas em alguns processos de Licenciamento, ocorre que isso se torna um processo exaustivo. Então, é interessante ampliar possíveis automatizações para as etapas do processo, focando em recomendação de ações com base na identificação de padrões existentes.
- Extração de relatórios inteligentes: Há uma dificuldade para os analistas da prefeitura em identificar pontos para melhoria no processo de Licenciamento. Nesse sentido, uma solução seria a construção de relatórios com extrações inteligentes, que poderiam ajudar em análises e no aprimoramento da solução, auxiliando o trabalho dos analistas na observação dos dados para a identificação de pontos para melhoria, ou até na identificação das necessidades de maior investimento de trabalho em determinados processos, por exemplo.

Com base nas oportunidades, o propósito do trabalho é realizar um experimento de aplicação de Ciência de Dados e Aprendizagem de Máquina, em uma base de dados de Licenciamento Urbanístico, focando em uma das oportunidades elencadas. Após uma análise sobre o conjunto de dados, foi decidido por selecionar o problema de prevenção de erros, por diversas possibilidades de experimentação encontradas.

Para a execução do trabalho, com teor prático, foi desenvolvido um projeto de Ciência de Dados e Aprendizado de Máquina, cujos resultados servem como aproveitamento para otimização dos sistemas existentes e para futuras melhorias no projeto. Através de avaliações no projeto sobre a solução, foi possível verificar que a mesma é adequada, considerando os resultados e diversos critérios.

Os resultados deste processo servem como base para trabalhos futuros a serem construídos, seja por integrações possíveis, como através da aplicação de uma inteligência, dentro do projeto de Licenciamento Urbanístico na Procempa.

O objetivo principal do trabalho é buscar a otimização de um aspecto importante nos sistemas de informação do Licenciamento Urbanístico de Porto Alegre.

Os objetivos secundários do trabalho contemplam:

- Identificar oportunidades para melhorias nos sistemas de Licenciamento Urbanístico da Procempa;
- Construir um conjunto de dados que contenham informações suficientes para obter soluções;
- Aplicar métodos de Ciência de Dados para análise das informações e descoberta de características;
- Aplicar Aprendizagem de Máquina ao conjunto de dados, visando identificar padrões, para automatizar soluções.
- Executar o processo de forma iterativa e experimental, descobrindo e revelando novas opções e hipóteses para melhorias.

Inicialmente, o Capítulo 1, Introdução, demonstra o contexto de utilização do trabalho, de forma abrangente, os objetivos e as oportunidades e desafios relacionados ao problema e a solução.

O Capítulo 2, Fundamentação Teórica, contempla toda a pesquisa relacionada ao trabalho e aos conceitos utilizados para execução.

No Capítulo 3, Proposta, são apresentados a caracterização do problema e a solução, bem como os levantamentos iniciais do problema, para o desenvolvimento do projeto prático. Apresenta, ainda, a contribuição que se espera com a realização do trabalho.

O Capítulo 4, Desenvolvimento, apresenta o relatório da aplicação prática desenvolvida, desde as ferramentas utilizadas até as avaliações finais da proposta.

E, por fim, o Capítulo 5, Conclusão, demonstra as implicações do experimento realizado e as opções de trabalhos futuros para prosseguimento do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Licenciamento Urbanístico

O Licenciamento Urbanístico de Porto Alegre trata do crescimento urbanístico no município, com a concessão de licenças para execução de processos referentes a edificações, cuja análise pode passar por diversos setores da prefeitura. A responsabilidade das edificações cabe ao profissionais legalmente habilitados e aos proprietários dos imóveis, enquanto a aprovação, licenciamento e fiscalização das obras cabem ao município.

O Decreto Municipal Nº 19.741/2017 (Prefeitura Municipal de Porto Alegre, 2017), estabelece o Licenciamento Expresso, com a finalidade de simplificar e atualizar procedimentos administrativos para aprovação e licenciamento de habitações unifamiliares e licenças (Prefeitura Municipal de Porto Alegre, 2021a).

Para Giugno e Mello (2007), Porto Alegre se destaca entre os municípios da região metropolitana do Estado do Rio Grande do Sul, pela sua consolidada experiência em planejamento urbano, comprovada por seu Plano Diretor (Prefeitura Municipal de Porto Alegre, 2007), pelo pioneirismo de instrumentos urbanísticos estabelecidos antes mesmo da vigência do Estatuto da Cidade (Presidência da República: Casa Civil: Subchefia para Assuntos Jurídicos, 2001), entre outros instrumentos de gestão da cidade.

2.1.1 Expediente Único

Um Expediente Único contém as informações referentes a uma propriedade, dentro do Licenciamento Urbanístico de Porto Alegre.

De acordo com a Prefeitura Municipal de Porto Alegre (2018), Expediente Único é o conceito dado ao processo administrativo referente a uma única propriedade, onde constam todas as informações históricas necessárias ao licenciamento urbanístico desta propriedade, com documentos. Exemplos de informações correspondem a etapas de aprovação de projetos de edificação, vistorias para o Habite-se e parcelamento de solo, entre outras.

2.1.2 Processos

Dentre os Processos referentes a licenças no âmbito de Licenciamento Urbanístico, podem ser citados os processos que tramitam de forma digital. Conforme a Prefeitura Municipal de Porto Alegre (2020), muitos desses são estabelecidos através de decretos ou Lei Complementar, como: Decretos 19.741/2017 (Prefeitura Municipal de Porto Alegre, 2017), 18.623/2014 (Prefeitura Municipal de Porto Alegre, 2014a), 18.828/2014 (Prefeitura Municipal de Porto Alegre, 2014b), 18.886/2014 (Prefeitura Municipal de Porto Alegre, 2014c), 12.715/2000 (Prefeitura Municipal de Porto Alegre, 2000a) e 12.923/2000 (Prefeitura Municipal de Porto Alegre, 2000b), e Lei Complementar 806/2016 (Prefeitura Municipal de Porto Alegre, 2016).

De acordo com a Prefeitura Municipal de Porto Alegre (2020), os processos que tramitam de forma digital, são referentes a:

- Aprovação e licenciamento de edificações (diversos);
- Aprovação e licenciamento de projeto de unidades autônomas em condomínio;
- Aprovação e licenciamento de projeto urbanístico de condomínio por unidades autônomas de habitações unifamiliares ou mistas;
- Auto de infração (diversos);
- Cadastramento de logradouro;
- Certidões (diversos);
- Comissão Consultiva do Código de Edificações (CCCE);
- Comunicação de conclusão das fundações;
- Consulta EPAHC (setor) quanto a bloqueio preventivo constante na DMWeb (sistema);
- Croqui de logradouro;
- Definição de regime urbanístico para lotes matriculados que se encontrem em AEIS (Áreas Especiais de Interesse Social);
- Envio de Expediente Único para outra secretaria para plantão técnico;
- Enquadramento de atividade;
- Estudo de Viabilidade Urbanística (EVU) para projetos especiais de impacto urbano de 1º grau (diversos);
- Fracionamento do solo;

- Impugnação ao inventário do Bairro Petrópolis (localizado no município de Porto Alegre);
- Inclusão de endereço para emissão da DMI (Declaração Municipal Informativa);
- Informação de alinhamento predial;
- Laudos (diversos);
- Licenciamento digital (diversos);
- Licenças na hora (diversos);
- Licenças expressas (diversos);
- Licenciamento expresso de habitação unifamiliar (autolicenciamento);
- Licença para estações transmissoras de radiocomunicação;
- Pedido de tombamento ou inventário de imóvel;
- Potencial construtivo (diversos);
- Ressarciamento de Taxa DAM (Documento de Arrecadação Municipal);
- Vistoria e Habite-se (diversos);

Há ainda, outros processos de Licenciamento Urbanístico, que não tramitam de forma digital.

2.1.3 Documento de Prancha de Projeto Arquitetônico

De acordo com a Prefeitura Municipal de Porto Alegre (2020), dentre os documentos comumente necessários para os Processos de Licenciamento Urbanístico, pode ser citado o Documento de Prancha de Projeto Arquitetônico.

Conforme a Prefeitura Municipal de Porto Alegre (2014a), o documento de Prancha de Projeto Arquitetônico consiste em um documento contendo uma ou mais plantas do projeto arquitetônico, conforme o tipo de projeto, e portanto também o processo (Prefeitura Municipal de Porto Alegre, 2020), com diversos critérios estabelecidos.

2.1.4 Procempa

A Procempa, ou Companhia de Processamento de Dados do Município de Porto Alegre, consiste na empresa pública responsável por manter e desenvolver os sistemas e rede utilizados pela prefeitura de Porto Alegre, bem como para as secretarias do muni-

cípio, entre outros serviços. A Procempa disponibiliza os serviços de interação entre o cidadão porto-alegrense com a Prefeitura de Porto Alegre (Prefeitura Municipal de Porto Alegre, 2021b).

No contexto de Licenciamento Urbanístico, a empresa atua no desenvolvimento e manutenção dos sistemas, fornecendo o desenvolvimento de novos projetos e atualização de sistemas antigos, com o foco em tornar o processo digital e integrado.

2.2 Automação de processos com BPM e RPA

Para os possíveis caminhos nos sistemas, o mapeamento através de *workflow* é empregado como forma de gerenciamento de diferentes caminhos possíveis e automatização de tarefas e processos existentes no fluxo, com o uso de BPM (*Business Process Management*, ou Gerenciamento de Processos de Negócio).

É amplamente aceita a ideia de que as organizações modernas são sustentadas por uma série de processos de negócios que descrevem suas atividades principais. Dessa forma, a noção de processo, ao longo do tempo, foi reconhecida como uma abordagem útil para coordenar atividades complexas baseadas em serviços e conhecimento (RUSSELL; AALST; HOFSTEDE, 2008).

2.2.1 Tarefas BPM

As tarefas BPM são tarefas específicas utilizadas em um sistema BPM, com entradas e saídas e conexão através dos *workflows* do processo.

Conforme mencionam Russell, Aalst e Hofstede (2008), as tarefas constituintes descrevem as atividades de trabalho individuais que compõem o processo e estão conectadas por arcos direcionados indicando os vários caminhos de execução ao longo do processo, estando o modelo de processo de negócios centrado em torno dessa descrição dos aspectos do fluxo de controle.

2.2.2 Utilização de RPA

Para automação de processos em *workflows* é possível aplicar a tecnologia de RPA (*Robotic Process Automation*, ou Automação Robotizada de Processos).

O processo de automação com a tecnologia RPA imita as ações de programação, observando e registrando a execução manual, usando a interface do usuário. Isso minimiza o trabalho do programador que deseja automatizar uma lista de tarefas como um *workflow*, não necessitando que *scripts* dedicados sejam programados (MARTINS et al., 2020).

2.3 Prevenção de erros

A oportunidade de melhoria identificada através da prevenção de erros no sistema de Licenciamento Urbanístico leva em conta a interação do usuário com o sistema diretamente. Os causadores dos erros, nesse caso, são potencialmente os próprios usuários.

Para a Filgueiras e Rodrigues (2006), os analistas obtêm respostas sobre a previsão de erros apenas a partir do momento em que os usuários começam a interagir com o sistema. Dada a complexidade do comportamento cognitivo dos usuários, não se tem uma previsão de erros em determinado sistema ou do momento em que podem ocorrer, até que a interação dos usuários com o sistema comece efetivamente.

2.4 Ciência de Dados

A Ciência de Dados é uma área de conhecimento que compreende toda uma análise que é feita sobre um conjunto de dados de estudo, de modo a descobrir características, podendo estas serem utilizadas para identificação de padrões.

Na Ciência de Dados é executado um trabalho de análise que requer uma quantidade substancial de habilidades de engenharia de *software* (CADY, 2017).

2.4.1 Processos de Ciência de Dados

Conforme afirma Cady (2017), o processo de Ciência de Dados é composto pelas seguintes etapas:

- Enquadramento do problema: Consiste em entender o caso de uso de negócios, e através dele construir um ou mais problemas analíticos, de forma bem definida.
- Compreensão dos dados: Uma análise exploratória, onde se deve vasculhar os da-

dos, visualizá-los de várias maneiras diferentes e experimentar maneiras diferentes de transformá-los.

- Extração de características: Consiste em extrair boas características, sendo o principal ponto a considerar, para que a análise funcione. É considerada a parte mais criativa na Ciência de Dados, estando intimamente ligada à experiência de domínio.
- Modelo: Após a extração de características, esta etapa corresponde a algum tipo de modelo de Aprendizado de Máquina ou de Estatística aplicado, onde são permitidas correções de curso no projeto, e se tem ideias sobre o que fazer de maneira diferente se houver outra iteração. Está presente na maior parte dos projetos de Ciência de Dados.
- Apresentação de resultados: Consiste em apresentar o trabalho feito e seus resultados, seja através de um relatório ou um conjunto de *slides*.
- Implantação de código: É o trabalho de produzir o código, que será executado regularmente por outras pessoas.
- Iterações: Saber quais características são úteis para extrair e qual modelo usar para treinamento não é algo evidente. Por esse motivo, o processo de Ciência de Dados deve ser construído de modo a facilitar alterações, sendo profundamente iterativo.

Como base estabelecida, e de forma análoga e semelhante às etapas apresentadas por Cady (2017), o CRISP-DM (do inglês *Cross-Industry Standard Process for Data Mining*, ou Processo Padrão do Mercado para Mineração de Dados), é um processo padrão estabelecido para executar projetos de mineração de dados. Contém um conjunto de fases, visando alcançar a melhoria contínua do processo. De acordo com SPSS Inc. (2000), as fases são:

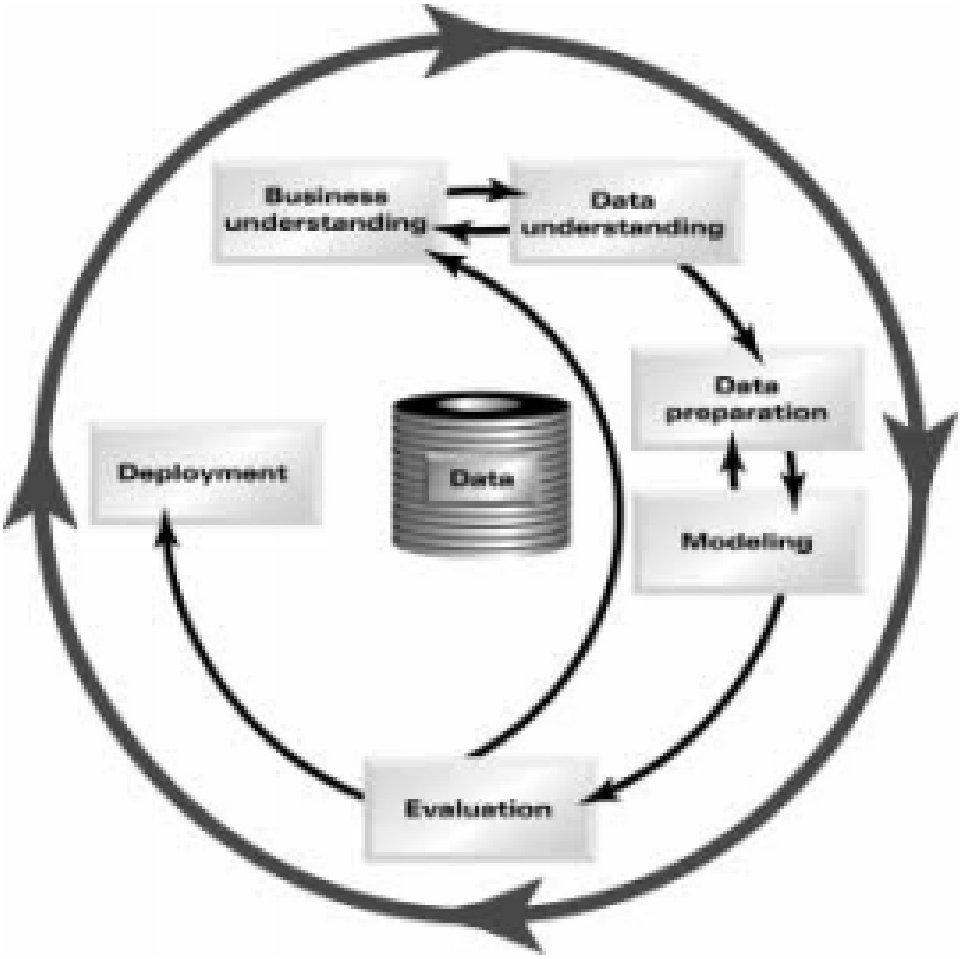
- Entendimento do negócio (*Business understanding*): É a fase inicial, onde se deve compreender os objetivos e e requisitos do projeto da perspectiva de negócios e assim, converter na definição de um problema e em um plano preliminar para atingir esses objetivos de negócios;
- Entendimento dos dados (*Data understanding*): Consiste em coletar os dados e realizar atividades para explorá-los, identificando problemas de qualidade e fazendo descobertas interessantes sobre os dados;
- Preparação dos dados (*Data preparation*): Consiste na construção do conjunto de dados final para as ferramentas de modelagem, a partir dos dados brutos. Essas tarefas geralmente são realizadas várias vezes no decorrer do projeto. Inclui tarefas

como transformação e limpeza de dados e seleção de atributos;

- Modelagem (*Modeling*): Consiste na execução de seleção e aplicação de várias técnicas de modelagem e calibragem de parâmetros com valores ideais;
- Avaliação (*Evaluation*): Consiste na etapa de avaliação mais detalhada do modelo ou dos modelos construídos e na revisão de etapas executadas para garantir que o modelo esteja de acordo com os objetivos de negócio, para então ser implantado;
- Implantação (*Deployment*): Consiste na fase onde são apresentados os resultados e os conhecimentos adquiridos no processo de forma organizada, para então, de acordo com a tomada de decisão da empresa, executar a implantação de modelos construídos no processo.

A Figura 2.1 demonstra o fluxo de fases em CRISP-DM.

Figura 2.1: Fluxo de fases em CRISP-DM



Fonte: SPSS Inc. (2000)

2.4.2 Dataset

O *dataset* corresponde ao conjunto de dados a ser explorado em um projeto de Ciência de Dados.

Para Cady (2017), os dados precisam poder resolver o problema de negócios que se busca resolver. Caso contrário é preciso buscar fontes de dados adicionais ou modificar o trabalho planejado.

2.5 Aprendizado de Máquina

O Aprendizado de Máquina, ou *Machine Learning*, é utilizado para reconhecimento de padrões em um conjunto de dados, onde o sistema, através de um processo de treinamento, cria um modelo que pode ser utilizado para fazer inferências e predições.

Em Aprendizado de Máquina, os sistemas são programados para aprender através da experiência passada. A partir de um conjunto de dados, se observa um problema a ser resolvido, e com um conjunto particular de exemplos representando instâncias desse problema, são utilizados algoritmos de Aprendizado de Máquina, que aprendem a induzir uma função ou hipótese capaz de resolver o problema. Isso é um princípio denominado indução, no qual são obtidas conclusões genéricas (FACELI et al., 2021).

Para Faceli et al. (2021), as tarefas de Aprendizado de Máquina podem ser divididas de acordo com o paradigma de aprendizado adotado para lidar com a tarefa, podendo então ser descritas como Aprendizado Supervisionado e Aprendizado não Supervisionado (Subseções 2.5.1 e 2.5.2, respectivamente). Porém, há ainda outros tipos de aprendizado possíveis, sendo esses:

- Aprendizado Semi-supervisionado: O modelo induzido através de rótulos disponíveis é utilizado para rotular dados não rotulados, procurando aumentar o número de objetos rotulados.
- Aprendizado Ativo: São selecionados apenas os dados que apresentam uma característica ausente nos dados rotulados e que possam melhorar a qualidade do modelo induzido.
- Aprendizado por Reforço: Usado com formas de recompensar uma ação considerada positiva e punir uma ação considerada negativa.

2.5.1 Aprendizado Supervisionado

O Aprendizado Supervisionado se utiliza de rótulos pré-estabelecidos, para cada amostra, que indica um resultado que se quer prever, e com o qual os modelos de Aprendizado de Máquina aprendem com base nas características da amostra.

Segundo Faceli et al. (2021), o paradigma de aprendizado supervisionado ocorre quando um supervisor externo conhece a saída desejada, ou rótulo, para cada exemplo. São tarefas ditas como Preditivas, onde a meta é encontrar uma função, também chamada de modelo ou hipótese, a qual, com base em valores de atributos de entrada nos dados de treinamento, possa ser utilizada para prever um rótulo ou valor que caracterize um novo exemplo. Para isso, cada objeto no conjunto de treinamento deve possuir tanto atributos de entrada como de saída.

Para Faceli et al. (2021), as tarefas supervisionadas se distinguem pelo tipo de rótulos dos dados:

- Classificação: O rótulo é do tipo discreto, onde o atributo contém um número finito ou infinito contável de valores;
- Regressão: O rótulo é do tipo contínuo, no qual o atributo pode assumir um número infinito de valores, geralmente sendo resultados de medidas.

2.5.2 Aprendizado Não Supervisionado

São tarefas não supervisionadas em Aprendizado de Máquina, ou seja, não se utilizam de um rótulo, sendo geralmente tarefas para exploração e descobertas sobre os dados.

Segundo Faceli et al. (2021), o paradigma de aprendizado não supervisionado ocorre quando não se faz uso de um atributo de saída. São tarefas ditas como Descritivas, em que a meta é explorar ou descrever um conjunto de dados. São exemplos:

- Agrupamento: Os dados são agrupados de acordo com sua similaridade;
- Sumarização: O objetivo é encontrar uma descrição simples e compacta para um conjunto de dados;
- Associação: Se encontra padrões frequentes de associações entre os atributos de um conjunto de dados.

2.5.3 Classificação

O processo de Classificação pode ser descrito em termos de passos, modelos, conceitos e técnicas, utilizados na execução do trabalho.

2.5.3.1 Passos

Conforme análise sobre estudos em Harrison (2020), alguns passos para criação de um modelo preditivo, que expandem a metodologia CRISP-DM, utilizados no trabalho, podem ser sintetizados como:

- Problema e solução: Pergunta e descrições iniciais do negócio, problema e solução;
- Coleta: Coleta do *dataset* e limpeza nos dados;
- Seleção de atributos: Seleção de atributos para o modelo, envolvendo transformações no conjunto de dados.
- Análise exploratória: Visa explorar os dados para obter descobertas sobre os atributos e alvos;
- Modelagem: Envolve as adequações para os modelos, as gerações de modelos de Aprendizado de Máquina e inspeções dos resultados.
- Avaliação: Avaliação do modelo através de métricas;
- Iterações: Possibilitam que, com base na avaliação do modelo, se retorne a passos anteriores e se refaça as gerações para melhores resultados;
- Implantação: Após a seleção do modelo mais adequado, com base na avaliação, pode ser feita a implantação do modelo.

2.5.3.2 Modelos de Classificação

2.5.3.2.1 Múltiplos Modelos Harrison (2020) explica que nenhum algoritmo tem bom desempenho em todos os dados. Sendo assim, testes com múltiplos modelos nos ajudam a ter uma noção de qual algoritmo pode ser melhor para o conjunto de dados a ser testado.

Conforme estudos sobre Harrison (2020), para os testes em múltiplos modelos, podem ser executados o conjunto de classificadores a seguir:

- *dummy*;
- regressão logística;
- árvore de decisão;

- K vizinhos mais próximos;
- *Naive Bayes* gaussiano;
- classificador de vetores suporte;
- floresta aleatória;
- classificador *XGBoost*.

Além disso, Harrison (2020) usa um classificador de empilhamento (*Stacking*), que combina a saída dos outros modelos para fazer a predição de um alvo.

2.5.3.2.2 Árvore de Decisão A Árvore de Decisão é um modelo baseado em perguntas sobre os valores de atributos, que faz as tomadas de decisões sobre o caminho a seguir com base nas respostas, até que se chegue a um resultado final, relativo ao alvo a ser previsto.

De acordo com Géron (2019), as Árvores de Decisão são algoritmos versáteis, podendo executar tarefas tanto de classificação como de regressão, entre outras. São muito poderosos, sendo capazes de moldar conjuntos complexos de dados.

Como cita Harrison (2020), em uma árvore de decisão podemos ter uma série de perguntas para prever uma classe alvo. Esse modelo contêm vantagens, como suporte para dados não numéricos em algumas implementações, suporte para lidar com relacionamentos não lineares e uma explicação facilitada com relação a importância dos atributos. O algoritmo padrão utilizado é o CART (*Classification And Regression Tree*, ou Árvore de Classificação e Regressão), que usa a impureza de Gini ou uma medida de índices para tomar as decisões, percorrendo os atributos em um laço e encontrando o valor que forneça a menor probabilidade de erro de classificação.

2.5.3.2.3 Floresta Aleatória A Floresta Aleatória consiste em um conjunto de Árvores de Decisão, que, através de várias predições diferentes, forma um resultado final pela maioria dos votos.

Géron (2019) explica que, ao agregar as previsões de um conjunto de previsores, muitas vezes se obtém uma previsão melhor. O conjunto de Árvores de Decisão pode ser chamado de Floresta Aleatória, sendo um dos mais poderosos algoritmos de Aprendizado de Máquina disponível atualmente.

Conforme cita Harrison (2020), a Floresta Aleatória usa *bagging* para corrigir a tendência das árvores de decisão à super adequação, através da criação de várias árvores

treinadas com subamostras e atributos aleatórios dos dados.

2.5.3.2.4 XGBoost *XGBoost* é um modelo de Aprendizado de Máquina que utiliza o conceito de *Gradient Boosting*.

Conforme menciona Géron (2019), *Gradient Boosting* é um algoritmo de *Boosting* que se tornou popular. Ele adiciona previsores de forma sequencial a um conjunto, cada um corrigindo seu antecessor. A cada iteração, este método ajusta o novo previsor corrigindo os erros residuais feitos pelo previsor anterior.

Harrison (2020) descreve que o *XGBoost* tenta capturar e tratar qualquer padrão nos erros, atá que pareçam ser aleatórios.

2.5.3.3 Conceitos

Conforme estudos sobre Géron (2019) e Harrison (2020), alguns conceitos utilizados no trabalho podem ser descritos como:

- Hiperparâmetros: São parâmetros dos algoritmos de aprendizado, que são regulados para definir critérios de execução desses algoritmos. Esse ajuste de hiperparâmetros é parte importante da construção dos modelos de Aprendizado de Máquina;
- *Outliers*: Refere-se a prováveis anomalias em dados de instâncias, ou amostras;
- Imputações: Consiste em imputar dados em valores ausentes no conjunto de dados, de modo a preencher com valores adequados.
- Vazamento de informações, ou *Leaky Features*: refere-se a atributos que contêm dados sobre o futuro ou um objetivo;
- Matriz de Confusão: Visualização de quantidades de classificações corretas, bem como falso-positivos e falso-negativos.
- Acurácia: É a porcentagem de classificações corretas de um modelo;
- Revocação, ou *Recall*: é a porcentagem de valores positivos classificados corretamente;
- Precisão: É a porcentagem de predições positivas que estão corretas;
- F1: Média harmônica de precisão e revocação;
- Área sob a curva, ou AUC: Métrica para comparar diferentes classificadores, correspondendo a taxa dos realmente positivos em relação à taxa dos falso-positivos. Um valor mais próximo de 1 é melhor e abaixo de 0,5 é pior.

- Curva ROC (*Receiver Operating Characteristic*): É utilizada em gráfico para avaliar classificadores, com base no cálculo de AUC. Quanto mais saliente for a curva, em geral, melhor é o classificador.

2.5.3.4 Técnicas

Através de estudos sobre Géron (2019) e Harrison (2020), técnicas abordadas no trabalho podem ser descritas da seguinte forma:

- Redução de Dimensionalidade: Tarefa cujo objetivo é simplificar os dados sem perder muita informação. Uma das maneiras de efetuar isso é mesclar várias características correlacionadas em uma única característica;
- Detecção de *outliers* (ou anomalias): Consiste na detecção de anomalias em dados de amostras, por exemplo detecção de amostras com transações incomuns para evitar fraudes, defeitos de fabricação etc. Também indica a opção de remover automaticamente *outliers* antes de fornecer a outro algoritmo de Aprendizado de Máquina;
- Balanceamento de classes: Consiste em técnicas aplicadas sobre classes desbalanceadas, ou seja, com proporções de ocorrências muito desiguais, a fim de torná-las balanceadas para os modelos de Aprendizado de Máquina.
- Codificação de rótulos: Técnica de pré-processamento em atributos, a fim de tornar as variações representadas por números. É conveniente para atributos categóricos com alta cardinalidade;
- Codificação *One-Hot*: Técnica de pré-processamento em atributos, usada para separar cada variação de um atributo categórico em uma nova coluna, formando valores binários 0 e 1.

2.6 Plataformas e Linguagem Python

A Linguagem *Python* (Python Software Foundation, 2021g), em conjunto com outras plataformas, se tornam uma combinação importante ao se trabalhar em projetos de Ciência de Dados e Aprendizado de Máquina. *Python* é muito popular nesse contexto.

Segundo Cady (2017), *Python* foi desenvolvida por Guido van Rossum e lançada pela primeira vez em 1991, e por ter várias bibliotecas *open-source* de computação técnica, se tornou uma ferramenta poderosa para análise.

2.6.1 Vantagem do software Open-Source

A Iniciativa *Open-Source* (Opensource.org, 2021) é baseada fundamentalmente no livre licenciamento de código fonte, sem a necessidade de se pagar pelo mesmo, com a possibilidade de livre modificação também.

Para Deek e McHugh (2008), a ideia por trás de *Open-Source* é manter os avanços científicos de desenvolvimento de software, abertos e disponíveis para todos. Com a colaboração da comunidade, é possível entender e melhorar um produto de *software*, continuamente e, quase que instantaneamente, ao ser examinado pela internet. O princípio está no livre acesso e compartilhamento.

2.6.2 Bibliotecas

Exemplos de bibliotecas disponibilizadas na Linguagem *Python*, são:

- *pandas* (NumFOCUS, 2021);
- *NumPy* (NumPy, 2021);
- *random* (Python Software Foundation, 2021h);
- *Matplotlib* (The Matplotlib development team, 2021);
- *Yellowbrick* (The scikit-yb developers, 2021);
- *seaborn* (WASKOM, 2021);
- *Scikit-plot* (NAKANO, 2017);
- *Pandas Profiling* (Python Software Foundation, 2021d);
- *Sweetviz* (Python Software Foundation, 2021j);
- *Category Encoders* (MCGINNIS, 2016);
- *scikit-learn* (scikit-learn, 2021);
- *XGBoost* (xgboost developers, 2020);
- *mlxtend* (Python Software Foundation, 2021b);
- *rfpimp* (Python Software Foundation, 2021i);
- *pickle* (Python Software Foundation, 2021e);
- *PyDotPlus* (Python Software Foundation, 2021f);
- *IPython* (The IPython Development Team, 2021);
- *dtreeviz* (PARR; LAPUSAN; GROVER, 2021);

- *io* (Python Software Foundation, 2021a);
- *Xgbfir* (GitHub, Inc., 2021);
- *os* (Python Software Foundation, 2021c).

2.6.3 Uso do Jupyter Notebook

O *Jupyter Notebook* (Project Jupyter, 2021) é uma ferramenta capaz de prover os recursos necessários para os projetos de Ciência de Dados e Aprendizado de Máquina, com o uso da Linguagem *Python*.

Conforme consta em Project Jupyter (2021), o *Jupyter* é um aplicativo *open-source*, disponibilizado através de um navegador *web*. Alguns usos da ferramenta incluem limpeza e transformação de dados, simulação numérica, modelagem estatística, visualização de dados e Aprendizado de Máquina, entre outros.

2.6.4 MongoDB

MongoDB (MongoDB, Inc., 2021) é um *software open-source* de banco de dados, orientado a documentos.

Conforme se observa em MongoDB, Inc. (2021), MongoDB consiste em uma plataforma de dados com um conjunto de ferramentas abrangente, que visa facilitar o trabalho de quem a utiliza.

2.6.5 Javascript

A linguagem *Javascript* consiste, inicialmente, em uma linguagem de *script* utilizada em navegadores *web*, para controle de elementos das páginas, embora seja também utilizada em aplicações fora do contexto de páginas *web* mais recentemente.

De acordo com Mozilla and individual contributors (2021), *Javascript* é uma linguagem de programação interpretada ou compilada *just-in-time*, sendo uma linguagem dinâmica baseada em protótipo, multiparadigma, *thread* único e suportando estilos orientados a objetos, imperativos e declarativos.

2.7 Trabalhos relacionados

Para comparar e embasar o trabalho, foram feitas pesquisas relacionadas a aplicações de Aprendizado de Máquina nos setores públicos, e possibilidades de automatizações diversas.

Joner, Santos e Silva (2020) realizaram um estudo no âmbito de Segurança Pública no Estado do Rio Grande do Sul, mais especificamente em dados de Porto Alegre. Foi feita uma análise exploratória visando identificar áreas com maior probabilidade de crimes, e utilizando Aprendizado de Máquina, através de modelos preditivos diversos e métricas próprias, foram obtidos ganhos em resultados, que análises convencionais não obteriam. O estudo foi capaz de demonstrar uma obtenção de melhorias ao utilizar o Aprendizado de Máquina, sobretudo em um contexto tão crítico que é a Criminalidade, e reforçou a necessidade do emprego de tecnologias mais recentes, como o Aprendizado de Máquina, no setor público. Por fim, o melhor modelo obtido foi o de Classificação.

Este estudo possui relação com o presente trabalho, na medida em que se observa as oportunidades de melhoria demonstradas no setor público, e sobretudo no mesmo município, de Porto Alegre. É possível desempenhar aplicações de Ciência de Dados, com modelos de Aprendizado de Máquina, que notadamente terão grande oportunidade de melhorias, com valor a ser agregado no emprego de questões públicas. Isso traz benefícios aos cidadãos porto-alegrenses, e serve como modelo de utilização para outros municípios.

É possível verificar que diferentes abordagens de Aprendizado de Máquina funcionam melhor em questões específicas, com isso há relação sobre os diversos testes executados no decorrer do trabalho com diferentes modelos de Aprendizado de Máquina.

Martins et al. (2020) demonstram uma automatização combinando o uso de RPA (*Robotic Process Automation*) com CNN (*Convolutional Neural Network*). Esta técnica está presente no contexto de *Deep Learning*, sendo mais complexa e eficiente se comparada com técnicas tradicionais de Aprendizado de Máquina.

Este estudo visa trazer uma proposta de automatização de tarefas repetitivas, onde o Aprendizado de Máquina é usado para detectar objetos nas telas e classificá-los, e assim aprender questões relativas às tarefas para aplicar automações através de RPA.

O trabalho possui uma semelhança no sentido de poder utilizar um RPA ligado ao Aprendizado de Máquina, para buscar uma automatização de tarefas de fluxo BPM presentes no contexto do sistema de Licenciamento Urbanístico, ou para automatização de soluções relacionadas a essas tarefas, como por exemplo, a prevenção de erros.

3 PROPOSTA

3.1 Problema

Dentre os problemas elencados (Capítulo 1), decidiu-se pela seleção do problema de prevenção de erros (Seção 2.3), dadas as possibilidades a serem exploradas pelos modelos de Aprendizado de Máquina.

3.1.1 Seleção do problema

Através de uma análise na base de dados do sistema de Licenciamento Urbanístico (Seção 2.1), foi identificado que o problema de prevenção de erros pode ser tratado com maior facilidade em comparação com as outras oportunidades de melhoria levantadas, sendo estas: detecção de fraudes, recomendações de ações dentro do processo e extração de relatórios inteligentes.

Há a possibilidade de se trabalhar com diferentes tipos de erros (Subseção 3.1.2), para exploração pelos modelos de Aprendizado de Máquina.

A ocorrência de um erro possui alguns indicativos, ao observar o registro de um Processo (Subseção 2.1.2) de Licenciamento Urbanístico, o que leva a se esperar uma boa solução através de uma aplicação de Aprendizado de Máquina (Seção 2.5).

Dessa forma, por esses motivos, a prevenção de erros foi o problema escolhido para enfrentamento.

3.1.2 Tipos de erros

Os tipos de erros podem ser separados por sua natureza. Todos ocorrem através da interação entre um requerente e um analista da prefeitura, através de um Processo de Licenciamento Urbanístico dentro do sistema. Foram identificados cinco tipos diferentes. São eles:

- Complemento de documentos: Ocorre quando um requerente envia um ou mais documentos ao Processo, que é então analisado pelo analista da prefeitura, que, ao verificar alguma inconsistência, devolve o Processo ao requerente para que seja feita a complementação de documentos;

- Complemento de prancha: Ocorre quando é solicitado algum complemento para o documento referente a Prancha do Projeto Arquitetônico (Subseção 2.1.3);
- Documento inválido: Ocorre quando o analista identifica um documento inválido e atualiza sua situação;
- Expediente Único inválido: Ocorre quando o analista identifica que o Expediente Único (Subseção 2.1.1) informado pelo requerente é inválido;
- Indeferimento de Expediente Único: Ocorre quando há a situação de Indeferimento do Expediente Único associado ao Processo.

3.2 Solução

Para a solução do problema, foi construído um projeto prático.

Através do processo de Ciência de Dados (Seção 2.4), as informações obtidas em dados do Licenciamento Urbanístico puderam ser compreendidas e analisadas.

Com a extração de características, e com a aplicação de Aprendizagem de Máquina (Seção 2.5), foi possível executar treinamentos através de modelos, de modo a predizer padrões, para então se obter soluções automatizadas no aspecto de prevenção de erros.

3.2.1 Definição da solução para o problema

A pergunta adequada a se fazer para solucionar o problema identificado é:

"Existe erro de determinado tipo, com base nas características de um Processo de Licenciamento Urbanístico?"

Esta pergunta se caracteriza como um problema comum de Classificação (Subseção 2.5.3), pois deve-se prever um resultado verdadeiro ou falso. Dessa forma, o projeto deve ter como resultado um modelo preditivo de Aprendizado de Máquina. Para isso, é necessária a construção de um modelo de Aprendizado Supervisionado (Subseção 2.5.1), com a atribuição de rótulo(s) identificando o resultado desejado para cada amostra.

3.2.2 Descrição do projeto aplicado

O projeto aplicado ao problema para compor a solução, consiste em um projeto de Ciência de Dados e Aprendizado de Máquina.

O projeto é executado de forma iterativa, na qual é gradualmente melhorado através de novos *insights* sobre os dados, utilizando métodos (Subseção 2.4.1) de Ciência de Dados, e novas aplicações (Subsubseção 2.5.3.2) presentes no contexto de Aprendizado de Máquina.

Para a execução do projeto, é necessária a construção de um *dataset* (Subseção 2.4.2), contendo os dados a serem utilizados, referentes aos Processos do sistema de Licenciamento Urbanístico.

Para o desenvolvimento do projeto (Capítulo 4) foi utilizado um roteiro e uma sequência de passos própria, tendo como base as metodologias de Ciência de Dados (Subseção 2.4.1) e Aprendizado de Máquina (Subsubseção 2.5.3.1).

3.3 Contribuição do trabalho

O trabalho desenvolvido serve como um modelo para aplicação de técnicas de Ciência de Dados e Aprendizado de Máquina dentro do ambiente dos sistemas de Licenciamento Urbanístico de Porto Alegre, na Procempa (Subseção 2.1.4), podendo ser reaproveitado no sentido de prevenção de erros automatizada, bem como para as demais oportunidades que podem ser exploradas (Capítulo 1).

Com o trabalho desenvolvido, se obtém a opção de utilização dentro do Projeto de Licenciamento Urbanístico na Procempa, como forma de auxílio ao sistema para o qual foi desenvolvido.

O trabalho contribui, ainda, com a ideia de possíveis automatizações de soluções para o sistema de Licenciamento Urbanístico, através da utilização e estudo da aplicação desenvolvida.

4 DESENVOLVIMENTO

4.1 Ferramentas utilizadas

O conjunto de ferramentas utilizadas no projeto prático pode ser sintetizado como todos os dispositivos de *software* usados para o desenvolvimento.

4.1.1 Linguagem e plataforma de programação

Para a execução do projeto, foi empregada a *Linguagem Python* (Seção 2.6) como linguagem de programação, bem como sua plataforma. A linguagem é *open-source* (Subseção 2.6.1), o que indica a possibilidade de colaboração constante em seu desenvolvimento, pela comunidade ativa de usuários. Possui possibilidades para inúmeras aplicações, e acabou se tornando a linguagem mais utilizada para as áreas de Ciência de Dados e Aprendizado de Máquina, com diversas bibliotecas disponibilizadas, e constantemente aprimoradas.

4.1.2 Ambiente de desenvolvimento

Como ambiente de desenvolvimento para o projeto prático, foi utilizado o *Jupyter Notebook* (Subseção 2.6.3). O *Jupyter Notebook* consiste em uma ferramenta *open-source*, executada através de um navegador web. É amplamente difundido e aceito entre os iniciantes na área de Ciência de Dados e Aprendizado de Máquina, permitindo implementar e executar os passos correspondentes a essa área.

4.1.3 Bibliotecas

As bibliotecas, disponíveis através da Linguagem *Python*, que foram utilizadas, podem ser separadas por funcionalidade utilizada dentro do projeto. As funcionalidades e as bibliotecas (Subseção 2.6.2) relacionadas, são:

- Análise e manipulação de dados: *pandas*;
- Computação numérica: *NumPy*;

- Uso de aleatoriedade para elementos em vetor: *random*;
- Geração e visualização de gráficos: *Matplotlib*, *Yellowbrick*, *seaborn*, *Scikit-plot*;
- *Dashboards* automáticos com análises: *Pandas Profiling*, *Sweetviz*;
- Transformação de dados categóricos: *Category Encoders*;
- Geração e utilização de modelos preditivos: *scikit-learn*, *XGBoost*, *mlxtend*, *rfpimp* (apenas para utilização de modelo, para *Random Forest*, ou Floresta Aleatória), *pickle* (para implantação de modelos);
- Geração de imagens: *PyDotPlus*, *IPython*, *dtreeviz* (para *Decision Tree*, ou Árvore de Decisão);
- Geração e leitura de arquivos: *io* (entrada e saída), *Xgbfir* (para *XGBoost*);
- Chamadas ao Sistema Operacional: *os*.

4.2 Conjunto de dados

O conjunto de dados está sendo descrito como a etapa anterior à execução do projeto prático, que envolve a geração do *dataset*, extraído da base de dados do sistema, além de estudo e verificações de possibilidades sobre os dados.

4.2.1 Extração de dados e construção do dataset

A extração dos dados para a criação do *dataset* foi feita sobre o banco de dados de Produção do sistema de Licenciamento Urbanístico, de uma base de dados MongoDB (Subseção 2.6.4). Através de um *script* desenvolvido para isso, na Linguagem *JavaScript* (Subseção 2.6.5), o *dataset* é confeccionado. Cada registro criado no *dataset* é uma imagem de um estado de registro de Processo de Licenciamento Urbanístico, por operação executada dentro do sistema, através da extração do histórico dos Processos. Para cada registro gerado é rotulado um valor verdadeiro ou falso para cada um dos cinco tipos de erros possíveis. Além disso, para cada registro também é criado um sexto rótulo, verdadeiro para o caso de existir qualquer um dos cinco erros, ou falso para o caso contrário.

4.2.2 Descrição do dataset

As características e detalhamento do *dataset* são os seguintes:

- Arquivo: o arquivo do *dataset* está no formato *xls*. A planilha contém um total de 7961 linhas e 177 colunas.
- Observações: O *dataset* possui um total de 7961 observações (linhas). Cada observação do *dataset* corresponde a um estado de registro de um Processo, com seus atributos (variáveis), por operação executada no sistema.
- Variáveis: No *dataset* existem 171 variáveis (colunas que não são rótulos). A Tabela 4.1 exibe a lista de variáveis, contendo nome, tipo de dado e descrição do atributo. Os atributos referentes a vetores (listas) possuem duas colunas, a primeira representando o nome do vetor e a segunda os atributos contidos neste.

Tabela 4.1: Variáveis do dataset

Nome		*Tipo*	*Descrição*
bpmProcessInstance		numérico (*float64*)	número do processo no BPM
bpmTasks (lista de tarefas BPM)	*contrato.docsInvalidosContractInput*	booleano (*object*)	indica que há documentos inválidos
	data	*string* (*object*)	data e hora de execução
	id	numérico (*float64*)	identificador da tarefa BPM
	taskData	*string* (*object*)	objeto contendo o motivo de indeferimento
	taskName	*string* (*object*)	nome da tarefa BPM
	username	*string* (*object*)	*username* do usuário que executou a tarefa BPM
	contrato.existeEUContractInput	booleano (*object*)	indica se há Expediente Único já existente

taskData.solicitadaCriacaoNovoEU	booleano (*object*)	indica se foi solicitada Criação de Novo Expediente Único
contrato.analiseEVUContractInput	booleano (*object*)	indica se foi solicitada Análise de Expediente Único
contrato.complementacaoDocsContractInput	booleano (*object*)	indica se foi solicitada Complementação de Documentos
contrato.consAssessoriaContractInput	booleano (*object*)	indica se foi solicitada Consulta à Assessoria Jurídica
contrato.consultaEDIContractInput	booleano (*object*)	indica se foi solicitada Consulta à EDI
contrato.euInvalidoContractInput	booleano (*object*)	indica se Expediente Único é inválido
contrato.indeferirEUContractInput	booleano (*object*)	indica se Expediente Único deve ser indeferido
contrato.prazoComparecimentoContractInput	*string* (*object*)	data de prazo para Complementação de Documentos
contrato.tipoEuContractInput	enumeração (*object*)	tipo de Expediente Único
contrato.tramitandoContractInput	booleano (*object*)	indica se está tramitando
taskData.dataAprovProjModificacao	*string* (*float64*)	data de aprovação de Projeto de Modificação
taskData.dataEvu	*string* (*float64*)	data do Expediente Único
taskData.isDemandadoEvuPreviamente	booleano (*object*)	indica se Expediente Único foi demandado previamente

	taskData.isEm CondicaoDeAnalise	booleano (*object*)	indica se está em condição de análise
	taskData.motivo Indeferimento	*string* (*object*)	texto de motivo de indeferimento
	taskData.indeferirEU	booleano (*object*)	indica se Expediente Único deve ser indeferido
	taskData.parecer	*string* (*object*)	texto de parecer
convertido		booleano (*object*)	indica se processo foi convertido
createdAt		*string* (*object*)	data e hora de criação do processo
dadosFormulario.idTipoFormulario		enumeração (*object*)	tipo de processo
dadosFormulario.idTipoProcessoSei		enumeração (*float64*)	identificador de tipo de processo do SEI
dadosFormulario.idUnidadeSei		enumeração (*object*)	nome da Unidade do SEI
data.dadosHomologacaoIndeferimento.indeferirEU		booleano (*object*)	indica se deve ser indeferido o Expediente Único, na Homologação de Indeferimento
data.dadosHomologacaoIndeferimento.parecer		*string* (*object*)	texto de parecer, na Homologação de Indeferimento
data.dadosTriagem.dataAprovProj Modificacao		*string* (*float64*)	data de aprovação de Projeto de Modificação, na Triagem
data.dadosTriagem.dataEvu		*string* (*float64*)	data do Expediente Único, na Triagem

data.dadosTriagem.isDemandadoEvu Previamente		booleano (*object*)	indica se Expediente Único foi demandado previamente, na Triagem
data.dadosTriagem.isEmCondicaoDe Analise		booleano (*object*)	indica se está em condição de análise
data.dadosTriagem.motivoIndeferimento		*string* (*object*)	texto de motivo de indeferimento, na Triagem
data.dataCriacaoEtapa		*string* (*object*)	data de criação de Etapa
data.enderecoCdl.cep		numérico (*float64*)	CEP do endereço
data.enderecoCdl.nomeBairro		*string* (*object*)	nome do bairro do endereço
data.enderecoCdl.nomeLogradouro		*string* (*object*)	nome do logradouro do endereço
enderecoCdlList (lista de endereços)	*cep*	numérico (*float64*)	CEP do endereço
	nomeBairro	*string* (*object*)	nome do bairro do endereço
	nomeLogradouro	*string* (*object*)	nome do logradouro do endereço
data.motivoIndeferimento		*string* (*object*)	texto de motivo de inferimento
documentos (lista de documentos)	*extensao*	enumeração (*object*)	extensão do documento
	id	*string* (*object*)	identificador do documento
	idDocumento	enumeração (*object*)	tipo de documento
	newDoc	booleano (*object*)	indica se é novo documento
	obrigatorio	booleano (*object*)	indica se é obrigatório
	tituloDocumento	enumeração (*object*)	título do documento

	descricaoOutro Documento	*string* (*object*)	descrição de outro do-documento
	fromSolicitante	booleano (*object*)	indica se é oriundo do solicitante
updatedAt		*string* (*object*)	data e hora de atualização do registro do processo
operations.0		enumeração (*object*)	descrição da operação executada

Fonte: Autor

- Rótulos: Há seis tipos de rótulos por observação. Os rótulos são supervisionados e foram criados para indicar os tipos de erros possíveis. A Tabela 4.2 exibe os rótulos, contendo as informações de nome, tipo de dado e descrição de qual erro é referenciado pelo rótulo.

Tabela 4.2: Rótulos supervisionados

Nome	*Tipo*	*Descrição*
Rótulo_Erro _Complemento_Docs	booleano (*bool*)	indica erro de Complemento de Documentos
Rótulo_Erro _Complemento_Prancha	booleano (*bool*)	indica erro de Complemento de Prancha
Rótulo_Erro _Documento_Inválido	booleano (*bool*)	indica erro de Documento Inválido
Rótulo_Expediente_Único _Inválido	booleano (*bool*)	indica erro de Expediente Único Inválido
Rótulo_Indeferimento _Expediente_Único	booleano (*bool*)	indica erro de Indeferimento de Expediente Único
Rótulo_Possui_Erro	booleano (*bool*)	indica que qualquer um dos cinco erros listados está ocorrendo

Fonte: Autor

4.3 Coleta

Nesta etapa inicial do projeto prático, foi realizada a coleta (Subsubseção 2.5.3.1) do *dataset* e as adaptações para que se tenha um conjunto de dados adequado para utilização nas etapas posteriores do projeto.

4.3.1 Limpezas e correções do dataset

Para executar uma leitura mais adequada do conjunto de dados dentro do projeto, são feitas correções e limpezas necessárias.

4.3.1.1 Limpeza manual e colunas duplicadas

A versão do *dataset* gerado possuía muitos dados sensíveis de usuários do sistema. Além disso, alguma falha imprevista na conversão para o formato *xls* dentro do script de construção do *dataset*, causou problemas de colunas ou linhas com dados desorganizados. Para resolver esses pontos, foi feita uma limpeza manual diretamente no arquivo do *dataset*.

As observações duplicadas são excluídas, para que não se tenha o mesmo registro de estado de Processo mais de uma vez.

4.3.1.2 Correções de dados discrepantes

Após carregar o conjunto de dados na aplicação, ao analisar o *dataframe* com uma varredura, se observa algumas inconsistências nos dados, principalmente por colunas terem dados fora do padrão esperado.

Para realizar a correção desses dados discrepantes (*outliers* (Subsubseção 2.5.3.3)), foi montado e codificado um *script* específico.

4.3.1.3 Consideração sobre dados ausentes

Pela estratégia de se utilizar os dados ausentes como possibilidade de estado de um atributo de Processo de Licenciamento Urbanístico, os dados ausentes nas colunas não são removidos.

O processo de executar imputações (Subsubseção 2.5.3.3) nesses dados ausentes,

traria o mesmo resultado, ao considerar as colunas em questão como dados categóricos nas etapas posteriores.

4.3.1.4 Observação sobre vazamento de informações em atributos

No projeto, existe a estratégia de tentar prever um rótulo a partir do maior número de atributos possível nas amostras, mesmo que apenas um atributo leve ao resultado esperado. Além disso, nem todos os atributos são considerados ao gerar os modelos de Aprendizado de Máquina, em etapas posteriores.

Diante desses pontos, não são observados atributos que possam causar vazamento de informações, ou *leaky features* (Subsubseção 2.5.3.3).

4.3.2 Ajustes e transformação de variáveis

Com o objetivo de melhorar a estrutura de atributos, para as colunas que são do tipo lista no *dataset*, que são as referentes a tarefas BPM (Subseção 2.2.1) e documentos, conceitos do sistema, foi projetada uma etapa de transformação destas, pegando-se a coluna chave respectiva ao elemento e criando novas colunas em substituição à estrutura de colunas como índices de vetores. Essa etapa é muito importante para a correta adequação dos atributos para os modelos de Aprendizado de Máquina. Com isso foi montado um *script* capaz de executar essa transformação de atributos.

Atributos cujos dados são ausentes em todas as amostras foram excluídos.

Após efetuar os ajustes nos dados se obtém um conjunto de dados com o total de 6583 observações e 1179 variáveis (atributos).

4.4 Seleção de atributos

Nesta etapa, foi feita a seleção de atributos (Subsubseção 2.5.3.1) com maior relevância dentro do conjunto de dados, para compor a modelagem (Subsubseção 2.5.3.1) de Aprendizado de Máquina.

4.4.1 Ausência de atributos de dados contínuos

Ao verificar os dados no conjunto, se observa que não há atributos numéricos relativos a possíveis medidas.

Sendo assim, a estratégia para seleção de atributos passa a ser identificar possíveis atributos que possam ser categóricos, a fim de obter um bom aproveitamento para as predições.

4.4.2 Escolha de atributos para a modelagem

A estratégia é selecionar o máximo possível de atributos que possuem variações, sendo categóricos, para ter um modelo com maior aproveitamento possível das características presentes no registro, e assim, aumentar a exatidão do modelo de dados.

Ao fazer uma varredura sobre os dados e atributos, se observa que atributos referentes a tarefas e documentos, sendo estes conceitos dentro do sistema, são características importantes dentro de um Processo, e por isso, são bons candidatos a compor o modelo. Outros atributos também apresentam esta característica.

As colunas referentes a tarefas e documentos, possuem 7 e 203 variações de tipos, respectivamente. Dessa forma, a quantidade de atributos selecionados, dentro destas colunas, é multiplicada por cada variação.

Dentre os atributos, se apurou que os que melhor se enquadram para seleção são:

- *bpmTasks* (7 diferentes):
  - *contrato.tipoEuContractInput*;
  - *taskName*;
- *dadosFormulario.idTipoFormulario*;
- *dadosFormulario.idTipoProcessoSei*;
- *dadosFormulario.idUnidadeSei*;
- *documentos* (203 diferentes):
  - *extensao*;
  - *idDocumento*;
  - *tituloDocumento*;
- *operations.0.*

4.4.3 Adaptações para dataframe e séries

São gerados *dataframe* e séries para utilização no projeto.

4.4.3.1 Criação de dataframe para os atributos e séries para os rótulos

Os atributos selecionados são separados em um *dataframe* (X) contendo todos os atributos de todas as amostras, e cada um dos seis rótulos é separado em uma série específica (y) contendo o rótulo respectivo para todas as amostras. (Subsubseção 2.5.3.1)

4.4.3.2 Transformação do dataframe

No *dataframe* X, referente aos atributos, é utilizada a Codificação de Rótulos (*Label Encoder*) (Subsubseção 2.5.3.3) para transformar os dados em números, de acordo com as variações presentes em cada atributo. Essa escolha foi feita para se obter uma boa visualização de determinados gráficos nas análises de dados exploratórias.

4.5 Análise exploratória

Com relação a esta etapa, são apresentadas as principais análises de dados exploratórias realizadas.

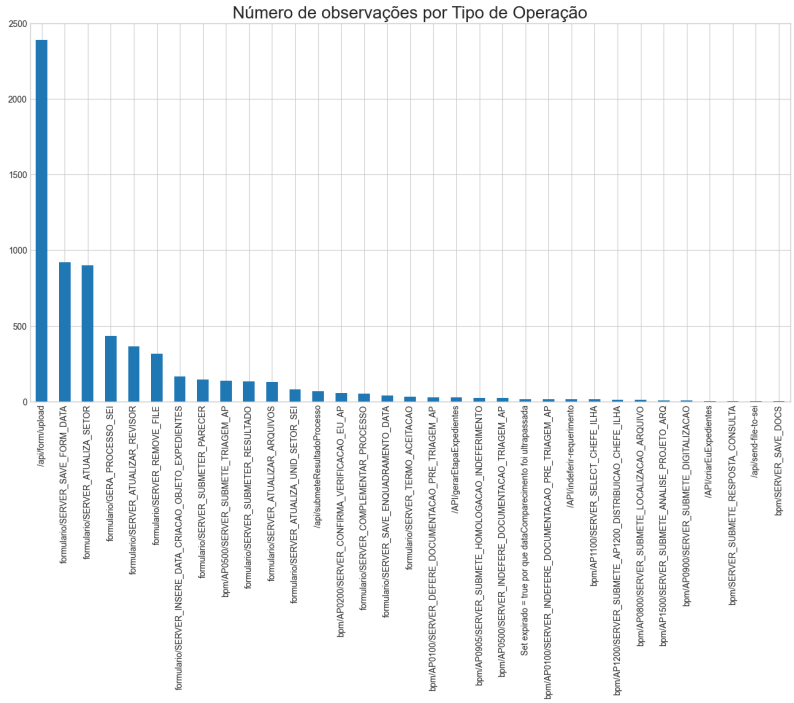
Foram realizadas diferentes abordagens para conferências dos dados e comparações entre eles. Gráficos variados foram gerados, e através da visualização foi possível obter uma percepção ampla com relação aos dados. Por não haver variáveis numéricas representando medidas, ou dados contínuos, não se fez uso de maiores opções de gráficos.

4.5.1 Quantidades de amostras por operações e por rótulos

A Figura 4.1 apresenta o gráfico, representando a quantidade de amostras por tipo de operação executada no sistema, e a Figura 4.2 apresenta os gráficos com as quantidades de amostras por tipo de erro.

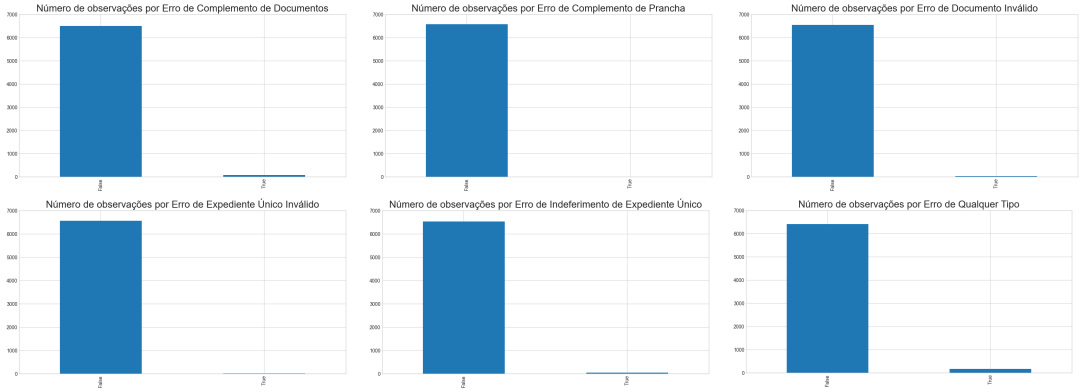
Com relação a Figura 4.2, se observa que as quantidades de erros são poucas se comparadas com os registros sem erros. Isso é aceitável, na medida em que os casos de erro, embora sejam comuns em alguns casos, não são tanto quanto em registros sem erros.

Figura 4.1: Quantidades de amostras por tipo de operação



Fonte: Autor

Figura 4.2: Quantidades de amostras por tipos de erros



Fonte: Autor

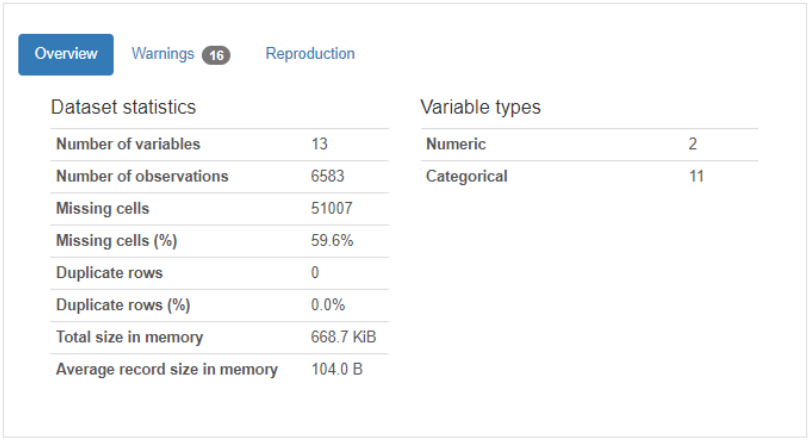
4.5.2 Visualização de dashboards sobre os atributos

Foram gerados *dashboards* automáticos sobre o *dataset*, sem considerar os atributos referentes a documentos.

A Figura 4.3 informa alguns dados de estatística sobre as amostras e atributos, de modo geral, para o conjunto de dados analisado.

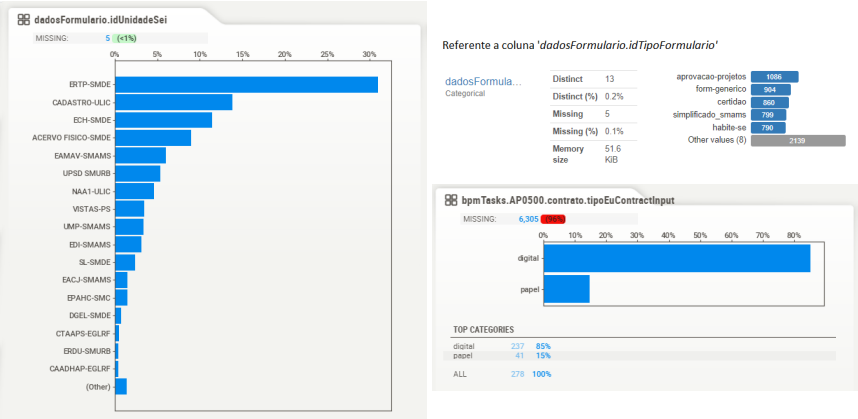
A Figura 4.4 exibe as distribuições por variações e outras estatísticas sobre determinados atributos categóricos.

Figura 4.3: Estatísticas sobre amostras e atributos



Fonte: Autor

Figura 4.4: Distribuições por variações e outras estatísticas

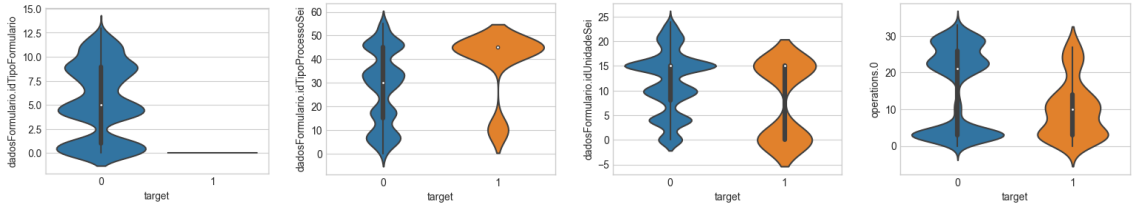


Fonte: Autor

4.5.3 Análises comparativas com os alvos

A Figura 4.5 exibe gráficos onde se pode ter uma ideia da distribuição das diferentes variações de cada um dos atributos com relação a um alvo (ou rótulo). Nesse caso o rótulo escolhido como alvo é o de erro de qualquer tipo.

Figura 4.5: Distribuição de variações com relação ao alvo

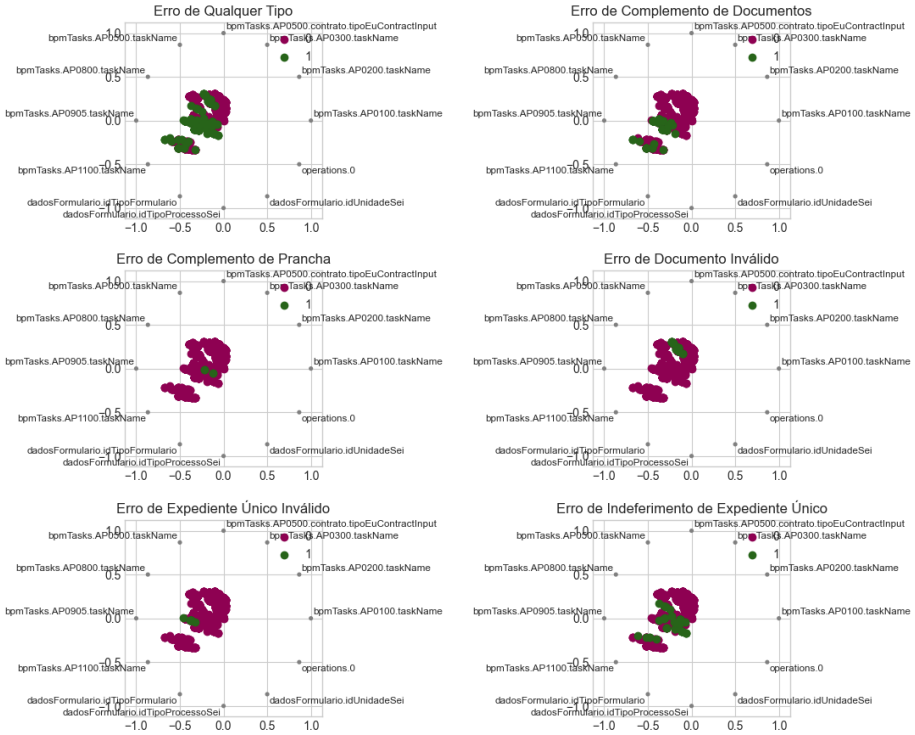


Fonte: Autor

A Figura 4.6 exibe gráficos para cada um dos rótulos de erros, onde se pode perceber a concentração de todas as amostras, sendo exibido o grau de separação entre ter

erro ou não ter, nas amostras, sendo influenciado pelos valores dos atributos ao redor.

Figura 4.6: Concentração de amostras por alvos



Fonte: Autor

4.6 Modelagem de Aprendizado de Máquina

Esta etapa contemplou a geração de modelos (Subsubseção 2.5.3.1) e as inspeções dos resultados, bem como adequações anteriores para utilização dos modelos e a posterior otimização de um modelo selecionado.

As gerações dos modelos começam com diversos testes com um modelo de base e múltiplas famílias de algoritmos. Após isso, se explora os modelos baseados em árvore, que são árvore de decisão, floresta aleatória e *XGBoost*, dada suas características de classificação. O modelo de floresta aleatória, escolhido para aplicação da solução, é analisado com relação a uma possível otimização.

4.6.1 Técnicas não utilizadas

Pela estratégia de considerar que todos os atributos podem ser importantes na previsão dos alvos, técnicas para redução de dimensionalidade (Subsubseção 2.5.3.4) de

colunas não são aplicadas para não prejudicar as previsões. Em consonância com a ideia de redução de atributos para otimizar os modelos, já é feita uma remoção de atributos.

Como se considera que foi feita uma avaliação sobre os dados diretamente no *dataset*, bem como uma limpeza de discrepâncias nos dados, técnicas para detecção de *outliers* (Subsubseção 2.5.3.3) também não são aplicadas.

Como não há dados numéricos contínuos, ou medidas, no *dataframe*, não são necessários pré-processamentos de dados (Subsubseção 2.5.3.1) para padronização de dados e escala de intervalo.

4.6.2 Adequação para a modelagem

Para adequação de dados categóricos para o modelo, esses dados, anteriormente transformados com a Codificação de Rótulos, agora são transformados com a Codificação *One-Hot* (*One-Hot Encoding*) (Subsubseção 2.5.3.4), onde são criadas colunas de atributos por variação de cada atributo. Em cada coluna criada, é colocada a representação binária 1 ou 0, para verdadeiro ou falso, para utilização nos modelos. Isso acaba multiplicando e aumentando a quantidade de colunas no conjunto de dados.

4.6.3 Utilização de balanceamento de classes

Para a execução e geração dos modelos, foram propostos dois caminhos possíveis:

- Sem considerar balanceamento de classes: Ao observar o contexto do problema, que é a detecção de erros, e sendo os dados do *dataset* provenientes da base de dados de produção do sistema de Licenciamento Urbanístico, se considera que os erros são casos raros, tendo um peso proporcional já distribuído em conjunto com o restante dos registros sem erros. Assume-se, então, o conjunto de amostras da forma que está.
- Considerando o balanceamento de classes: Foi considerada a aplicação de um balanceamento de classes (Subsubseção 2.5.3.4) no rótulo referente a qualquer erro, na proporção de 50% para erro e 50% caso contrário. Para isso foi criado um *script* manualmente, onde se realiza a divisão do conjunto de amostras retirando-se o excedente de amostras sem erros, de forma aleatória, com relação a quantidade de amostras com erros. Assim se chega em um conjunto com distribuição da classe

em metade para cada possibilidade, contendo 338 amostras (169 com erro, e 169 sem erro). Como há aleatoriedade na exclusão de amostras sem erros, os resultados nas gerações de modelos podem variar, mas pouco. Aqueles que compõem esse relatório correspondem a uma das variações de balanceamento.

4.6.4 Separação de amostras de treinamento e teste

Para cada um dos seis rótulos são criadas separações dos dados em *dataframes* (X) de treino e teste, e séries (y) de treino e teste. Essa é a estrutura usada nos modelos para predizer os resultados.

4.6.5 Testes com múltiplos modelos

São feitos testes com múltiplos modelos, para se verificar resultados iniciais que possam servir como base para escolha de modelos.

4.6.5.1 Modelo de base

Como modelo de base, cujos resultados podem servir como referência para a geração de outros modelos, é usado um classificador que usa regras simples, do tipo *dummy*.

Ao gerar os modelos para cada um dos alvos (rótulos), se alcança os seguintes resultados:

- *Rótulo_Possui_Erro*: Acurácia de 97% sem balanceamento, e 49% com balanceamento;
- *Rótulo_Erro_Complemento_Docs*: Acurácia de 98%, e 79% com balanceamento;
- *Rótulo_Erro_Complemento_Prancha*: Acurácia de 100% sem balanceamento, e 99% com balanceamento;
- *Rótulo_Erro_Documento_Inválido* : Acurácia de 99% sem balanceamento, e 92% com balanceamento;
- *Rótulo_Expediente_Único_Inválido*: Acurácia de 99% sem balanceamento, e 98% com balanceamento;
- *Rótulo_Indeferimento_Expediente_Único*: Acurácia de 99% sem balanceamento, e 82% com balanceamento.

4.6.5.2 Modelos de diversas famílias de algoritmos

Sobre todos os rótulos, são gerados modelos diversos, obtendo a pontuação de área sob a curva (AUC) e o desvio padrão (STD) de cada modelo, por validação cruzada. Os classificadores testados correspondem a: classificador *dummy*, regressão logística, árvore de decisão, K vizinhos mais próximos, *Naive Bayes* gaussiano, classificador de vetores suporte, floresta aleatória, classificador *XGBoost*, e um classificador de empilhamento, que utiliza a saída dos demais modelos para fazer a predição.

Para *Rótulo_Erro_Complemento_Prancha*, *Rótulo_Erro_Documento_Inválido* e *Rótulo_Expediente_Único_Inválido*, os classificadores não trazem resultados.

Ao gerar os modelos e considerando todos os outros rótulos, se observa que o classificador *Naive Bayes* gaussiano não consegue obter resultados, e os classificadores *dummy* obtêm AUC de 50% e STD de 0%.

Sem usar o balanceamento nas classes, se observa que os demais classificadores, sem considerar os classificadores de empilhamento, obtêm AUC entre 91% e 99% e STD entre 0% e 10%. Os classificadores de empilhamento obtêm AUC de 99% e STD entre 0% e 3%.

Ao usar o balanceamento nas classes de erro de qualquer tipo, se observa que os classificadores, sem considerar o de empilhamento, obtêm AUC entre 97% e 99% e STD entre 0% e 6%. Os classificadores de empilhamento obtêm AUC de 98% a 99% e STD entre 1% e 3%.

4.6.5.3 Testes com modelos de floresta aleatória

Para todos os rótulos, são gerados modelos de classificação usando Floresta Aleatória (Parágrafo 2.5.3.2.3), com e sem otimização nos hiperparâmetros.

Sem utilizar o balanceamento, os resultados da floresta aleatória são para a maior parte dos rótulos, acurácia do modelo de 99% a 100%, métrica de precisão de 87% a 95%, e atributos mais importantes variando entre os referentes a operações, tarefas BPM e documentos, com índices de importância entre 5% e 28%, para todas as florestas aleatórias geradas.

Usando o balanceamento, os resultados mudam: A acurácia fica entre 96% a 100%, a métrica de precisão entre 50% a 100%, e atributos mais importantes variando entre os referentes a operações, tarefas BPM e dados de formulário, sobre os índices de importância que variam entre 4% e 25%, para todas as florestas aleatórias geradas.

Para a maior parte dos rótulos, os melhores hiperparâmetros das árvores são: *max_features: 0.4*, *min_samples_leaf: 1*, *n_estimators: 15* e *random_state: 42*. Ao não usar balanceamento de classes, a exceção é apenas para o rotuloIndeferimentoExpedienteUnico, que altera o valor de *max_features* para *'auto'*. Usando balanceamento, o rótulo *Rótulo_Possui_Erro* altera o valor de *min_samples_leaf* para *'auto'*, e os rótulos *Rótulo_Erro_Documento_Inválido* e *Rótulo_Indeferimento_Expediente_Único* alteram o valor de *max_features* para *'auto'*.

As acurácias alcançadas nos modelos de floresta aleatória com a otimização nos hiperparâmetros variam entre 97% e 100%, sem usar balanceamento, e entre 79% e 99%, ao usar balanceamento.

4.6.6 Geração de modelos baseados em árvore

São gerados modelos baseados em árvore, para verificações e interpretação dos resultados, e inspeções mais aprofundadas sobre os modelos.

4.6.6.1 Árvore de Decisão

4.6.6.1.1 Geração do modelo Para todos os rótulos são gerados modelos de árvore de decisão, que faz as tomadas de decisões sobre o caminho a seguir com base na análise de valores de atributos, até chegar no resultado do alvo a ser previsto.

Os resultados alcançados para cada alvo (rótulo) são:

- *Rótulo_Possui_Erro*: Acurácia de 99,14% sem balanceamento e 97,06% com balanceamento;
- *Rótulo_Erro_Complemento_Docs*: Acurácia de 99,65% sem balanceamento e 96,08% com balanceamento;
- *Rótulo_Erro_Complemento_Prancha*: Acurácia de 100% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Erro_Documento_Inválido*: Acurácia de 99,90% sem balanceamento e 97,06% com balanceamento;
- *Rótulo_Expediente_Único_Inválido*: Acurácia de 99,95% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Indeferimento_Expediente_Único*: Acurácia de 99,80% sem balanceamento

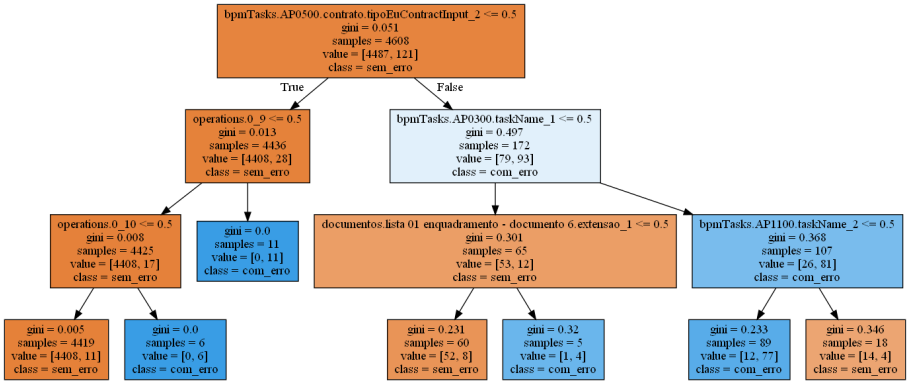
e 95,10% com balanceamento.

Tão logo, se observa que as acurácias mínimas são mais altas ao comparar com os modelos gerados anteriormente.

4.6.6.1.2 Inspeção do modelo O modelo de árvore de decisão gerado para o rótulo referente a erro de qualquer tipo, sem usar balanceamento, é usado para fazer a inspeção.

A Figura 4.7 mostra a estrutura gerada na árvore de decisão, onde se percebem os diferentes caminhos percorridos através dos valores presentes nos atributos, que são binários, 0 ou 1.

Figura 4.7: Estrutura da árvore de decisão



Fonte: Autor

Através da Figura 4.7, se observa que os atributos mais importantes se concentram no topo. E, para este caso de rótulo de qualquer tipo, entre os atributos usados para tomar as decisões, estão apenas os relativos a tarefas BPM, operações e documentos.

A Figura 4.8 demonstra a mesma árvore, porém com histogramas para cada atributo, onde se observa as proporções de rótulos verdadeiros e falsos, refletindo os resultados nas saídas.

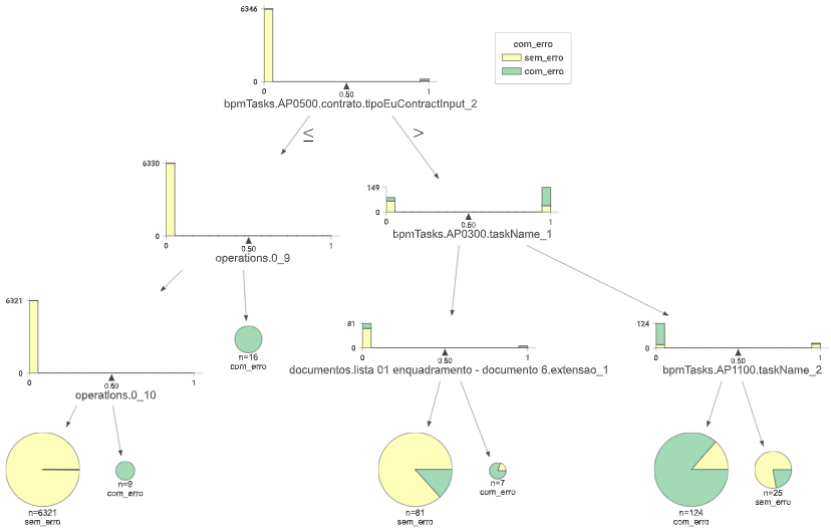
4.6.6.2 Floresta Aleatória

4.6.6.2.1 Geração do modelo São gerados modelos de floresta aleatória para todos os rótulos, consistindo na junção de mais de uma árvore de decisão para compor as saídas.

Os resultados atingidos para cada alvo são:

- *Rótulo_Possui_Erro*: Acurácia de 99,59% sem balanceamento e 96,08% com balanceamento;
- *Rótulo_Erro_Complemento_Docs*: Acurácia de 99,70% sem balanceamento e 98,04% com balanceamento;

Figura 4.8: Árvore de decisão com histogramas



Fonte: Autor

- *Rótulo_Erro_Complemento_Prancha*: Acurácia de 100% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Erro_Documento_Inválido*: Acurácia de 100% com ou sem balanceamento;
- *Rótulo_Expediente_Único_Inválido*: Acurácia de 99,95% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Indeferimento_Expediente_Único*: Acurácia de 99,80% sem balanceamento e 98,04% com balanceamento.

Ao comparar o modelo de floresta aleatória com o modelo de árvore de decisão gerado anteriormente, se observa que houve uma pequena melhora nas acurácias, com obtenção de 100% de acurácia em dois rótulos, sem balanceamento, e em um rótulo, com balanceamento.

4.6.6.2.2 Inspeção do modelo Para fazer a inspeção é usado o modelo de floresta aleatória gerado para o alvo referente a erro de qualquer tipo, sem balanceamento.

Para a floresta aleatória, os atributos considerados mais importantes na previsão do rótulo relacionado a qualquer tipo de erro são os referentes a tarefas BPM, documentos e operações.

4.6.6.3 XGBoost

4.6.6.3.1 Geração do modelo São feitas as gerações de modelos de classificador *XGBoost*. O classificador desse tipo cria uma árvore fraca e, então, a aprimora para reduzir os

erros residuais.

Os resultados do modelo gerado para cada alvo são:

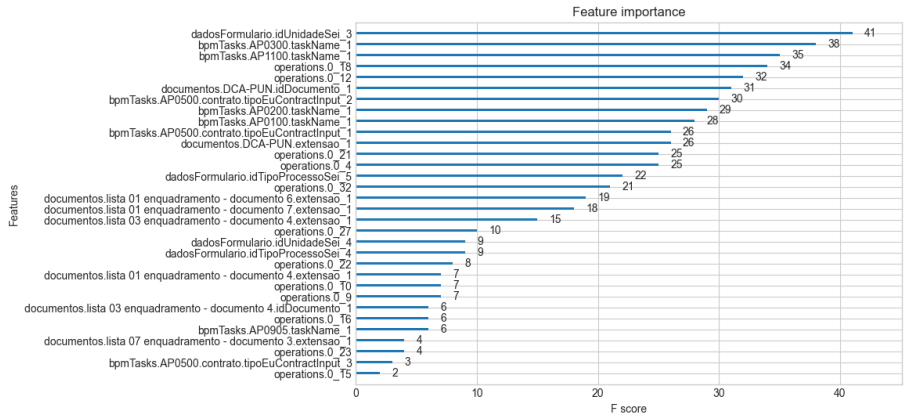
- *Rótulo_Possui_Erro*: Acurácia de 99,65% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Erro_Complemento_Docs*: Acurácia de 99,90% sem balanceamento e 100% com balanceamento;
- *Rótulo_Erro_Complemento_Prancha*: Acurácia de 100% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Erro_Documento_Inválido*: Acurácia de 100% com ou sem balanceamento;
- *Rótulo_Expediente_Único_Inválido*: Acurácia de 99,95% sem balanceamento e 99,02% com balanceamento;
- *Rótulo_Indeferimento_Expediente_Único*: Acurácia de 99,80% sem balanceamento e 99,02% com balanceamento.

Ao comparar com os outros modelos de árvore gerados, se observa uma pequena melhora na acurácia com relação a dois alvos, sem usar balanceamento. Enquanto, ao usar balanceamento, se tem um valor constante para quatro dos rótulos e 100% de acurácia para os outros dois rótulos.

4.6.6.3.2 Inspeção do modelo Assim como nos outros modelos de árvore, em *XGBoost* é possível verificar a importância de atributos.

A Figura 4.9 exibe o gráfico, onde são apresentadas as pontuações pela quantidade de vezes que o atributo aparece nas árvores, para uma interpretação de importância dos atributos, sem usar balanceamento.

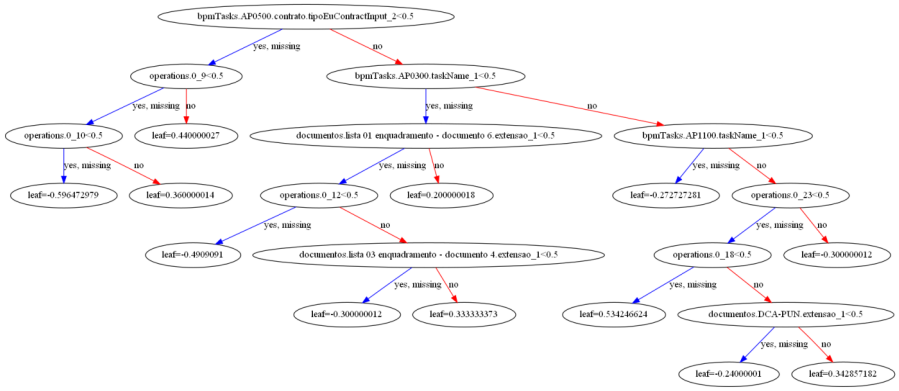
Figura 4.9: Importância de atributos em XGBoost



Fonte: Autor

A Figura 4.10 exibe a estrutura da primeira árvore criada no modelo.

Figura 4.10: Estrutura da árvore em XGBoost



Fonte: Autor

4.6.7 Seleção do modelo

É feita a seleção do modelo, através da verificação dos melhores hiperparâmetros, para sua respectiva otimização.

4.6.7.1 Análise sobre os parâmetros

O modelo de floresta aleatória foi escolhido para compor a solução, pela sua performance em comparação com outros modelos.

O modelo de floresta aleatória possui diversos hiperparâmetros que podem ser alterados.

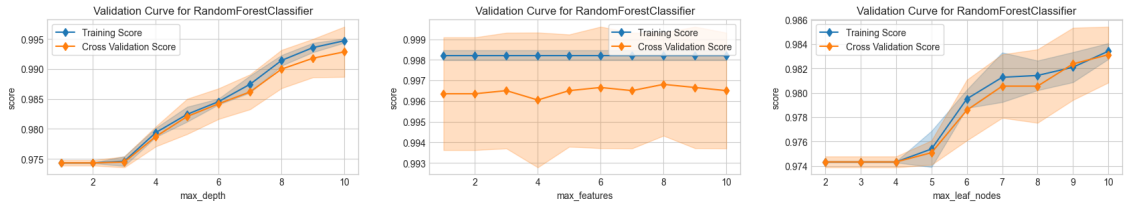
São feitos testes com três hiperparâmetros do classificador de floresta aleatória, são eles:

- *max_depth*: Profundidade de árvore;
- *max_features*: Quantidade máxima de atributos a analisar;
- *max_leaf_nodes*: Número máximo de folhas.

Para esses hiperparâmetros, são fornecidos valor mínimo de 1 e máximo de 11, e gerado um gráfico de curva de validação para cada um, onde é verificado o desempenho do modelo ao modificar o valor do hiperparâmetro em questão, de 1 a 11. O resultado é exibido na Figura 4.11.

É possível perceber pela Figura 4.11 que, conforme se aumenta os valores dos hiperparâmetros, a performance do modelo é melhorada, com exceção de *max_features*,

Figura 4.11: Curvas de validação

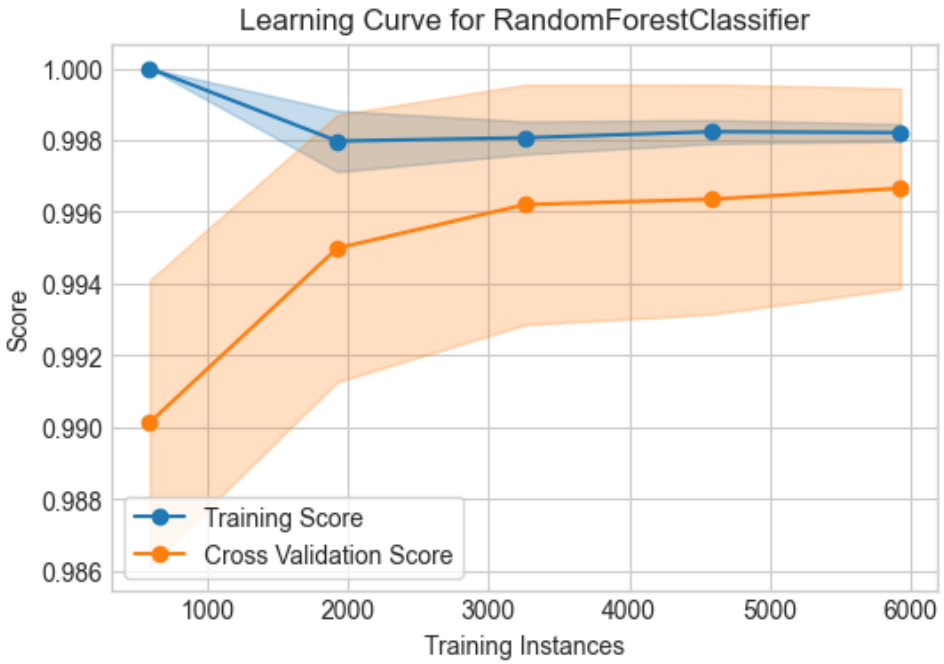


Fonte: Autor

que se mantém em um mesmo patamar, com poucas variações.

A Figura 4.12 exibe uma curva de aprendizagem, que apresenta os desempenhos das instâncias de treino e de teste, à medida que se aumenta a quantidade de amostras.

Figura 4.12: Curva de aprendizado



Fonte: Autor

Pela Figura 4.12, se demonstra que ao chegar em 3000 amostras, o desempenho se mantém quase constante até o final das cerca de 6000 amostras. Portanto a quantidade de amostras não precisa ser aumentada.

4.6.7.2 Verificação de parâmetros e otimização do modelo

O conjunto dos três hiperparâmetros selecionados é verificado com relação aos parâmetros padrão do classificador de floresta aleatória, atribuindo-se o valor máximo escolhido, 11, a cada um e realizando uma comparação de desempenho do modelo com esses parâmetros e os parâmetros padrão.

O resultado (*max_depth: None*, *max_features: 'auto'*, *max_leaf_nodes: None*) demonstra que, na comparação de desempenho, os melhores hiperparâmetros são os padrão.

A partir da verificação de parâmetros, se constata que não é necessário otimizar o modelo, pois o mesmo já se encontra com boa otimização.

4.7 Avaliação do modelo

Esta etapa demonstra as avaliações e as considerações feitas sobre os resultados obtidos na geração do modelo. O passo posterior a esta etapa corresponde a efetiva implantação do modelo.

4.7.1 Considerações sobre as acurácias obtidas

A acurácia obtida no modelo escolhido, como também nos demais modelos gerados é bastante elevada, ao não se utilizar o balanceamento de classes proposto. Entre os modelos testados, alguns ainda melhoraram ainda mais os valores.

Embora a acurácia seja sempre elevada, não se pode dizer que isso é o que torna o modelo bom. Dentre as cerca de 6,5 mil amostras, pouquíssimas correspondem a erros. Isso faz com que as predições tenham uma forte tendência a classificar as amostras como sendo sem erro. E como a maior parte realmente não tem erro, a porcentagem de predições corretas é elevada. Então, para comparação a isso, foi proposto o balanceamento de classes, a fim de ter uma avaliação sem que exista esse viés de classificação para classes sem erro.

Para se ter uma avaliação mais completa sobre o modelo, é necessário verificar outras métricas relacionadas a detecção de falso-positivos, falso-negativos, verdadeiros-positivos e verdadeiro-negativos.

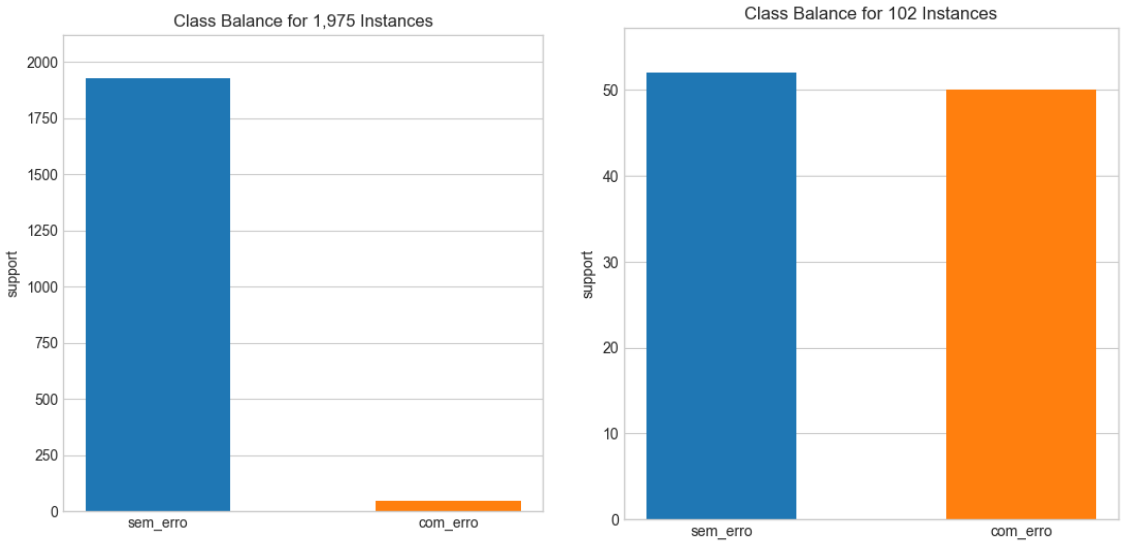
4.7.2 Considerações sobre o uso de balanceamento

Se observa, pelos resultados obtidos, que ao utilizar o balanceamento de classes no rótulo, da forma que foi proposto, os valores de acurácias são menores, porém a medida de AUC, ou área sob a curva, tende a ser maior. E também, as importâncias de atributos

nas árvores parecem ter maior variação.

A Figura 4.13 exibe a diferença entre usar ou não o balanceamento de classes para o rótulo proposto. À esquerda está o gráfico de balanceamento sem fazer uso do balanceamento, e à direita, fazendo uso.

Figura 4.13: Comparação entre balanceamentos



Fonte: Autor

Pela Figura 4.13, se observa nitidamente a aproximação das quantidades entre as duas classes diferentes, causada pelo uso do balanceamento.

4.7.3 Matriz de confusão

A matriz de confusão relata as quantidades de falso-positivos, falso-negativos, verdadeiros-positivos e verdadeiro-negativos.

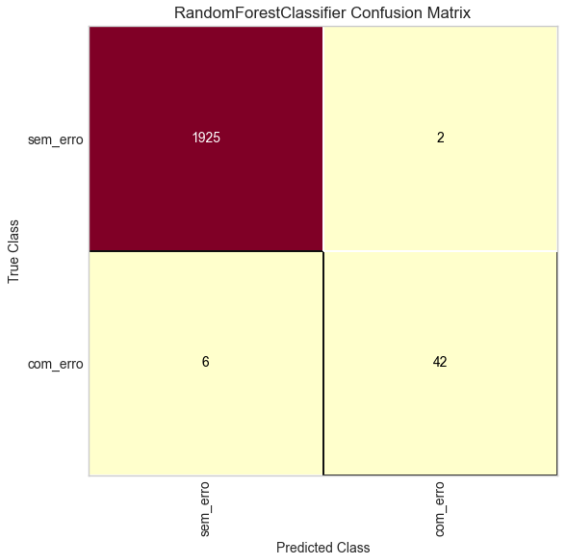
A Figura 4.14 mostra os resultados da classificação do modelo de floresta aleatória utilizado, onde os quadrados superior à esquerda e inferior à direita, são as classificações corretas, ou verdadeiro-positivos e verdadeiro-negativos.

Pela Figura 4.14, se nota que a quantidade de falso-positivos, 2, e falso-negativos, 6, é bem inferior aos outros valores, que são as classificações corretas. As cores exibidas também ajudam a realçar as diferenças nas quantidades.

Se observa que, de 48 erros, foi possível prever 42, e 6 o modelo previu de forma errada que não teria erro. Esses seriam erros que passariam despercebidos pelo modelo. De forma semelhante, de 1927 casos sem erros, o modelo acusou como se 2 deles tivessem erros.

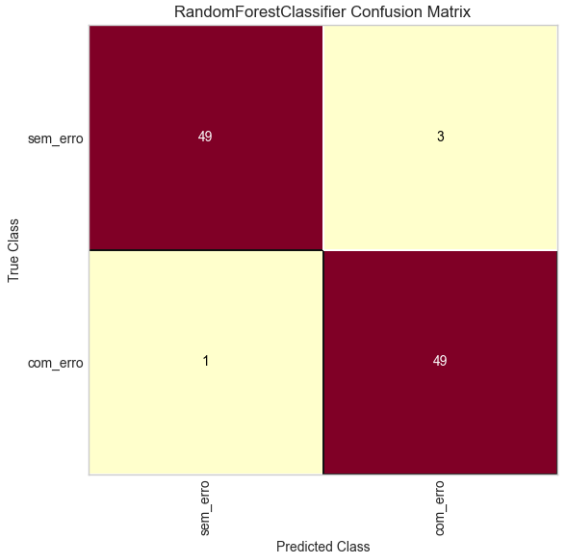
Para comparação, a Figura 4.15, mostra os resultados usando o balanceamento, onde se nota uma melhora com relação a detecção de falso-negativos.

Figura 4.14: Matriz de confusão sem balanceamento



Fonte: Autor

Figura 4.15: Matriz de confusão com balanceamento



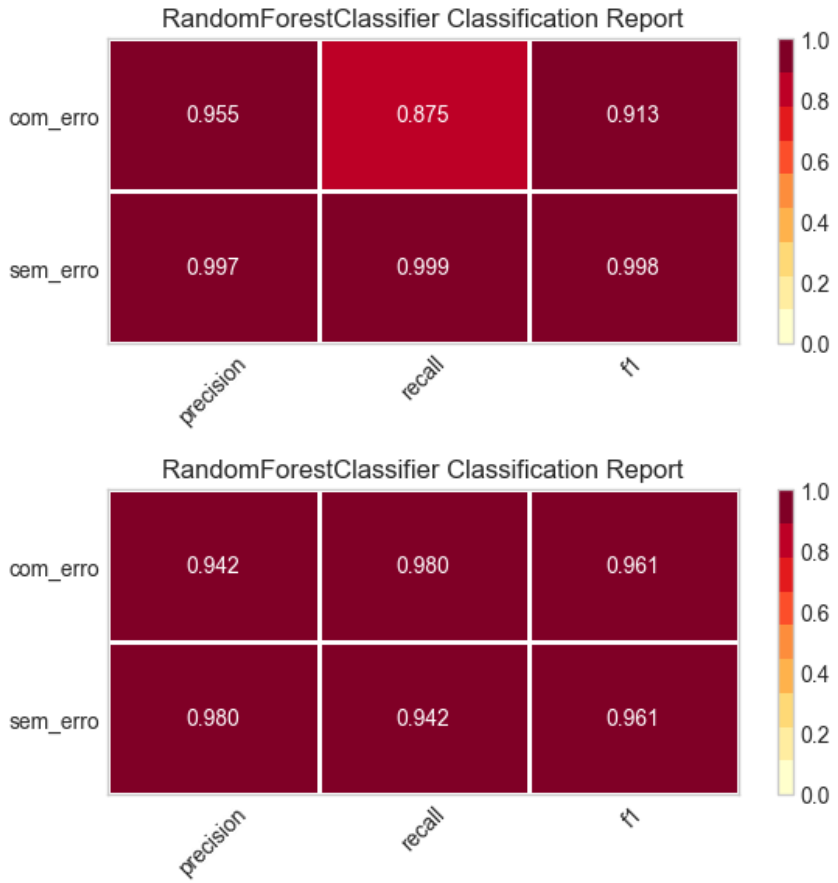
Fonte: Autor

4.7.4 Relatório de classificação

A Figura 4.16 mostra o relatório de classificação do modelo, exibindo métricas relacionadas às relações entre falso-positivos, falso-negativos, verdadeiro-positivos e verdadeiro-negativos.

Na parte de cima da Figura 4.16 é exibido o relatório sem usar o balanceamento de classes. A porcentagem de revocação (*recall*) para casos com erro, 87,5%, é mais baixa se comparada com as outras, afinal o modelo previu 6 falso-negativos. Ainda assim o impacto não se torna tão elevado, considerando as porcentagens das demais métricas.

Figura 4.16: Relatórios de classificação



Fonte: Autor

O mesmo relatório, porém usando o balanceamento, é exibido na parte de baixo da Figura 4.16. Com relação ao uso do balanceamento, se percebe uma melhora na porcentagem de revocação (*recall*).

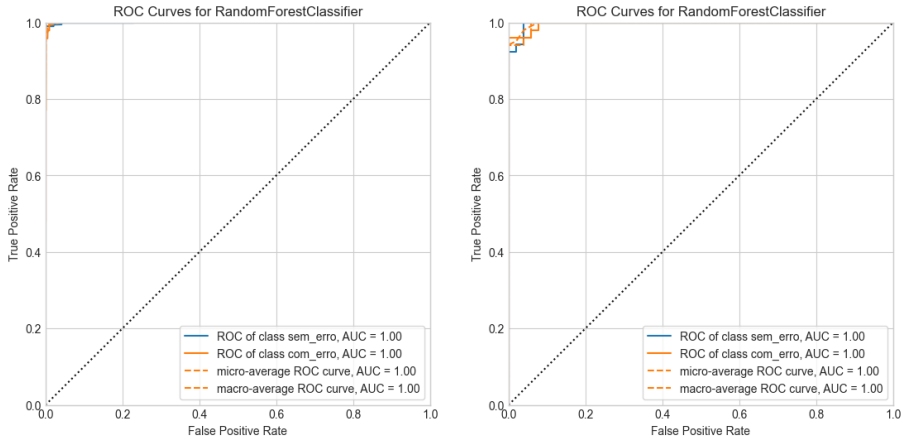
4.7.5 Curva ROC

A AUC, ou área sob a curva, é uma métrica para se avaliar o desempenho do modelo. O valor de AUC obtido é 0,93. Esse valor indica um desempenho elevado.

A Figura 4.17 exibe o gráfico de curva ROC para o modelo, situado à esquerda, o qual demonstra que as pontuações AUC (área sob a curva) para as classes com erros e sem erros são ambas iguais a 1, assim a protuberância em direção ao canto superior esquerdo

é um indicativo de bom desempenho do modelo.

Figura 4.17: Curvas ROC



Fonte: Autor

Entretanto, é importante salientar que a curva ROC pode ser otimista demais com a presença de classes desbalanceadas. A Figura 4.17 mostra o resultado, no gráfico à direita, usando o balanceamento proposto, onde o valor de AUC é ainda mais elevado, obtendo pontuação de 0,96.

4.7.6 Resultado da avaliação e implantação do modelo

Casos de falso-negativos, que são despercebidos pelo modelo, não são considerados críticos para o sistema, podendo ser contornados facilmente, através de verificações no próprio sistema de Licenciamento Urbanístico, pelos usuários.

Como o objetivo central da solução é fornecer um mecanismo de prevenção de erros, como uma forma de otimizar e dar suporte para o sistema de Licenciamento Urbanístico, e considerando que não se trata de uma questão crítica a prevenção de erros para o sistema, o modelo se mostra suficiente para com os objetivos do negócio, dadas as acurácias e resultados obtidos. Mesmo que o modelo tenha alguns erros de predição, ele se mostrou adequado para a solução do problema.

O uso do balanceamento proposto nas classes ajudou a impulsionar a utilização do modelo, ao verificar a consistência nas métricas mesmo sem a utilização de balanceamento.

Com base nas avaliações realizadas, o modelo contempla o propósito da solução de maneira satisfatória, podendo ser implantado.

5 CONCLUSÃO

Através do projeto de aplicação construído, foi possível realizar iterações para se obter diversas percepções sobre os dados, com a utilização dos mais diversos métodos de Ciência de Dados (Subseção 2.4.1). As iterações também possibilitaram executar testes com diferentes modelos de Aprendizado de Máquina (Subsubseção 2.5.3.2), para obter descobertas sobre os funcionamentos e sobre diferentes resultados, nas gerações de modelos.

Foi possível alcançar como resultado a otimização do aspecto de prevenção de erros (Seção 2.3) no sistema de Licenciamento Urbanístico, dado o resultado das avaliações do modelo de Aprendizado de Máquina.

O trabalho trouxe uma importante contribuição na questão da automatização de soluções para o sistema de Licenciamento Urbanístico de Porto Alegre, com a opção de utilização dentro dos projetos de *software* da Procempa, que estão em constante evolução. Pelos resultados, se observa que o trabalho contemplou seu propósito de auxílio ao projeto para o qual foi desenvolvido.

As predições finais, bem como as avaliações, foram executadas sobre o objetivo de erro correspondente a qualquer tipo. Novas iterações podem compor os resultados para os demais erros, para se obter análises específicas sobre estes.

Novas iterações também implicam em obter novos conjuntos de dados, com atributos diferentes, ou novos, na medida em que o sistema de Licenciamento Urbanístico avança em seu desenvolvimento, para assim adaptar o projeto à coleta e uso destes.

Como próxima opção de oportunidade de melhoria do sistema de Licenciamento Urbanístico, com a utilização de Ciência de Dados e Aprendizado de Máquina, pode ser elaborado o projeto sobre a oportunidade de detecção de fraudes (Capítulo 1), dada a relação com a oportunidade de prevenção de erros, já explorada, no sentido de que um erro causado por um usuário pode ser em função de uma tentativa de fraude.

Da mesma forma, trabalhos futuros envolvem aproveitar os conhecimentos adquiridos no processo de Ciência de Dados e aplicação de Aprendizado de Máquina sobre o conjunto de dados, para construção de soluções nos demais problemas elencados: recomendação de ações dentro do processo e extração de relatórios inteligentes (Capítulo 1).

Com a implantação do modelo, é possível disponibilizá-lo através de uma API (*Application Programming Interface*), que outros sistemas poderão utilizar.

Com a ideia de automatizar os processos de *workflow* com relação aos resultados

dos modelos, a ideia e conceito de aplicação RPA (Subseção 2.2.2) pode ser aproveitada, com relação ao modelo gerado, podendo detectar um erro ao utilizar a API do modelo disponibilizada, como um componente dentro do sistema maior, de forma integrada. Essa ideia é utilizada como forma de detectar os problemas de erro dentro do fluxo BPM, já existente.

Também, de modo semelhante, a API pode ser utilizada em sistemas de BI (*Business Intelligence*) para geração de dados e relatórios importantes para a gestão.

Por fim, o modelo pode ser integrado com diversos outros sistemas de informação, visando a automatização de soluções possíveis no contexto de Licenciamento Urbanístico.

REFERÊNCIAS

CADY, F. **The Data Science Handbook.** [S.l.]: Wiley, 2017. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=edsebk&AN=1456617&lang=pt-br&site=eds-live&scope=site&authtype=ip,guest&custid=s5837110&groupid=main>. ISBN 9781119092940.

DEEK, F. P.; MCHUGH, J. A. **Open Source : Technology and Policy.** [S.l.]: Cambridge University Press, 2008. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=214426&lang=pt-br&site=eds-live&scope=site&authtype=ip,guest&custid=s5837110&groupid=main>. ISBN 9780521881036.

ETHEM, A. **Machine Learning : The New AI.** [S.l.]: The MIT Press, 2016. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1369420&lang=pt-br&site=eds-live&scope=site&authtype=ip,guest&custid=s5837110&groupid=main>. (MIT Press Essential Knowledge Series). ISBN 9780262529518.

FACELI, K. et al. **Inteligência artificial : uma abordagem de aprendizado de máquina.** [S.l.]: LTC, 2021. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=edsmib&AN=edsmib.000021044&lang=pt-br&site=eds-live&scope=site&authtype=ip,guest&custid=s5837110&groupid=main>. ISBN 9788521637349.

FILGUEIRAS, L. V. L.; RODRIGUES, L. V. Modelagem de tarefas para simulação do desempenho humano em erro. In: **Proceedings of VII Brazilian Symposium on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 2006. (IHC '06), p. 21–24. ISBN 1595934324. Available from Internet: <https://doi.org/10.1145/1298023.1298054>.

GACOVSKI, Z. **Soft Computing and Machine Learning with Python.** [S.l.]: Arcler Press, 2019. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2013960&lang=pt-br&site=eds-live&scope=site&authtype=ip,guest&custid=s5837110&groupid=main>. ISBN 9781773616230.

GitHub, Inc. **XGBFIR**. 2021. Disponível em: <https://pypi.org/project/xgbfir/>. Acesso em: 17 julho 2021.

GIUGNO, N. B.; MELLO, O. d. S. O. **Grandes empreendimentos urbanos : o desempenho municipal qualificando a cidade.** [S.l.: s.n.], 2007. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=cat07377a&AN=sabi.000637627&lang=pt-br&site=eds-live&scope=site>.

GéRON, A. **Mãos à Obra: Aprendizado de Máquina com Scikit-Learn e TensorFlow: Conceitos, Ferramentas e Técnicas Para a Construção de Sistemas Inteligentes.** Rio de Janeiro: Alta Books, 2019. ISBN 9788550809021.

HARRISON, M. **Machine Learning – Guia de Referência Rápida: Trabalhando com dados estruturados em Python.** São Paulo: Novatec Editora, 2020. ISBN 9788575228180.

JONER, H.; SANTOS, N. S. d. O.; SILVA, C. E. S. d. O. **Inferência preditiva geoespacial da criminalidade em Porto Alegre : uma abordagem de aprendizado de**

máquina. [S.l.: s.n.], 2020. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=cat07377a&AN=sabi.001122428&lang=pt-br&site=eds-live&scope=site&authtype=ip,guest&custid=s5837110&groupid=main>.

MARTINS, P. et al. Using machine learning for cognitive robotic process automation (rpa). **2020 15th Iberian Conference on Information Systems and Technologies (CISTI), Information Systems and Technologies (CISTI), 2020 15th Iberian Conference on**, p. 1 – 6, 2020. ISSN 978-989-54659-0-3. Available from Internet: <https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.9140440&lang=pt-br&site=eds-live&scope=site>.

MCGINNIS, W. **Category Encoders**. 2016. Disponível em: <https://contrib.scikit-learn.org/category_encoders/#>. Acesso em: 17 julho 2021.

MongoDB, Inc. **MongoDB**. 2021. Disponível em: <https://www.mongodb.com/pt-br>. Acesso em: 17 julho 2021.

Mozilla and individual contributors. **JavaScript**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>. Acesso em: 17 julho 2021.

NAKANO, R. S. **Welcome to Scikit-plot's documentation!** 2017. Disponível em: <https://scikit-plot.readthedocs.io/en/stable/#>. Acesso em: 17 julho 2021.

NumFOCUS. **pandas**. 2021. Disponível em: <https://pandas.pydata.org/>. Acesso em: 06 julho 2021.

NumPy. **NumPy**. 2021. Disponível em: <https://numpy.org/>. Acesso em: 06 julho 2021.

Opensource.org. **Open Source Initiative**. 2021. Disponível em: <https://opensource.org/>. Acesso em: 20 julho 2021.

OZDEMIR, S. **Principles of Data Science.** [S.l.]: Packt Publishing, 2016. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1441463&lang=pt-br&site=eds-live&scope=site>. ISBN 9781785887918.

PARR, T.; LAPUSAN, T.; GROVER, P. **dtreeviz: Decision Tree Visualization**. 2021. Disponível em: <https://github.com/parrt/dtreeviz>. Acesso em: 17 julho 2021.

Prefeitura Municipal de Porto Alegre. **Decreto Nº 12.715**. 2000. Disponível em: <http://www2.portoalegre.rs.gov.br/cgi-bin/nph-brs?s1=000023808.DOCN.&l=20&u=%2Fnetahtml%2Fsirel%2Fsimples.html&p=1&r=1&f=G&d=atos&SECT1=TEXT>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Decreto Nº 12.923 de 25 de setembro 2000**. 2000. Disponível em: <http://lproweb.procempa.com.br/pmpa/prefpoa/smov/usu_doc/dc12923.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Lei Complementar Nº 434/99**. 2007. Disponível em: <http://lproweb.procempa.com.br/pmpa/prefpoa/spm/usu_doc/3_-_texto_lc_434.pdf>. Acesso em: 05 março 2021.

Prefeitura Municipal de Porto Alegre. **Decreto 18.623, de 24 de abril de 2014**. 2014. Disponível em: <http://lproweb.procempa.com.br/pmpa/prefpoa/spm/usu_doc/decreto_18623_republicado.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Decreto Nº 18.828, de 24 de outubro de 2014**. 2014. Disponível em: <http://dopaonlineupload.procempa.com.br/dopaonlineupload/1254_ce_109119_1.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Decreto Nº 18.886, de 18 de dezembro de 2014**. 2014. Disponível em: <http://dopaonlineupload.procempa.com.br/dopaonlineupload/1300_ce_113900_1.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Lei Complementar Nº 806, de 27 de dezembro de 2016**. 2016. Disponível em: <http://lproweb.procempa.com.br/pmpa/prefpoa/spm/usu_doc/lc_806.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Decreto Nº 19.741, de 12 de maio de 2017**. 2017. Disponível em: <http://dopaonlineupload.procempa.com.br/dopaonlineupload/2080_ce_190631_1.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Escritório de Licenciamento unifica sistema de consultas**. 2018. Disponível em: <http://www2.portoalegre.rs.gov.br/edificapoa/default.php?p_noticia=999198044&ESCRITORIO+DE+LICENCIAMENTO+UNIFICA+SISTEMA+DE+CONSULTAS>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Manual do Protocolo Setorial : Lista de Documentos e Orientações Gerais**. 2020. Disponível em: <http://lproweb.procempa.com.br/pmpa/prefpoa/edificapoa/usu_doc/manualprot.pdf>. Acesso em: 15 julho 2021.

Prefeitura Municipal de Porto Alegre. **Aprovação e licenciamento de edificações : Licenças e Licenciamento Expresso**. 2021. Disponível em: <https://prefeitura.poa.br/carta-de-servicos/licencas-e-licenciamento-expresso>. Acesso em: 03 março 2021.

Prefeitura Municipal de Porto Alegre. **Procempa : Quem Somos**. 2021. Disponível em: <https://prefeitura.poa.br/procempa/quem-somos>. Acesso em: 15 julho 2021.

Presidência da República: Casa Civil: Subchefia para Assuntos Jurídicos. **Lei Nº 10.257, de 10 de julho de 2001**. 2001. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/leis_2001/l10257.htm>. Acesso em: 03 março 2021.

Project Jupyter. **Project Jupyter**. 2021. Disponível em: <https://jupyter.org/>. Acesso em: 06 julho 2021.

Python Software Foundation. **io : Core tools for working with streams**. 2021. Disponível em: <https://docs.python.org/3/library/io.html>. Acesso em: 17 julho 2021.

Python Software Foundation. **mlxtend**. 2021. Disponível em: <https://pypi.org/project/mlxtend/>. Acesso em: 17 julho 2021.

Python Software Foundation. **os : Miscellaneous operating system interfaces**. 2021. Disponível em: <https://docs.python.org/3/library/os.html>. Acesso em: 17 julho 2021.

Python Software Foundation. **Pandas Profiling**. 2021. Disponível em: <https://pypi.org/project/pandas-profiling/>. Acesso em: 06 julho 2021.

Python Software Foundation. **pickle : Python object serialization**. 2021. Disponível em: <https://docs.python.org/3/library/pickle.html>. Acesso em: 17 julho 2021.

Python Software Foundation. **pydotplus**. 2021. Disponível em: <https://pypi.org/project/pydotplus/>. Acesso em: 17 julho 2021.

Python Software Foundation. **Python**. 2021. Disponível em: <https://www.python.org/>. Acesso em: 06 julho 2021.

Python Software Foundation. **random : Generate pseudo-random numbers**. 2021. Disponível em: <https://docs.python.org/3/library/random.html>. Acesso em: 17 julho 2021.

Python Software Foundation. **rfpimp**. 2021. Disponível em: <https://pypi.org/project/rfpimp/>. Acesso em: 17 julho 2021.

Python Software Foundation. **Sweetviz**. 2021. Disponível em: <https://pypi.org/project/sweetviz/>. Acesso em: 06 julho 2021.

RUSSELL, N.; AALST, W. v. d.; HOFSTEDE, A. T. **Workflow patterns : the definitive guide.** [S.l.]: MIT Press, 2008. Disponível em: <https://search.ebscohost.com/login.aspx?direct=true&db=cat07377a&AN=sabi.001056398&lang=pt-br&site=eds-live&scope=site>. (Information systems). ISBN 9780262329408.

scikit-learn. **scikit-learn : Machine Learning in Python**. 2021. Disponível em: <https://scikit-learn.org/stable/>. Acesso em: 06 julho 2021.

SPSS Inc. **CRISP-DM 1.0 : Step-by-step data mining guide**. 2000. Disponível em: <https://moodle.inf.ufrgs.br/pluginfile.php/142998/mod_resource/content/2/CRISPWP-0800.pdf>. Acesso em: 20 julho 2021.

The IPython Development Team. **IPython Documentation**. 2021. Disponível em: <https://ipython.readthedocs.io/en/stable/#>. Acesso em: 17 julho 2021.

The Matplotlib development team. **Matplotlib: Visualization with Python**. 2021. Disponível em: <https://matplotlib.org/>. Acesso em: 06 julho 2021.

The scikit-yb developers. **Yellowbrick: Machine Learning Visualization**. 2021. Disponível em: <https://www.scikit-yb.org/en/latest/>. Acesso em: 06 julho 2021.

WASKOM, M. **seaborn**. 2021. Disponível em: <https://seaborn.pydata.org/>. Acesso em: 06 julho 2021.

xgboost developers. **XGBoost Documentation**. 2020. Disponível em: <https://xgboost.readthedocs.io/en/latest/#>. Acesso em: 17 julho 2021.