UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

Tiago Pedro da Silva Sitonio

Sistema de Aprendizado de Máquina para Predição do Tempo de Esforço de Tarefas de Desenvolvimento de Software

Recife

2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

S623s Sitonio, Tiago Pedro da Silva Sitonio
Sistema de Aprendizado de Máquina para Predição do Tempo de Esforço de Tarefas de Desenvolvimento de Software / Tiago Pedro da Silva Sitonio Sitonio. - 2021.
58 f. : il.

Orientador: Cleviton Vinicius Fonseca Monteiro.
Inclui referências.

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Sistemas da Informação, Recife, 2021.

1. Estimativa. 2. Esforço. 3. AutoML. 4. Tokenização. I. Monteiro, Cleviton Vinicius Fonseca, orient. II. Título

CDD 004

Tiago Pedro da Silva Sitonio

Sistema de Aprendizado de Máquina para Predição do Tempo de Esforço de Tarefas de Desenvolvimento de Software

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada em: 14 de Dezembro de 2021.

BANCA EXAMINADORA

Cleviton Monteiro (Orientador)
Departamento de Estatísitca e Informática
Universidade Federal Rural de Pernambuco

Rinaldo Lima
Departamento de Computação
Universidade Federal Rural de Pernambuco

À Deus, minha família e todos que me ajudaram para que este trabalho se torne realidade.

Agradecimentos

Primeiramente agradeço a Deus por ter me dado forças para concluir mais esta etapa importante da minha vida, pois tudo na vida devo a Ele que me sustenta a cada dia. Também quero agradecer em especial à minha família e à minha namorada por ter estado comigo, pela paciência nos momentos difíceis e por ter me incentivado durante esses anos de Graduação.

Agradeço grandemente a meu orientador: Cleviton Monteiro, Rinaldo Lima, Abílio, todos os professores, e à coordenação do curso de Sistemas de Informação da UFRPE. Vocês foram cruciais em possibilitar a chegada neste objetivo por meio de sua grande competência, por sempre estarem disponíveis mesmo fora do horário de aula, e pela qualidade nas atividades de ensino e apoio.

Eu nunca chegaria ao final deste curso sem o apoio aos meus amigos que fizeram parte deste sonho dos quais serei sempre grato. Agradeço também aos veteranos que nos ajudavam naquelas cadeiras mais complicadas. Este curso foi construído pelo trabalho em equipe, seja pelos tantos projetos em grupo, como pelas tantas ajudas antes das provas com as leituras de revisões como também ao final da aula com as trocas de dúvidas que contribuíram para a construção do conhecimento.

Muito obrigado a esta instituição por ter fornecido excelentes espaços ao estudante e pelo suporte emocional durante a pandemia. Este Trabalho de Conclusão de Curso e meu próprio curso nunca seriam concluídos sozinho, pois, com problemas de ansiedade, considero que o apoio recebido foi fundamental para o alcance deste tão desejado objetivo. Deixo minha mensagem de sempre valorizar seu próximo, não importando a situação, visto que, em algum tempo você ou este precisará de uma ajuda didática, ou até maior, uma ajuda humana. A todos que prestam algum suporte aos alunos da UFRPE, pois cada um de vocês é fundamental na construção de um mundo melhor.

“Por isso mesmo, aplicando todo o vosso esforço, acrescentai a virtude à vossa fé e o conhecimento à virtude,”

(2 Pedro 2:5)

Resumo

A estimativa de esforço é uma das principais métricas para o planejamento e gerenciamento do processo de desenvolvimento de software, pois proporciona auxílio na previsão de custos e prazos para realização de um projeto. Em consideração a isso, este trabalho teve como objetivo realizar uma análise do processo de construção de um modelo de Aprendizado de Máquina com base na metodologia CRISP-DM, utilizando Algoritmo de Aprendizado de Máquina Automatizado(*AutoML*) para encontrar o melhor algoritmo de regressão com o objetivo de encontrar a estimativa de tempo de determinada atividade. Para esta análise, dados de atividades realizadas por 29 empresas foram utilizados. O banco de dados é constituído por diferentes tipos de dados como, por exemplo, dado Numérico em formato de Linguagem Natural para descrever as atividades. Por causa disso foi realizado o processo de *Tokenização* a fim de transformar em dados totalmente regressivos para execução dos algoritmos. Em conjunto a isto, métodos de análise dos dados, pré-processamento, métodos de afunilamento como Seleção de *Feaures*, Alteração de Pesos e Combinação de Colunas serão aplicados para realizar análises do banco de dados. Este projeto foi desenvolvido através da linguagem de programação *Python* com apoio das suas bibliotecas, dentre elas a biblioteca *Pandas* para manipulação e análise de dados e *Scikit-learn* para acesso a algoritmos de Aprendizado de Máquina. Os resultados obtidos e avaliados apontam que o tratamento individual para cada empresa com pré-processamento e construção do modelo de algoritmo de previsão devem ser levados em consideração para encontrar os melhores resultados de estimativa de esforço por meio dos algoritmos.

Palavras-chave: Estimativa de esforço, Aprendizado de Máquina Automatizado, Engenharia de Software, Tokenização.

Abstract

Effort estimation is one of the main metrics for planning and managing the software development process, as it provides an aid in the cost and deadline forecast for the project's realization. In consideration, this paper aims to perform an analysis of the process of building a Machine Learning model based on the CRISPDM methodology, using Automated Machine Learning Algorithm (AutoML) to find the best regressive algorithm to estimate the time of a given activity. For this analysis, data from activities performed by 29 companies will be used. The database consists of different types of data such as Numeric and Natural Language format to describe the activities, because of this the Tokenization process was performed to transform into fully regressive data to run the algorithms. In conjunction with this, data analysis methods, pre-processing, tunneling methods such as Feature Selection, Weight Change and Column Combination will be applied to perform database analysis. This project was developed through the Python programming language with the support of its libraries, among them the Pandas library for data manipulation and analysis and Scikit-learn for access to Machine Learning algorithms. The results obtained and evaluated point out that the individual treatment for each company with pre-processing and construction of the prediction algorithm model must be taken into account to find the best results of effort estimation through the algorithms.

Keywords: Effort Estimation, Automated Machine Learning, Software Engineering, Tokenization.

Lista de tabelas

Sumário

1 Introdução

Este capítulo tem como objetivo expor o trabalho de uma forma objetiva e resumida, de forma que o trabalho seja apresentado ao leitor enfatizando o fomento ao qual foi produzido. Além disso, serão apresentados os objetivos e o modo em que o trabalho está organizado.

1.1 Apresentação

O projeto é definido por (INSTITUTE, 2018) como um esforço temporário empreendido para criar um produto, serviço ou resultado único. Para um projeto de desenvolvimento de software o esforço que é gasto para sua realização pode depender de vários fatores tornando difícil seu gerenciamento. Entre eles a falta de gerenciamento de tempo pode fazer com que entregas sejam atrasadas e até mesmo deixadas de lado, além de causar falhas de atendimentos que de fato impactam na visão que o cliente tem da qualidade da organização (ZANETTI, 2017).

”Evitar a perda de gastos ou problemas de entrega e obter o sucesso de qualquer projeto de software depende principalmente da precisão da estimativa de esforço”(Charette, 2005). Existem três formas de realizar a estimativa de esforço: a predição por meio de especialista, a predição realizada através do aprendizado de máquina, e a predição realizada com a junção das duas. Para a primeira um bom resultado da estimativa depende de maneira maior de um especialista da área em relação aos dados históricos. A segunda e a terceira necessitam também de um apoio realizado por especialista, mas em menor grau de importância dependendo da disposição dos dados históricos. Para criar a estimativa de esforço para o desenvolvimento de software é necessário reunir métricas dos projetos atuais e históricos a fim de traçar semelhanças e analogias para poder fazer previsões sobre o projeto atual (NAJADAT; ALSMADI; SHBOUL, 2012). A construção dos dados é de grande importância para a qualidade da estimativa de esforço. O algoritmo de aprendizado de máquina depende destes dados para realizar o treinamento do algoritmo que fará o reconhecimento de padões através das funções de matemática e estatística. Neste trabalho será aplicada a segunda abordagem com métodos de regressão que serão abordados no próximo capítulo. Utilizaremos uma base de dados que contêm informações de várias atividades as quais, o total de horas gastas em uma atividade será o valor objetivo do algoritmo de Aprendizado de Máquina.

1.2 Objetivos

Este trabalho tem como objetivo principal realizar um estudo sobre o desenvolvimento de um sistema de predição com o uso do Aprendizado de Máquina Automatizado (AutoML) baseado em um determinado banco de dados de empresas reais, comparando-o com o resultado do modelo proposto por (SHUKLA; KUMAR, 2021). Dentro desse conjunto de algoritmos, de forma mais específica, será utilizado o algoritmo de regressão selecionado através de algoritmo de AutoML para previsão do tempo de esforço a ser realizado por determinada atividade de desenvolvimento de software.

Para alcançar o objetivo principal, três objetivos específicos foram executados: o primeiro no qual consiste na avaliação e pré-processamento do banco de dados que é composto por atividades de empresas, o segundo é uma investigação do impacto e importância dos processos de Combinação de Colunas e Seleção de Features, o terceiro é a análise comparativa do processo de construção de um modelo a partir da junção de diversas empresas diferentes em relação à cosntrução de um modelo baseado apenas em uma única empresa.

1.3 Organização do trabalho

Este trabalho está dividido em 6 capítulos: o segundo capítulo, debaterá o referencial teórico utilizado para construção do trabalho; o terceiro abordará os trabalhos relacionados, trabalhos que realizam avaliações sobre a forma de realizar a estimativa de esforço por meio do aprendizado de máquina de acordo com seu contexto de banco de dados; o quarto descreverá o processo de experimentação e realização do projeto; o quinto consistirá na apresentação do processo de experimentação do modelo, bem como os resultados encontrados; por último, o sexto capítulo tratará das conclusões do trabalho.

2 Fundamentação Teórica

2.1 Estimativa de Software

A estimativa é um processo utilizado para predizer o esforço necessário para uma etapa, ou projeto inteiro. O esforço estimado é utilizado para definir cronogramas, construir orçamento e planejar entregas (AVILA, 2019). Sob esse contexto, empresas de software estão interessadas em determinar o custo de desenvolvimento de software nos estágios iniciais para controlar e planejar tarefas, riscos, orçamentos e cronogramas de software (NAJADAT; ALSMADI; SHBOUL, 2012). Além de importante, isso apresenta ser uma tarefa muito difícil para os gerentes de projeto preverem o custo e o esforço necessários nos estágios prematuros de planejamento (KUMAR et al., 2020). Para o desenvolvimento de software o orçamento excessivo, atraso na entrega e o não alcance das expectativas de um projeto se deve ao fato de não se conhecer o suficiente o que está sendo desenvolvido. De acordo com o estudo (CHARETTE, 2005), 66% dos projetos de software analisados apresentaram atraso no cronograma ou aumento do orçamento que foi anteriormente previsto, o que sucitou na perda significativa de recursos para os envolvidos. Em outro estudo mais recente (USMAN; MENDES; BÖRSTLER, 2015), foram detalhados as formas em que um grupo de empresas realizam a estimação de esforço. Dados coletados de 60 praticantes de metodologias ágeis em sua organização e gerência de 16 países diferentes apontam que as estimativas de Desenvolvimento de software Ágil frequentemente praticadas são Planning poker (COHN, 2005) (63%), analogia (47%) e julgamento de especialistas (38%). Dentre estas empresas participantes, 52% dos entrevistados acreditam que suas estimativas de esforço, em média, estão sub/superestimadas por um erro de 25% ou mais. Assim, *Falhas na estimativa de desenvolvimento de software são influenciados por diversos fatores, entre eles a falta de informação do conhecimento e experiência de experts do passado, projetos de estimativa de software baseados apenas na classificação humana onde não são repetíveis e afetados pela mudança e problemas com os dados do projeto incompletos, inconsistentes, incertos e confusos* (SEHRA et al., 2017). Dados estes problemas, é necessário atuar em diferentes meios de executar a estimativa de esforço.

Outra forma para tanto é o uso de aprendizado de máquina para obter a estimativa que de acordo com (WEN et al., 2012) é baseado em técnicas de inteligência artificial para prever o esforço de desenvolvimento. Quanto ao processo de estimação de software, estabelecer um processo no início do ciclo de vida é crucial, pois resultará em maior precisão e credibilidade das estimativas e uma compreensão mais clara dos

que influencia no custo de desenvolvimento de software (AGARWAL et al., 2001). Sob este contexto, é necessário acesso ao máximo de informações possíveis para realizar uma estimativa de melhor qualidade, tanto de informações do projeto atual como o histórico de outras atividades. A fidelidade nos dados históricos e dados do projeto atual geram melhores resultados. Os artigos: *Software effort estimation using machine learning methods* (BASKELES; TURHAN; BENER, 2007), *Systematic literature review of ensemble effort estimation* (IDRI; HOSNI; ABRAN, 2016) e *Predicting software projects cost estimation based on mining historical data* (NAJADAT; ALSMADI; SHBOUL, 2012) são exemplos de estudos que obtiveram resultados satisfatórios sobre o uso de ferramentas de AM para predição de esforço de software em diferentes contextos e necessidades.

2.2 CRISP-DM

O método de *CRoss Industry Standard Process for Data Mining* (CRISP-DM) será abordado como fundamento para execução do projeto. O CRISP-DM consiste numa série de etapas semelhantes a um conjunto de direcionamentos para auxiliar no processo de planejamento, organização e implementação do projeto de mineração de dados ou aprendizado de máquina (SALTZ; HOTZ, 2021).

Sheares 2000, desenvolveu um *framework* para o desenvolvivmento de projeto de aprendizado de máquina composto por 6 fases que descrevem o processo completo de um projeto de ciência de dados. As etapas do processo percorrem desde o entendimento do modelo de negócio até a entrega do sistema. A seguir são apresentadas as 6 etapas do CRISP-DM, descrevendo do que se trata, a entrada e saída do estado do produto em cada uma delas.

- **Entendimento dos Negócios:** compreender os objetivos e requisitos do projeto é o propósito desta fase. Um estudo bem formado sobre a necessidade do cliente, avaliabilidade dos recursos, definição dos critérios de sucesso, riscos e benefícios a serem percorridos, planejamento do projeto e quais ferramentas serão usadas são considerados os fundamentos de qualquer trabalho. Tendo todas estas partes bem estabelecidas é prosseguido para fase de entendimento dos dados.

- **Entendimento dos dados:** nesta etapa é onde se realiza a coleta inicial dos dados, o reconhecimento das falhas dos dados, o primeiro entendimento dos dados, bem como a criação de sua descrição. Os dados podem vir de maneira superficial ou não tratados corretamente, por isso é necessário *feedback* sobre seu estado. Na maioria dos casos é realizado a volta para a fase do Entendimento dos Negócios com instuito de realizar ajustes dos parâmetros preestabelecidos.

- **Preparação dos dados:** tem como objetivo principal criar a versão final da

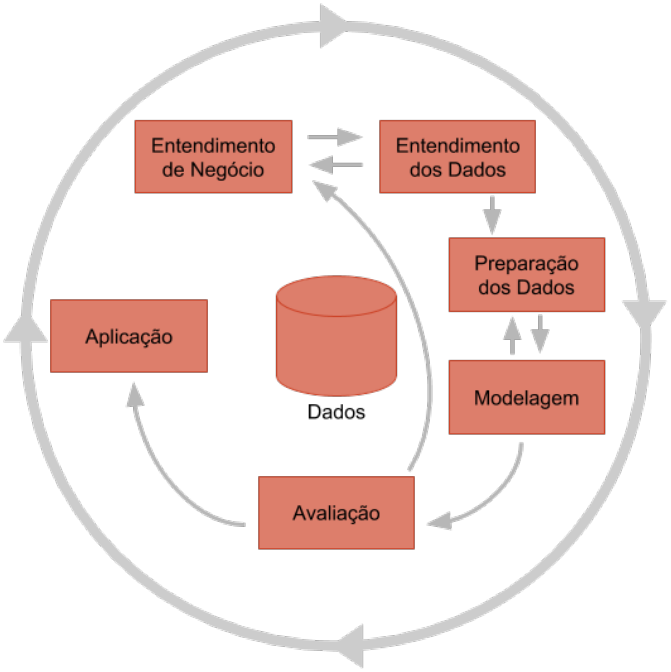

Figura 1 – Fonte: (MOURA, 2019)

base de dados que será utilizada na etapa de Modelagem. Para alcançar os melhores resultados tanto de Mineração de Dados como de AM, é necessário cumprir com etapas de preparação, sendo elas: seleção de quais dados devem ser usados baseado nos objetivos determinados na primeira fase; limpar os dados desnecessários de forma assídua e detalhada para que os resultados sejam aceitáveis; criar dados mais leves e simples derivados dos dados iniciais; integrar dados importantes para análise vindos de diferentes fontes a fim de agregar novas informações; e a formatação de dados como por exemplo transformar dados escritos em números para possibilitar operações matemáticas. A partir desta etapa deve-se garantir a qualidade dos dados a serem utilizados como entrada para a fase de Modelagem por isso é preciso considerar uma maior quantidade de tempo.

- **Modelagem:** técnicas de modelagem são aplicadas com os dados obtidos na fase anterior. Os modelos de AM são selecionados, assim como os dados de treino e teste, modelos de avaliação e diferentes exemplos de paradigmas.

- **Avaliação:** é importante que os modelos construídos na etapa anterior sejam avaliados e revisados para assegurar que os critérios estão sendo atendidos ou não. No caso negativo, é necessário retornar as discurssões de metas e planejamento do negócio. No final, deve-se decidir de que forma serão usados os resultados obtidos, revisão dos processos e os próximos passos.

- **Implantação:** esta última fase pode ser considerada a mais complexa. É nela que são definidos e documentados a forma na qual o algoritmo de produção será inserido, tal como a manuntenção contínua do projeto pós implantação. Relatórios de re-

trospectiva de todo trabalho devem ser criados com o objetivo de coletar informações do que foi um sucesso ou do que deu completamente errado para servir de orientações para os próximos trabalhos.

De acordo com a Figura 1 é possível observar o formato cíclico que envolve todas as fases do CRISP-DM. Ao finalizar a Implantação o projeto não deve ser concluído e, como descrito por essa fase de implantação, a manuntenção contínua é simbolizada pelas setas circulares na borda da figura. Uma estratégia de manutenção cuidadosamente preparada evita o uso incorreto de resultados de mineração de dados (SHEARER, 2000).

2.3 Ferramenta Flowup

Azure Devops, Jira Software, Trello, Github, Gitlab, Flowup são conhecidas como ferramentas para gerir o ciclo de vida de um projeto. Essas ferramentas [...] *são importantes para organização do projeto, armazenar dados. Para analisar o custo do software, temos que reunir métricas do projeto atual e do projeto anterior e traçar semelhanças e analogias para podermos fazer previsões sobre o projeto atual* (NAJADAT; ALSMADI; SHBOUL, 2012). *Flowup* é uma ferramenta que possibilita criar e acompanhar o andamento das atividades, alocação dos funcionários, descrição e diferentes status como, por exemplo: *novo, em andamento, impedido, reaberto, em aprovação do cliente, e concluído* assim possibilitando a otimização da eficiência e, principalmente, do controle dos esforços e documentação das atividades (FLOWUP, 2021).

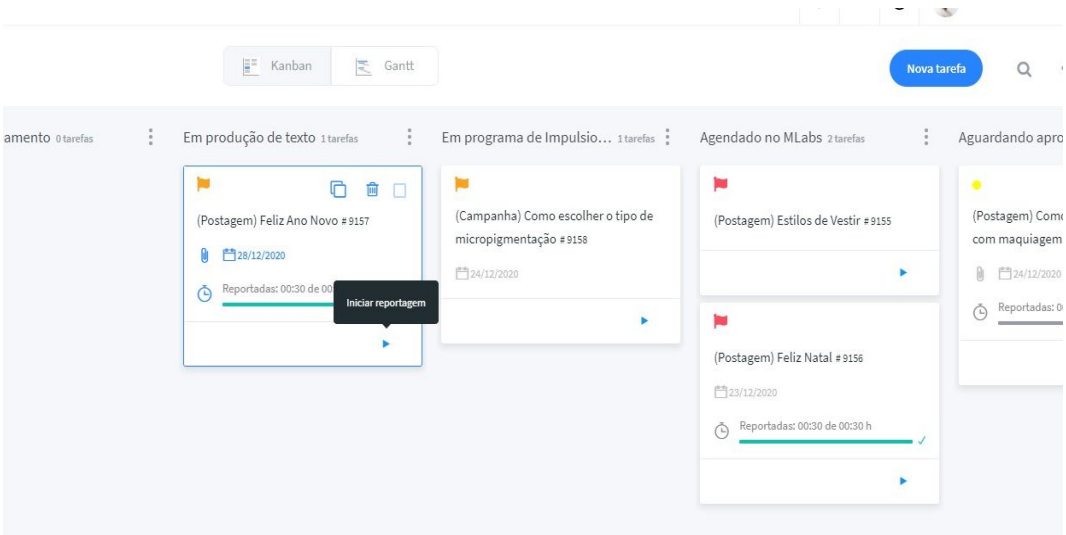

Figura 2 – Fonte: (FLOWUP, 2021)

A Figura 2 indica a tela de organização das atividades. Cada quadrado representa uma atividade e o lugar onde está disposto. O local onde o quadrado se enconta indica a etapa de desenvolvimento, podendo estar no *backlog* (lista de atividades a

serem realizadas), atividades em andamento, concluídas, revisadas ou novas etapas podem ser criadas de acordo com o usuário. É possível observar que na tela da semana, a ferramenta *Flowup* permite que o usuário inicie a contagem do tempo no início da realização da tarefa e parar quando for concluída. A ferramenta suporta a gestão de tarefas, registrando as estimativas, em tempo real de desenvolvimento e controle do status das atividades.

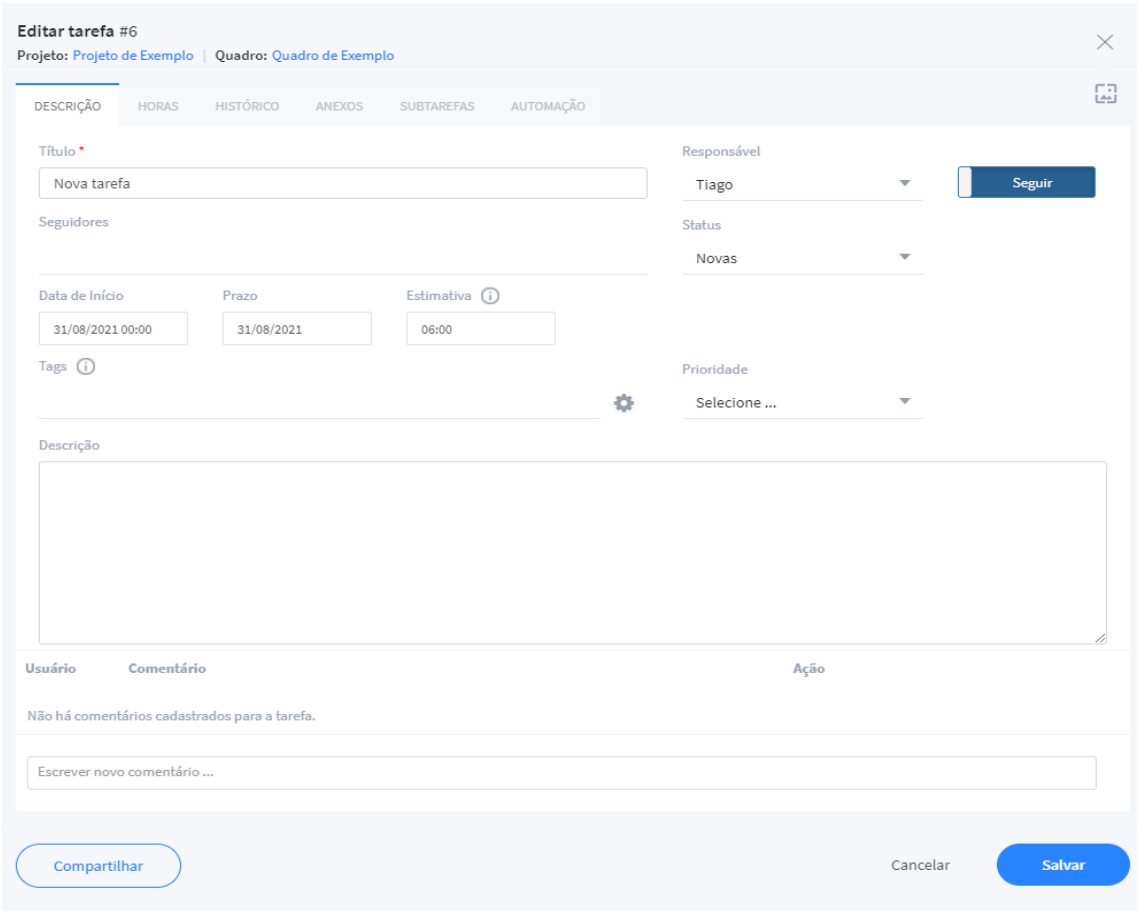

Figura 3 – Tela de editar tarefa - Fonte: (FLOWUP, 2021)

Como pode ser observado na Figura 3, o *Flowup* viabiliza o registro das atividades para organização e acompanhamento do projeto. Nelas são armazedadas informações como título, descrição, nome do projeto, responsável, anexos, quantidade de anexos, subtarefas, total de tempo gasto. Dados que podem ser armazenados e utilizados para modelos de algoritmos de predição.

2.4 *Bag of Words*

Utilizada para criação de *features*, Bag of Words (HARRIS, 1954), é uma ferramenta utilizada para transformar o texto de acordo com a ocorrência das palavras presentes (KOJI; RODRIGUES, 2020). O Processamento de Linguagem Natural (PLN)

(CUMMINGS,) utiliza este método em conjunto com outros para criar bases de dados numéricas a partir de dados categóricos devido a possibilidade de contabilização das palavras. A seguir é descrito o processo de tokenização lexical, empregado para criação de uma *bag of words*, no qual separa as palavras em forma de *tokens*.

```
Texto original: "O caso foi solucionado."

Tokenização Lexical: ["O", "caso", "foi", "solucionado", "."]
```

2.5 Aprendizado de Máquina (AM)

O Aprendizado de Máquina(AM) é uma área em evolução de algoritmos computacionais projetados para emular a inteligência humana aprendendo com o ambiente circundante (NAQA; MURPHY, 2015). A forma com que os dados são apresentados gera uma influência sobre qual modelo de algoritmo de AM deve ser utilizado. Eles são divididos principalmente em quatro categorias: aprendizado supervisionado, aprendizado não supervisionado, aprendizado semissupervisionada e aprendizado por reforço (MOHAMMED; KHAN; BASHIER, 2016). A seguir será apresentado brevemente o que significa cada categoria e quais tipos de dados cada algoritmo utiliza (MOHAMMED; KHAN; BASHIER, 2016).

- Aprendizado Supervisionado

No aprendizado supervisionado o objetivo é inferir uma função ou mapeamento de dados de treinamento rotulados. Os dados de treinamento consistem no vetor de entrada que será representado pela letra X, e no vetor de saída representado pela letra Y, podendo ser chamados de rótulos ou *tags*. Um rótulo ou *tag* do vetor Y é a explicação de seu respectivo exemplo de entrada do vetor de entrada X. Juntos eles formam um exemplo de treinamento. Em outras palavras, os dados de treinamento compreendem exemplos de treinamento. Se a rotulagem não existe para o vetor de entrada X, então X é um dado não rotulado. O vetor de saída Y consiste em rótulos para cada exemplo de treinamento presentes nos dados de treinamento. Esses rótulos para o vetor de saída são fornecidos pelo supervisor. Dois grupos ou categorias de algoritmos estão sob o conjunto de aprendizado supervisionado: classificação e regressão, sendo esta última categoria abordada por este trabalho.

- Aprendizado Não Supervisionado

Ao contrátio do aprendizado supervisionado, faltam supervisores ou dados de treinamento ou seja, os dados não são rotulados. A ideia é encontrar uma estrutura oculta nestes dados. Pode haver uma série de razões para os dados sem rótulo. Pode ser devido à indisponibilidade de fundos para pagar pela rotulagem manual ou a natureza inerente dos próprios dados. Com vários dispositivos de coleta de dados, agora

os dados são coletados em uma taxa sem precedentes. O variedade, velocidade e volume são as dimensões em qual Big Data é visto e julgado. Como descrito por (SARKER, 2021), as tarefas de aprendizagem não supervisionadas mais comuns são agrupamento, estimativa de densidade, aprendizagem de recursos, redução de dimensionalidade, localização de regras de associação, detecção de anomalias, etc.

- Aprendizado Semisupervisionado

Neste tipo de aprendizagem, os dados fornecidos são uma mistura de dados classificados e não classificados. Esta combinação de dados rotulados e não rotulados são usados para gerar um modelo apropriado para a classificação de dados. O alvo da classificação semissupervisionada é aprender um modelo que irá prever classes de dados de teste futuros melhores do que a partir do modelo gerado usando apenas os dados rotulados. Algumas áreas de aplicação onde a aprendizagem semissupervisionada é usada incluem tradução automática, detecção de fraude, dados de rotulagem e classificação de texto (SARKER, 2021).

- Aprendizado Por Reforço

O método de aprendizagem por reforço visa o uso de observações recolhidos a partir da interação com o meio ambiente para realizar ações que maximizem a recompensa ou minimizem o risco. (SARKER, 2021) complementa afirmando sua importância para treinar modelos de IA que podem ajudar a aumentar a automação ou otimizar a eficiência operacional de sistemas sofisticados, como robótica, tarefas de direção autônoma, manufatura e logística da cadeia de suprimentos, no entanto, não é preferível usá-las para resolver problemas básicos ou simples.

2.5.1 Métodos Regressivos

Assim como foi dito anteriormente o aprendizado supervisionado é dividido em dois grupos, classificação e regressão e para melhor entendimento é necessário diferenciá-los segundo suas principais características. A modelagem preditiva de Classificação é a tarefa de aproximar uma função de mapeamento de variáveis de entrada X para variáveis de saída discretas y, como por exemplo rótulos e categorias. O modelo de Regressão também aproxima uma função de mapeamento para variáveis de entrada X, porém em relação a uma variável de saída contínua, isto significa um valor real, um número inteiro ou valor de ponto flutuante (BROWNLEE, 2017). Nesta seção serão apresentados os algoritmos utilizados para o desenvolvimento deste projeto.

2.5.1.1 Linear Support Vector Regression

Segundo (CORTES; VAPNIK, 1995) citado por (FILHO, 2020) - algoritmo Support Vector Machines (SVM) é um classificador utilizado em problemas que buscam

classificar dois grupos. Para atingir esse objetivo, o SVM implementa um modelo que recebe vetores que não são mapeados linearmente e os projeta em uma dimensão mais alta a fim de que possam ser mapeados linearmente, garantindo uma decisão. Para que isso ocorra é necessária uma função chamada de kernel para manipular os dados com um conjunto de funções matemáticas usadas no SVM, a qual geralmente transforma o conjunto de dados de treinamento de forma que uma superfície de decisão não linear seja capaz de se transformar em uma equação linear em um número maior de espaços dimensionais (BARAIK, 2020). A seguir é demonstrado pela Figura 4 a aplicação da função kernel e a projeção de uma dimensão acima para encontrar a superfície de decisão.

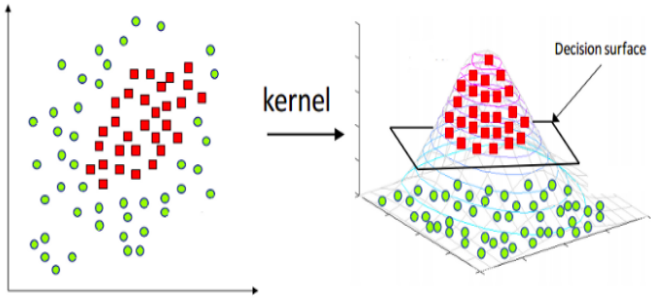

Figura 4 – Aplicação da função kernel. Fonte: (JAIN, 2017)

Às vezes, no entanto, pode não ser impossível separar as duas classes corretamente, ou pode-se ter alguns valores discrepantes, também chamados de *outliers*, que estão dentro da margem. Quaisquer pontos classificados incorretamente ou pontos dentro da margem seriam penalizados. É aqui que entra o valor “folga”, denotado pela letra grega ξ (xi, pronuncia-se “Ksi”). O *Support Vector Regressor* (SVR) se diferencia em alguns aspectos, de uma maneira simplificada é imaginar um tubo com uma função estimada (hiperplano) no meio e limites de cada lado definidos por ε. O objetivo do algoritmo é minimizar o erro identificando uma função que coloque mais dos pontos originais dentro do tubo e, ao mesmo tempo, reduza a ”folga”como pode ser observado na Figura 5 (DOBILAS, 2020).

Segundo descrito pelo Sklearn (PEDREGOSA et al., 2011), o LinearSVR é semelhante ao SVR com parâmetro kernel = ”linear”, mas implementado em termos de liblinear em vez de libsvm. Em razão disso, ele tem mais flexibilidade na escolha de penalidades e funções de perda e deve escalar melhor para um grande número de amostras. Ou seja, o SVR regular com valores padrões usa uma função de base radial como o kernel SVM. Este é basicamente um núcleo gaussiano, também conhecido como curva de sino. Já o SVR linear usa um kernel linear para a função base onde a região entre as classes tem uma forma de V.

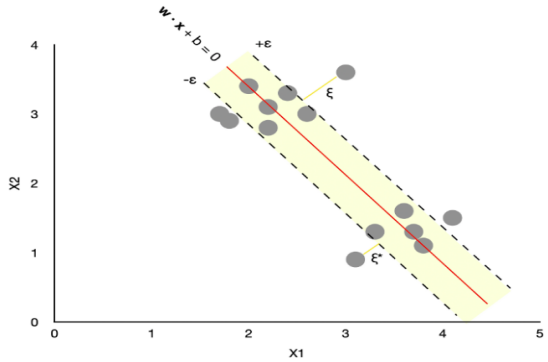

Figura 5 – Support Vector Regression - linha do hiperplano junto com linhas de limite definidas por +- epsilon Fonte: (DOBILAS, 2020)

2.5.1.2 Floresta Aleatória

O Random Forest - Floresta Aleatória é um algoritmo de aprendizagem supervisionada que realiza predições pela combinação de resultados de Árvores de Decisão, podendo ser formado ou por Árvores de Decisão(AD) de classificação ou regressão, ou até mesmo por ambos, dependendo dos dados de entrada.

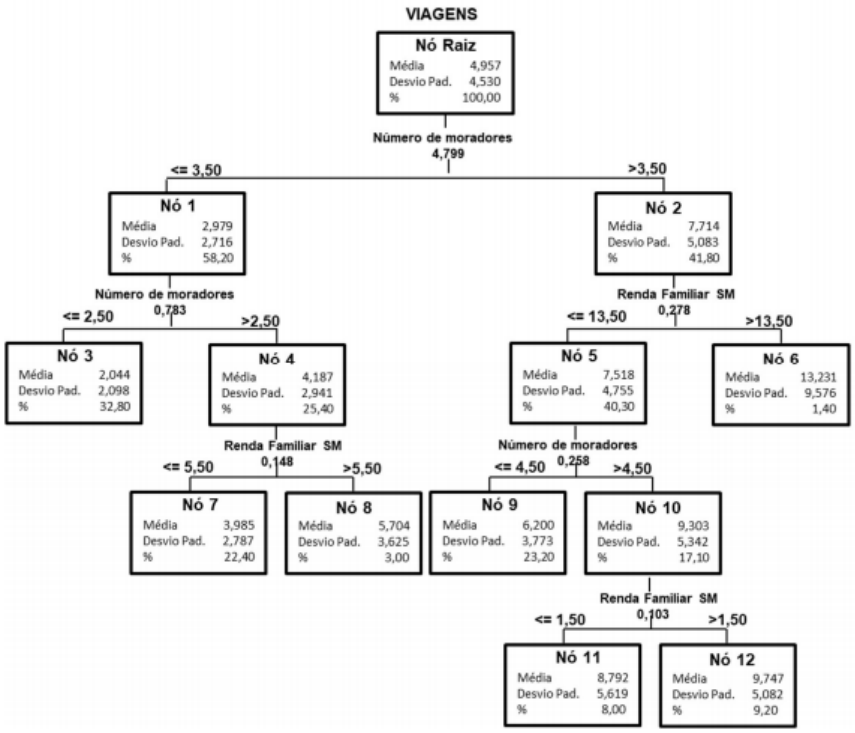

Figura 6 – Exemplo de Árvore de Regressão - Fonte: (PIANUCCI; PITOMBO, 2019)

Na Figura 6 é possível observar a estrutura de uma AD. Ela é Formada inicialmente pelo nó raiz, ponto mais alto da imagem de, maior nível hierárquico e que, a partir dele, ramificam-se os nós filhos. O nó que não possui filhos é conhecido como nó folha ou terminal (LIMA; AMORIM, 2020). Em cada nó, ou quadrado apresentado,

armazena-se uma pergunta. Todos os dados passam pelos nós no sentido da raiz em direção às folhas e, cada folha apresenta uma decisão ou rótulo.

Toda AD presente na FA é uma estrutura de dados gerada por vetores aleatórios e identicamente distribuídos de um mesmo conjunto de dados de entrada do algoritmo. Em seguida um modelo de FA é apresentado Figura 7. É possível também observar como é realizado o *Bootstrap*, embaralhamento dos dados para criar o conjunto de AD. A FA utliliza o clássico método ”a união faz a força”. Os resultados são agregados por meio de votos de modelo ou média em um único modelo de conjunto que acaba superando a saída de qualquer árvore de decisão individual (CHAKURE, 2019). Gerando uma maior diversidade em comparação com a AD, e geralmente leva a criação de modelos com melhores resultados.

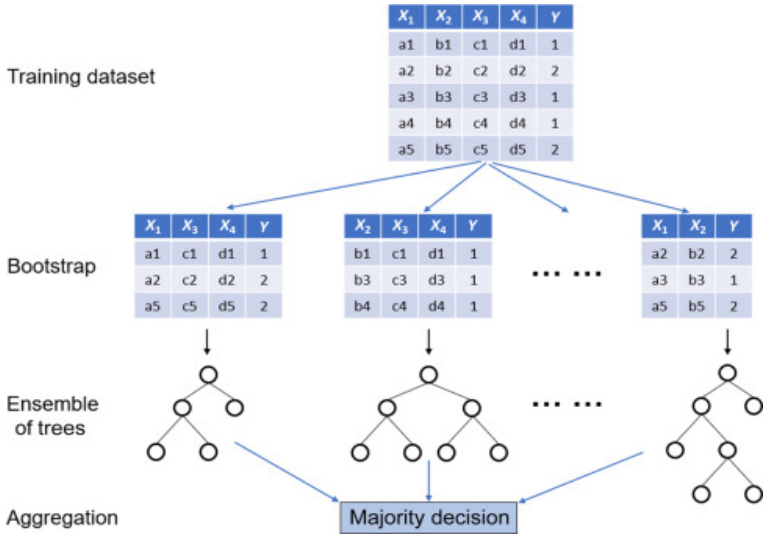

Figura 7 – Modelo de Floresta Aleatória e a formação de suas Arvores de Decisão - Fonte: (MISRA; LI; HE, 2019)

2.5.2 AutoML

Para realização deste trabalho foi utilizado o AutoML(Auto Machine Learning) também conhecido como AM Automático cujo objetivo é tornar o aprendizado de máquina mais acessível, gerando automaticamente um pipeline de análise de dados que pode incluir pré-processamento de dados, seleção de recursos e métodos de engenharia de recursos, juntamente com métodos de aprendizado de máquina e configurações de parâmetros que são otimizados para seus dados (MOORE, 2021).

Com o objetivo de encontrar o melhor modelo de AutoML, foi usado como base o resultado da análise de (BALAJI; ALLEN, 2018), onde foram realizadas análises de comparação das ferramentas TPOT, *Auto-Sklearn* (FEURER et al., 2015), *Auto-ML* (PARRY, 2017) e *H2O's AutoML* (LEDELL; POIRIER, 2020). Balaji e Allen, com o objetivo de apresentar os resultados precisos, reuniram dados de 30 repositórios abertos

de classificação e de regressão, localizados na OpenML (VANSCHOREN et al., 2013), um repositório online de conjuntos de dados de aprendizado de máquina padrão expostos por meio de uma aplicação de disposição de dados. Cada um dos algoritmos de AutoML foram testados com seus parâmetros sugeridos. Além de usar os mesmos dados, o tempo de execução dos AutoML foram os mesmos - 3 horas de busca para cada AutoML - e foi realizada uma abordagem de melhor esforço, para garantir que todos os testes fossem concluídos e que todos os testes tivessem pelo menos 3 chances de sucesso dentro do limite de 3 horas.

Framework	Median F1 score (IQR)	Median Inverted MSE (IQR)
auto-sklearn	**0.753** (0.264)	0.698 (0.224)
auto_ml	0.420 (0.262)	0.825 (0.289)
H2O	0.540 (0.204)	0.835 (0.261)
TPOT	0.679 (0.331)	**0.904** (0.197)

Figura 8 – Resultado da pesquisa realizada por Balaji e Allen para encontrar o melhor algoritmo AutoML para classificação e Regressão - Fonte: (BALAJI; ALLEN, 2018)

A Figura 8 expressa os resultados da pesquisa de Balaji e Allen. Para este trabalho será necessário apenas observar os resultados da *Median Inverted MSE(IQR)* que expressa a Mediana Invertida do Erro Quadrático Médio para avaliar resultados de algoritmos de AM de regressão, ou seja, a Mediana do conjunto de resultados de MSE das predições dos diferentes bancos de dados de regressão. O melhor AutoML observado, portanto, é o TPOT, por apresentar menor distancia de erro (MSE) entre as predições e os valores reais.

2.5.2.1 TPOT

Ferramenta de aprendizado de máquina automatizado em *Python* que otimiza pipelines de aprendizado de máquina usando programação genética (OLSON et al., 2016). TPOT é uma biblioteca de código aberto para a execução de AutoML em *Python*. Ele faz uso da biblioteca de aprendizado de máquina Scikit-Learn (PEDREGOSA et al., 2011) para transformações de dados e algoritmos de aprendizado de máquina e usa um procedimento de pesquisa global estocástico de Programação Genética para descobrir com eficiência um pipeline de modelo de alto desempenho para um determinado conjunto de dados (BROWNLEE, 2020). Como descrito, para execução desta ferramente é necessário desenvolver um script onde são informados a base de dados e os parâmetros da ferramenta TPOT a fim de obter como resultado o melhor *pipeline*

de regressão, ou seja, o melhor algoritmo de regressão com seus parâmetros, para tal conjunto de dados.

2.5.2.2 HyperOpt-Sklearn

O HyperOpt é uma poderosa biblioteca python para otimização de hiperparâmetros desenvolvida por James Bergstra (BERGSTRA; YAMINS; COX, 2013). A ferramenta usa uma forma de otimização Bayesiana para ajuste de parâmetros a fim de obter os melhores para um determinado modelo, além disso, pode otimizar um modelo com centenas de parâmetros em grande escala (DAVID, 2018), este algoritmo pode encontrar hiperparâmetros em menos tempo (iterações) em relação a outros algoritmos de ajuste de hiperparâmetros como o Grid Search e o Random Search (COMPARISON…, 2020). A abordagem de Otimização Bayesiana foca em um modelo de probabilidade P (pontuação | configuração), que é atualizado através de um processo iterativo de consulta a um histórico “H” (pontuação, configuração), cujo objetivo é a maximização da pontuação dada uma configuração “c” (LóPEZ, 2021). Neste projeto o HyperOpt foi utilizado com a extenção HyperOpt-Sklearn, criado com o objetivo de otimizar pipelines de aprendizado de máquina, abordando especificamente as fases de transformação de dados, seleção de modelo e otimização de hiperparâmetros (LóPEZ, 2021). Com isso o usuário deve criar o script informando como parâmetros o *dataset* e os do HyperOpt-Sklearn de acordo com a necessidade do desenvolvedor. No final da execução, o modelo de melhor desempenho é avaliado no conjunto de dados de validação e o pipeline descoberto é informado para o usuário.

2.6 Métodos e Medidas de Avaliação

Para categorizar os resultados é preciso entender como é gerado e qual a função de análise. Algoritmos de *Machine Learning* ou aprendizado de máquina são baseados na matemática e na estatística para que ocorra o reconhecimento de padrões. E surgiram da necessidade de processar e obter informações úteis a partir dos dados (ESCOVEDO, 2020). Para entender como funciona o processo de análise, é necessário entender como funciona a execução dos algoritmos de AM. Ela inicialmente divide os dados em duas partes, o primeiro treinamento e o segundo teste. A primeira parte realiza a criação de um modelo a partir dos dados de treinamento, no qual geralmente é retirado sua maior parte. Após o treinamento o modelo criado a partir dos dados são avaliados a partir dos dados de teste previamente separados, ou seja, o algoritmo não tem conhecimento prévio e não é ajustado especificamente para a nova entrada. O resultado obtido, isto é, a saída da predição a qual, no caso deste trabalho, é o valor em minutos ocorridos em uma tarefa é comparado com o valor real. A partir desta di-

ferença são aplicados os métodos de avaliação. A seguir é demonstrado os métodos utilizados por este projeto.

2.6.1 Raiz do Erro Quadrático Médio

Do termo em inglês *Root Mean Squared Error*(RMSE), Raiz do Erro Quadrático Médio é uma medida de verificação de acurácia e indica a diferença média do valor de predição gerado do modelo em relação ao valor real apresentado no banco de dados. Sua fórmula é apresentada por:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i' - y_i)^2}{n}}.$$

A fórmula acima indica a raiz quadrada do expoente a segunda da média do erro. Ou seja, a diferença de todos os valores preditos com os valores reais. Esta medida apresenta o valor na mesma amplitude da variável analisada, ou seja, se o valor RMSE for 30, significa que há em média 30 minutos de erro em determinado algoritmo analisado. O objetivo com esta medida é aproximar o valor de 0, pois quanto mais próximo de zero menor é a distância do ponto real com o ponto de predição. Esta medida expressa o erro médio do modelo preditivo comparado com os dados reais. Dados que podem ser de treino, teste ou ambos.

2.6.2 Desvio Padrão do Erro

O desvio padrão é a medida que indica o valor da variação em torno da média. Um conjunto pode ter a mesma média, porém, somente com este dado não é possível descrever corretamente o comportamento dos outros valores de um determinado conjunto. O objetivo deste método de avaliação é medir o grau de dispersão em um conjunto. Neste projeto foi utilizado essa medida ao conjunto de erros encontrado pela subtração do valor real em relação ao que foi predito, encontrando assim o valor da variação do erro em relação ao valor do erro médio. Outra característica encontrada é que quano mais próximo de 0 mais homogêneos são os dados. Assim, tem-se:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}.$$

O valor sigma é dado pela raiz do somatório de todos os termos do conjunto em que cada termo (x_i) é subtraído pela média aritmética ($\overline{x}$) dos dados, esta subtração é elevada ao quadrado e dividida pela quantidade de dados (N).

2.6.3 Intervalo de Confiança

O intervalo de confiança tem como objetivo indicar a probabilidade de certeza a partir de um conjunto limitado de dados. Esta probabilidade informa um grau de confiança e compreendendo o desvio padrão é possível determinar o intervalo de confiança. Para avaliação é possível utilizar valor da probabilidade do intervalo de confiança comparado a uma curva gaussiana (ou distribuição normal). A Figura 9 demonstra a relação do desvio padão com relação ao resultado do intervalo de confiança.

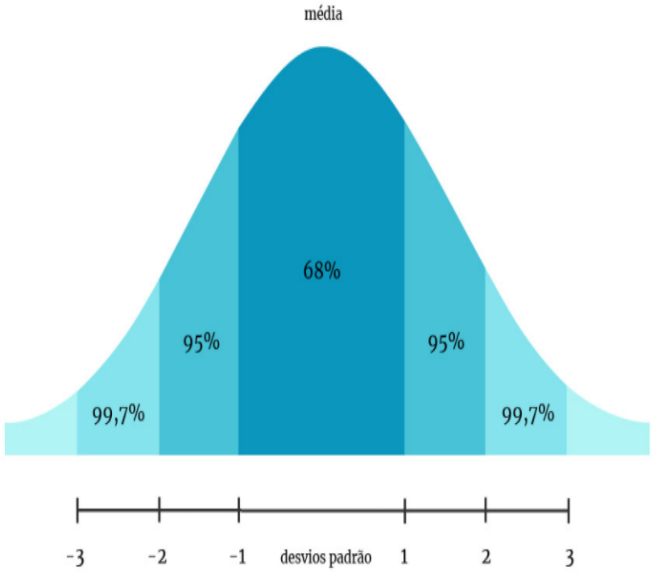

Figura 9 – Curva Gaussiana em relação aos desvios padrão. - Fonte: (SOUZA, 2020)

A partir desta figura é possível observar que 1 desvio padrão (dp) deve corresponder a 68% e, 2 dp a 95%. Ao comparar o valor do intervalo de confiança descoberto e seu desvio padrão encontrado no modelo, é possível observar se os valores estão dentro da margem encontrada na distribuição normal. Se não for o caso e, o valor percentual do intervalo de confiança for menor que 68% em relação a 1 dp o algoritmo pode não apresentar uma grau considerável de confiabilidade.

3 Trabalhos Relacionados

Estimativa de esforço de software é a tarefa de avaliar com precisão a medida de esforço necessária para criar *software* (SHUKLA; KUMAR, 2021). Como dito no capítulo anterior existem diferentes formas de estimativa, e as que se baseiam apenas na experiência humana tendem a ser menos precisas. A realização de predição de esforço por meio de especialistas pode se mostrar uma maneira simples e bem atrativa para as empresas. A seguir serão abordados outros trabalhos relacionados à predição de esforço de software, apresentando seus objetivos, metodologias de pré-processamento de dados e resultados.

Um *outlier* "[...] é uma observação que se desvia tanto de outras observaçõesa ponto de levantar suspeitas de que foi gerada por um mecanismo diferente"(HAWKINS, 1980). A influência do mesmo no *dataset* é investigado por (ONO et al., 2021), cujo objetivo desse estudo é apresentar diretrizes de quanto os outliers influenciam na acurácia da estimação de esforço no desenvolvimento de *software*. Como experimento, foi alterado o valor da variável-alvo, em diferentes conjuntos de dados, assim modificando porcentagem de outliers para efeito de comparação com os resultados da estimativa dos mesmos *datasets* não alterados. Concluiu-se que a influência dos *outliers* no tamanho e esforço do software é insignificante, a menos que a porcentagem ou extensão de outliers seja extremamente grande, por isso a relevância de calcular e retirar os dados que mais se distanciam grandemente da maioria dos outros dados da amostra.

Com objetivo de melhorar o processo de Estimação de Esforço de Software, foi desenvolvido um processo que reúne diferentes modelos de algoritmos de aprendizado de máqina. Este processo foi chamado de Empilhamento de AM (SHUKLA; KUMAR, 2021). O empilhamento de algoritmos é realizado em dois níveis. O primeiro é chamado de metamodelo o qual utiliza o conjunto de dados real; o segundo será usado a partir das previsões dos metamodelos para melhorar a previsão. Além dos métodos de preparação de dados citados acima, o método de *Grid Search* foi empregado para otimizar os parâmetros dos metamodelos. Os algoritmos utilizados foram baseados numa procura literária de acordo com seus resultados, são eles: o *Support Vector Machine* (SVM) (DRUCKER et al., 1997), *Multilayer Perceptron* (MLP) (MURTAGH, 1991) e o *Generalized Linear Model* (GLM) (HARDIN et al., 2007). O chamado *Stacked-SVR*, apresentou melhores resultados para a primeira e terceira parte do conjunto de dados, o *Stacked-GLM* para a segunda, e o *Stacked-MLP* para quarta. O banco de dados utilizado como entrada para o modelo proposto foi o *International Software Benchmarking Standards Group* (ISBSG).[1]. Os dados foram preparados por meio da retirada de

[1] https://www.isbsg.org/ - Acesso em 07 jul 2021

tarefas com campos vazios, onde os mais importantes foram selecionados e divididos em 4 partes diferentes para servir de entrada nos modelos comparativos. Segundo o artigo, a abordagem baseada em conjunto de empilhamento serve para retirar os *outliers* dos dados de entrada. Neste trabalho concluíu-se que diferentes modelos de conjunto de empilhamento de AM performam bem para diferentes datasets pelo fator da adaptabilidade do empilhamento de AM.

Guimarães 2019 apresenta um modelo com o uso de algoritmos de rede neural com o objetivo de encontrar a predição de esforço. Com uma base de dados composta de Relatórios de Problemas, em que é disponível o acompanhamento de todas as falhas e solicitações de melhoria para cada relatório, foram extraídos a Descrição e os Pontos de História para execução do experimento. A Descrição sendo um conjunto de palavras apresentando o problema e os Pontos de História uma sequência de *Fibonacci* utilizada para definir a classe pertencente de cada problema. Para melhorar seus resultados foi realizada uma série de pré-processamentos no banco de dados, que demonstra um importante fator na Classificação Textual. A Rede Neural *Paragraph Vector-Distributed Memory* (PV-DM) apresentou os melhores resultados com 93% *f-measure*, ou seja, a média harmônica entre Precisão (o quão puro é o algoritmo) e Cobertura (quantidade de acertos que o algoritmo conseguiu fazer para cada classe).

O tema estimativa de esforço já foi amplamente explorado por inúmeras publicações, elas possuem o mesmo objetivo de suprir os problemas causados pela estimativa de baixo grau de confiança. Os resultados dos estudos apresentados acima demonstram a importância dos uso do AM para predição de esforço demonstrando a relevância do histórico das atividades bem construído e a possibilidade de desenvolver modelos de diversos tipos para atender a uma grande necessidade do mercado. Mercado este que possui uma grande difereça de características e necessidades. Diferentes técnicas de ML têm diferentes pontos fortes e pontos fracos e, portanto, favorecem diferentes contextos de estimação (WEN et al., 2012). As distintas características do *dataset* geram os diferentes contextos de estimação e cabe ao cientista de dados ter o conhecimento necessário para identificar qual modelo ideal em que se desenvolve o modelo com os resultados mais relevantes.

Após análise dos trabalhos relacionados, evidencia-se o tratamento de dados, aplicação de métodos para alteração da composição dos dados como a retirada de dados ausentes e a selação de *features* como processos importantes para os modelos de regressão. Em conjunto a isso, os algoritmos de regressão se apresentam como ferramentas aptas para resolver problemas com base de dados de diferentes tamanhos e configurações, com ênfase se auxiliado por métodos de pré-processamento de dados.

4 Método Proposto

Neste capítulo serão descritos os métodos para execução deste trabalho. Este projeto abordará as 5 primeiras fases do CRISP-DM. Será apresentado nas próximas seções as etapas correspondentes a cada processo e no capítulo seguinte apenas a fase de Avaliação. A Figura 10 foi desenvolvida para ilustrar as técnicas e métodos usados no projeto. Cada etapa é representada pelos balões colorido e as técnicas usadas em cada etapa é simbolizado pelas figuras dentro dos balões. Observa-se que o ciclo descrito pelo CRISP-DM é representado através do fluxograma: Entendimento do Negócio e Entendimento dos Dados, Preparação dos Dados, Geração do algoritmo de Regressão através do AutoML, Seleção de Features, Avaliação é executado e repetido casos os critérios esperdos forem alcançados, caso contrário um novo ciclo se inicia na tentativa de encontrar resultados satisfatórios.

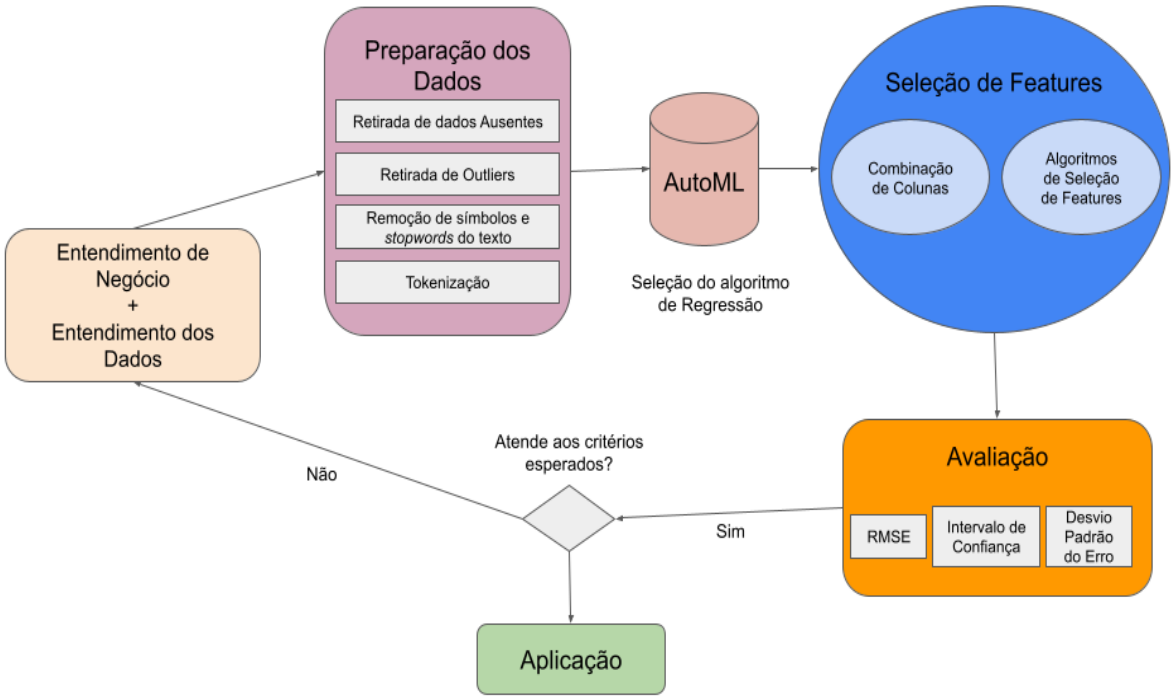

Figura 10 – Fluxograma do projeto. Fonte: Autor

4.1 Entendimento dos Negócios e Entendimento dos Dados

A primeira fase deste trabalho foi constituída pelas duas primeiras fases do CRISP-DM. Inicialmente, para entender o que é a Estimativa de Esforço e como está sendo abordada nos dias atuais, foram feitos estudos de diferentes artigos. Artigos re-

lacionados à predição de estimativa foram pesquisados através do Google Acadêmico, ferramenta de pesquisa que disponibiliza artigos acadêmicos de diferentes publicadoras.

A ferramenta *Flowup* disponibilizou o banco de dados e também as descrições, descrito através da Tabela 1 para auxiliar no entendimento do negócio. Cada coluna é descrita de acordo com sua funcionalidade junto com informações sobre as mesmas.

Coluna	Descrição
empresa	Nome da empresa que possui as tarefas na base de dados.
id	Identificador único da tarefa na base de dados da empresa (podem haver IDs repetidos entre empresas).
title	Título da tarefa, correspondendo a uma breve descrição do que é para fazer na tarefa.
creationdate	Data de criação da tarefa no sistema.
descricao	Campo com uma descrição mais detalhada do que o título da tarefa ou link que o responsável precise para fazer a tarefa.
tags	Tags relacionadas à tarefa. As tags são utilizadas para categorizar tarefas quanto à etapa do processo, tipo de tarefa, cliente, área da empresa, etc.
estimativa	O tempo estimado em horas para conclusão da tarefa.
minutos_trabalhadas	O tempo realmente necessário para realizar a tarefa. Ou seja, o usuário informa quanto tempo ele levou na tarefa.
total_anexos	Quantidade de anexos adicionados à tarefa.
anexos	Nome dos arquivos (anexos) adicionados na tarefa.
projeto	Nome do projeto ao qual essa tarefa foi adicionada (faz parte).
cliente	Nome do cliente para o qual essa tarefa está sendo feita.
tipo_rateio	Divisão proporcional dos custos de acordo com as demandas da execução de cada projeto ou serviço. Todos os seus projetos terão custos diretos e indiretos, e alguns deles irão precisar de mais recursos do que outros.
comentario_obrigatorio	Flag que indica se é obrigatório informar um comentário em cada reportagem de horas incluídas na tarefa. Essa flag é do projeto.
responsavel	O usuário responsável por fazer a tarefa.
automacao	Flag que indica se existe automação na tarefa. Ou seja, se essa tarefa vai gerar outra tarefa igual a ela periodicamente.
total_subtarefas	Quantidade de subtarefas incluídas na tarefa.
subtarefas	Texto das subtarefas incluídas na tarefa. As subtarefas estão separadas por vírgula.

Tabela 1 – Descrição das colunas presentes na base de dados. Fonte - Flowup.

Em seguida foram realizados mais estudos sobre a base de dados. Nesta fase foram realizadas verificações: constituição do banco de dados, verificação de dados ausentes e constituição da variável alvo (*horas trabalhadas*). Verificações, realizadas com auxílio da linguagem de programação *Python* e biblioteca *Pandas* (MCKINNEY, 2010), ferramenta de análise e manipulação de dados. O conjunto de dados inicial apresenta um total de 25012 tarefas, representadas por cada linha e com um total de 29 empresas. Uma amostra do *dataset* é demonstrada através da Figura 11 e da Figura 12,

as figuras representam a forma original do dataset com 19 colunas. Demonstrando a constituição do banco de dados onde cada linha significa uma tarefa e cada coluna uma informação sobre a tarefa. Pode-se observar também pela Figura 1, uma número de dados *NaN* em certas colunas. Este tipo de dado pode afetar na qualidade dos resultados dos modelos de AM, para tratar desta problemática, uma série de métodos de preparação de dados foram aplicados. Para preservar a identidade das empresas algumas informações serão protegidas.

	empresa	id	title	creationdate	descricao	tags	estimativa	horastrabalhadas	totalAnexos
0	[redacted]	54	Dieta, é coisa do passado?	2019-09-17 09:05:35.710	'Vídeo com 45 segundos.	NaN	0	00:53:00	1
1	[redacted]	61	Otimização de Anúncios	2019-09-17 09:46:01.383	NaN	NaN	0	01:30:00	0
2	[redacted]	62	Convidar para Curtir Página	2019-09-17 09:46:01.397	NaN	NaN	0	01:58:00	0
3	[redacted]	64	Verificação Anúncios Google	2019-09-17 09:53:53.473	NaN	NaN	0	00:09:00	0
4	[redacted]	71	Definição de Mote	2019-09-17 10:19:12.810	'ARTES FIXAS: Outdoor 9 x 3 Banner Posto de ...	#campanhaoff	1	02:29:00	0

Figura 11 – Amostra do Dataset Inicial. Parte 1

	anexos	projeto	cliente	TipoRateio	ComentarioObrigatorio	responsável	automção	totalSubtarefas	subtarefas	Unnamed: 18
Dieta coisa do passado..mp4	[redacted]	NaN	Não contribui	0	NaN	N	0	NaN	NaN	
NaN	[redacted]	NaN	Não contribui	0	NaN	N	0	NaN	NaN	
NaN	[redacted]	NaN	Não contribui	0	NaN	N	0	NaN	NaN	
NaN	[redacted]	NaN	Não contribui	0	NaN	N	0	NaN	NaN	
NaN	[redacted]	NaN	Não contribui	0	NaN	N	0	NaN	NaN	

Figura 12 – Amostra do Dataset Inicial. Parte 2

4.2 Preparação dos dados

Nesta seção serão abordados os métodos de preparação do banco de dados e por quais meios foram construídos. A linguagem de programação *Python* em conjunto com o módulo *Pandas* (MCKINNEY et al., 2011) também foram utilizados como ferramentas de desenvolvimento do projeto.

Quando não existem dados limpos, os resultados da mineração de dados estão em questão (SHEARER, 2000). Esta abordagem também se aplica ao AM, pois o déficit de informações de um certo atributo apresenta contribuição menor para o modelo, podendo até atrapalhar no resultado final. Em um modelo de dados pré-processados, ou seja, com zero ruído para cada adicional de 10% de dados de ruído aleatório existe um aumento de 3,2% de aumento de RMSE para modelos de regressão lineares e polinomiais. Basicamente, RMSE é a raiz quadrática média de erro entre valores reais

e predições, isto é, de acordo com os resultados do estudo, os danos em relação à performance dos modelos regressivos se apresentam de forma escalonada (SASEENDRAN et al., 2019). Com isso pode-se obervar o impacto do ruído(*Outliers*) para os algoritmos de AM. *Outliers* são dados que impactam negativamente na qualidade dos resultados de algoritmos de AM.

Nesta fase foi realizada a retirada de dados ausentes, retirada de *outliers*, remoção de símbolos e a tokenização da base de dados. Pode-se observar a seguir na Figura 13 a quantidade de dados ausentes em cada coluna do banco de dados original.

	Colunas	Valores Ausentes	% de Ausentes
0	empresa	0	0.000
1	id	0	0.000
2	title	1	0.004
3	creationdate	0	0.000
4	descricao	12238	48.927
5	tags	16810	67.205
6	estimativa	6	0.024
7	horastrabalhadas	0	0.000
8	totalAnexos	0	0.000
9	anexos	20893	83.529
10	projeto	22	0.088
11	cliente	19822	79.247
12	TipoRateio	21	0.084
13	ComentarioObrigatorio	0	0.000
14	responsável	19583	78.291
15	automção	21	0.084
16	totalSubtarefas	0	0.000
17	subtarefas	24910	99.588

Figura 13 – Quantidade de dados ausentes por cada coluna

As colunas com mais de 75% de dados ausentes foram retiradas do banco de dados, e das colunas restantes foram analisadas com objetivo de identificar a relevância das informações em relação ao modelo a ser criado segundo sua composição e estrutura. A primeira coluna passa por uma transformação em razão de possibilitar um melhor tratamento de retirada das anomalias, a coluna *horas trabalhadas* que, inicialmente, era disposta de diferentes formatos, como *datetime* (HH:mm:ss) e hora por escrito será convertido para minutos sendo nomeado *minutos_trabalhados*.

Para retirada dos *outliers*, em relação à variável alvo *minutos_trabalhados*. Os dados que estiverem no grupo de valor inferior ao percentil de 90 serão mantidos, sendo assim, as tarefas que possuírem um valor maior que 90 percentil serão consideradas *outliers*. Com isso é possível retirar os dados de entrada que se posicionam o

mais longe da curva normal. A Figura 14 demonstra simbolicamente a representação de 90 percentil em uma curva normal.

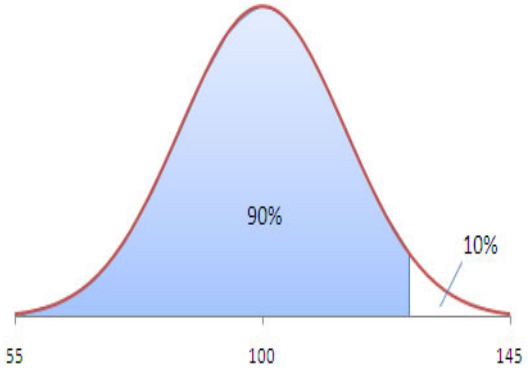

Figura 14 – Representação gráfica do 90 percentil. Fonte: (GUPTA, 2021)

Após a retirada dos *outliers*, as variáveis de texto como título, descrição, tags e projeto passaram pela remoção de símbolos, remoção de letras e números isoladas na frase, acentos das palavras, todas as palavras foram transformadas para minúsculo e a remoção de *stop words* (são palavras que não adicionam informações, ex.: as, e, os, de). A formulação de dados é importante para que os dados possibilitem o melhor resultado de acordo com seu determinado modelo.

A partir do pré-processamento das colunas, limpeza de textos, foi possível criar novas *features* para uso no algoritmo de regressão. Para a criação de novas *features*, utilizou-se a tokenização (WEBSTER; KIT, 1992) consiste em separar textos em palavras, os chamados tokens com a finalidade de extrair unidades mínimas de texto a partir de um texto livre. As frases são separadas em palavras ou por espaço com o objetivo de formar uma lista de *tokens*. Cada *token* é identificado de acordo com a coluna de qual foi originado, sendo assim uma mesma palavra presente em diferentes *features* continuará considerada como uma entrada diferente não provocando a perda de informações no dataset. Para realização do processo de *tokenização*, é criado um dicionário com todas as palavras de cada coluna. As palavras são processadas com a retirada de *stop words*, símbolos, plural para que o menor número de palavras seja introduzido na *tokenização*. Após o processamento, é criado um novo banco de dados com as palavras. Para cada palavra presente, uma nova coluna terá como prefixo a coluna oriunda da palavra: *título* - ”ti_palavra”, *descrição* - ”de_palavra”, *tags* - ”ta_palavra”e *projeto* - ”pr_palavra”, além da coluna alvo *minutos_trabalhados*. Cada empresa possuirá um dataset para realizar a execução dos algoritmos de AM de regressão e análise de resultados. Para cada coluna é observado a marcação ou do número 0 ou 1. O número 0 indica que nesta tarefa não houve a presença deste *token* e o número 1 existe esta palavra correspondente. Sendo reconhecido corretamente em um modelo de regressão.

4.3 Modelagem

Com a base de dados devidamente e estruturado foi possível executar a etapa de Modelagem. A utilização dos algoritmos de AutoML para encontrar os algoritmos de regressão e os hiperparâmetros a serem utilizados para a predição do tempo de esforço. Da mesma forma que a fase de Preparação foi utilizada a linguagem *Python*, junto com a biblioteca Pandas para separação dos dados que serviram de entrada e teste. Como abordado no Capítulo 2 foi utilizado AutoML (THORNTON et al., 2013) por três principais motivos:

- Encontrar os melhores hiperparâmetros de maneira mais acessível, sem a necessidade de conhecer todos os algoritmos de regressão.

- Melhorar a eficiência do sistema de aprendizado de máquina proposto.

- Selecionar o melhor algoritmo de regressão de forma mais rápida que o convencional.

Para esta execução foi realizado dois ciclos de execução dos algoritmos de AutoML, o primeiro chamado de Execução do AutoML global, e o segundo Execução do AutoML por empresa.

4.3.1 Ciclo 1

Para o primeiro ciclo, foi utilizado o conjunto do banco de dados de todas as empresas de forma unificada, a fim de encontrar os melhores resultados de predição de forma geral, evitando o mínimo de diferença de resultados entre si. Os algoritmos de AutoML foram utilizados através das bibliotecas HyperOpt-Sklearn e TPOT, e após a execução foram selecionados, a partir desses algoritmos de AutoML os algoritmos de Regressão e os valores de entrada dos hiperparâmetros propostos pelas ferramentas. Para encontrar o algoritmo de regressão e os valores de entrada dos hiperparâmetros, os algoritmos de AutoML realizam uma função de busca de acordo com cada algoritmo de AutoML. Como descrito no Capítulo 2 o TPOT realiza a busca através de um procedimento de pesquisa global estocástico de Programação Genética e o HyperOpt-Sklearn uma abordagem de otimização Bayesiana, o HyperOpt-Sklearn foi executado com seus parâmetros padrão, sendo que o TPOT com apenas uma alteração do parâmetro de configuração para 'TPOT light'. Os algoritmos encontrados por HyperOpt-Sklearn e o TPOT e seus parâmetros foram testados e comparados, como é possível observar no capítulo seguinte. A execução destes algoritmos foi realizada através da biblioteca *Scikit-learn*, a qual dispõe de um vasto banco de modelos de algoritmos de AM.

4.3.2 Ciclo 2

Com o objetivo de alcaçar melhores resultados e tirar proveito do aprendizado absorvido em todas as etapas do primeiro ciclo. O algoritmo AutoML foi executado novamente utilizando os dados de cada empresa individualmente, de modo que, para cada empresa foi gerado um novo algoritmo de Regressão a partir do AutoML.

Assim foram realizados dois ciclos de execução de AutoML com o segundo ciclo levando em consideração os resultados do primeiro ciclo.

4.3.3 Métodos de Seleção de *Features*

Outros métodos de experimentação foram aplicados nos dados de entrada para o algoritmo encontrado. Os métodos de seleção de *features* são usados com o objetivo de reduzir o tamanho dos dados inseridos como variáveis de entrada, essa redução é realizada de modo em que os dados menos importantes são retirados, através de uma busca realizada por um algoritmo, para execução do algoritmo de AM com os novos dados reduzidos. Com auxílio das bibliotecas Chi Squared, *Mutual Info Regression* e RFE (*Recursive Feature Elimination*) do scikit-learn em que selecionam as n *features*(colunas) que apresentarem os melhores valores dos algoritmos de seleção de *features*. Estes algoritmos foram selecionados pela simplicidade de codificação no projeto e por ser as ferramentas mais recomendadadas através das pesquisas realizadas.

4.4 Avaliação

Os métodos de avaliação dos algoritmos de regressão são métodos objetivos. Como os resultados analisados são dados numéricos, é possível mensurar a qualidade dos modelos gerados por meio de funções matemáticas. No Capítulo 2 foram descritas os indicadores: RMSE, Desvio Padrão do Erro e o Intervalo de Confiança utilizadas na avaliação e comparação dos modelos. O RMSE, por apresentar a diferença entre os valores preditos pelo modelo de AM e os valores reais foi a função mais utilizada para comparação de modelos.

Segundo o CRISP-DM, os modelos construídos na etapa de modelagem devem ser avaliados para assegurar o atendimento dos critérios esperados, em caso do não cumprimento das espectativas o modelo pode ser levado novamente a fase de entendimento e planejamento. Neste trabalho não houve metas ou níveis de acurácia estabelecidos no início do projeto, contudo a média dos valores de RMSE encontrados por todas as empresas foi o método de comparação aplicado. O modelo com a menor média de RMSE de todas as 29 empresas será considerado o modelo com me-

lhor desempenho. Seguindo sempre esta lógica também para o encontro da melhor composição das colunas do banco de dados.

As etapas funcionais de desenvolvimento do projeto e avaliação são descritas na Figura 10. Após a geração do melhor algoritmo de Regressão por meio do AutoML e execução da seleção de *features*. A etapa de seleção de *features* consiste na execução do processo de combinação de colunas e nos algoritmos de seleção de *features* no qual são construídos novos bancos de dados com a retirada de *features*. Após esta etapa é realizada as avaliações com os métodos de RMSE, intervalo de confiança e desvio padrão do erro, os resultados dessas avaliações servem de métrica para conferir se os critérios esperados foram alcançados.

5 Experimentação e Análise dos Resultados

Neste capítulo será descrito os resultados alcançados pelo projeto, os resultados estão divididos entre o ciclo 1 e o ciclo 2.

5.1 Resultados do Entendimento dos Negócios e Entendimento dos Dados

A Figura 15 a seguir, foi construída baseada em um modelo de gráfico a ser construído através da linguagem de código Python junto à biblioteca Seaborn (WASKOM, 2021) disponibilizado por (PIETRO, 2020). Na figura é possível observar que cada espaço vertical representa uma coluna e cada coluna é preenchida por diversas tarefas, tarefas que estão representadas pelas linhas horizontais coloridas. A cor avermelhada apresenta dados Numéricos, a escura dados Categóricos, e a branca Valores Ausentes. É possível observar que uma coluna preenchida pela cor característica do seu tipo de dado significa que todos, ou a grande maioria das tarefas nesta coluna apresentam dados categóricos. Porém se uma coluna apresenta várias barras brancas, significa que existe um valor considerável de dados ausentes.

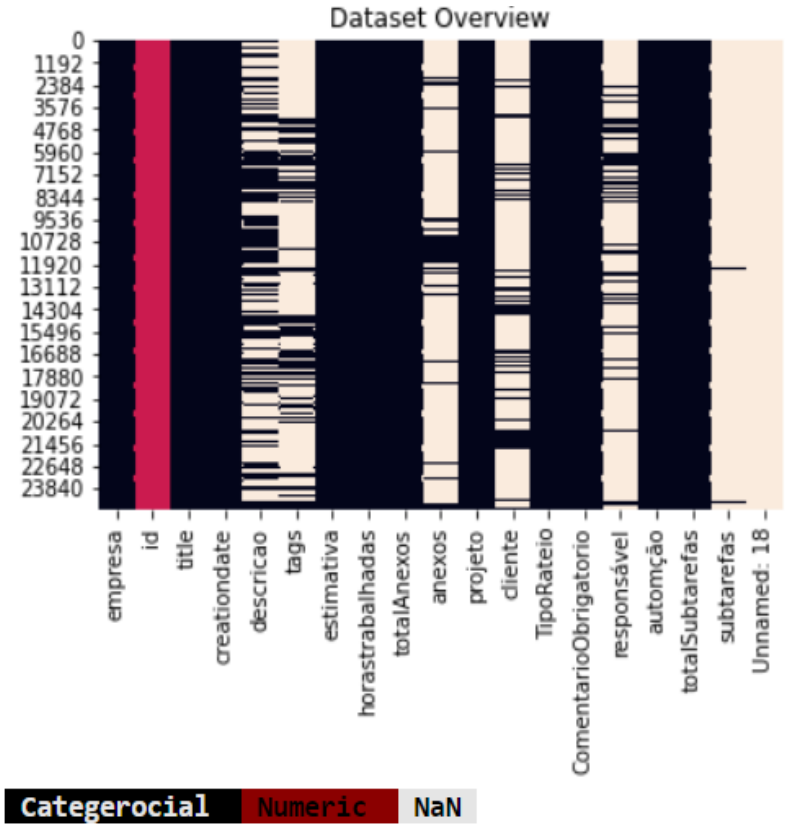

Figura 15 – Representação gráfica dos tipos de dados em relação as tarefas

Este trabalho foi realizado a partir de uma base de dados de empresas reais. E por este motivo esses tipos de dados podem conter muitas falhas como, por exemplo, dados ausentes e dados incorretos. Para auxiliar na tomada de decisão de refinamento e melhoramento dos dados foram criados dois gráficos, que são demonstrados na Figura 15 e na Figura 16, a fim de melhor entender sua constituição.

A Figura 16 demonstra a concentração maior das tarefas em relação à variável alvo. O histograma informa que existe uma maior concentração de tarefas com 0 a 80 minutos trabalhados que vai decaindo até uma concentração muito baixa a partir das tarefas dos 800 minutos trabalhados para mais.

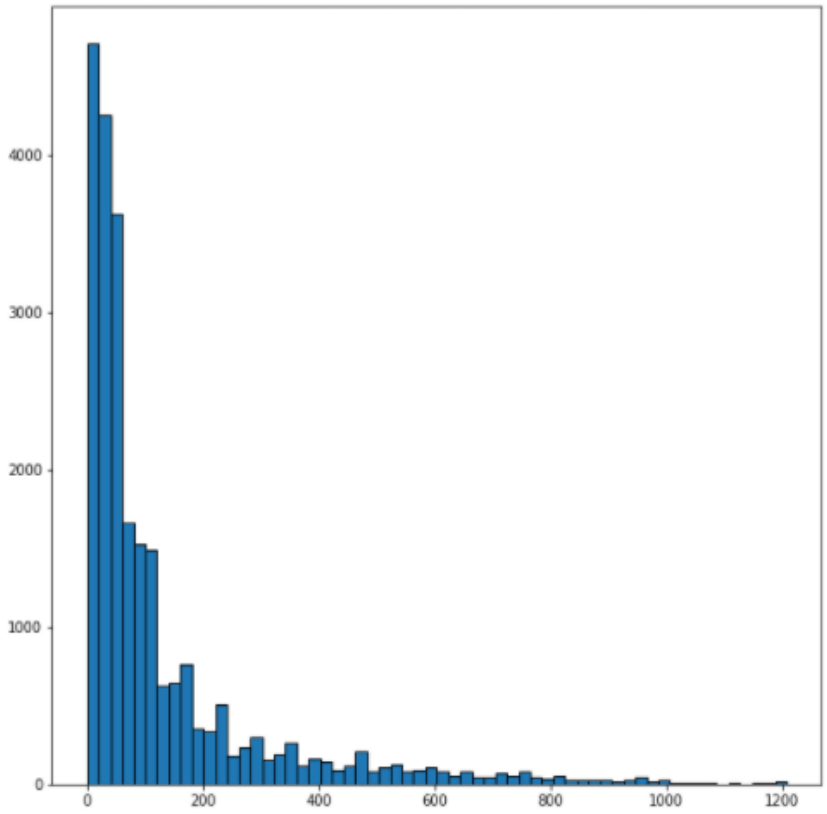

Figura 16 – Histograma que indica a quantidade de tarefas ordenadas pelo valor dos minutos trabalhados

5.2 Resultados da Preparação dos Dados

De acordo com os critérios descritos no capítulo anterior, após análise de todo o banco de dados foram selecionadas para compor um novo conjunto de dados, a fim de serem usadas neste projeto, as seguintes colunas: *empresa, título, descrição, tags,* horastrabalhadas *e projeto*. Seguindo da retirada das colunas com dados ausentes foi realizada a remoção de *outliers*. Na Figura 17, é demonstrado um exemplo do efeito da remoção dos dados (do lado esquerdo é visualisado *boxplot* da Empresa 5 com vários *outliers* e do lado direito o conjunto de dados sem a presença desses pontos). Desta determinada empresa, com um total de 1891 tarefas inicialmente, foram subtraídas 56 *outliers*.

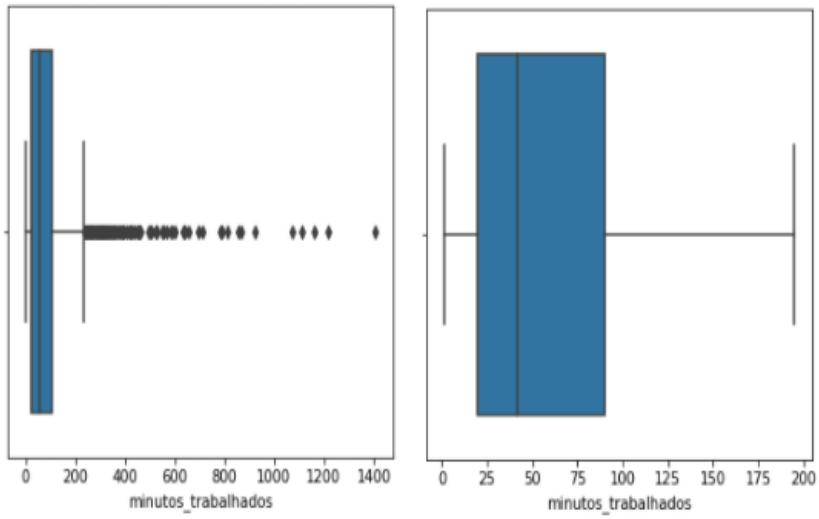

Figura 17 – Comparação do conjunto de dados da Empres antes e depois de retirar os *outliers*. Fonte: Autor

Acima é descrito o comportamento da retirada de *outliers* que ocorreu na Empresa 5 porém, nem todas as empresas tiveram o mesmo resultado, a maioria delas houve uma redução do número de outliers, mas com a permanência de algum número de *outliers*.

Seguinte à realização da retirada de *outliers* e todo o processo de preparação inicial dos dados, como último passo de pré-processamento, foi executada a *Tokenização*(WEBSTER; KIT, 1992), que consiste em transformar dados de linguagem natural para dados numéricos com as identificações das palavras feitas através dos *tokens*, que são as novas colunas. Essas colunas podem ser observadas através da Figura 18, que representa o resultado do processo de tokenização.

	ti_facebook	ti_pessoal	ti_inbox	ti_adsoft	ti_card	...	pr_sport	pr_jl	pr_mkt	pr_protec	minutos_trabalhados
0	0	0	0	0	0	...	0	0	0	0	86.0
1	0	0	0	0	0	...	1	0	0	0	194.0
2	0	0	0	0	0	...	0	0	0	0	159.0
3	0	0	0	0	0	...	0	0	0	0	105.0
4	0	0	0	0	0	...	0	0	0	0	112.0
...	...	...	...	...	...	...	...	...	...	...	...
146	0	0	0	0	0	...	0	0	0	0	67.0
147	0	0	0	0	0	...	0	1	0	0	37.0
148	0	0	0	0	0	...	0	0	0	0	96.0
149	0	0	0	0	0	...	0	0	0	0	129.0
150	0	0	1	0	0	...	0	0	1	0	92.0

Figura 18 – Exemplo do novo formato do dataset de uma empresa - Fonte: Autor

Após o resultado final, foi realizado uma comparação entre a base de dados inicial e a base de dados completamente processada. É possível observar na **??** a

redução de linhas realizada pela remoção de *outliers*, a criação de um número elevado de novas *features*, a retirada de todos os dados ausentes e a alteração do tipo do dataset através da tokenização.

Característica	Dataset Inicial	Dataset Final
Nº de Linhas	27197	24204
Nº de Features	18	17936
Dados Ausentes	107859	0
Tipo do Dataset	Categórico e Numérico	Numérico

Tabela 2 – Comparação entre o Dataset inicial e o Dataset final - Fonte: Autor

5.3 Resultados da Modelagem

Para o projeto, dois ciclos de execução foram performados: Execução do algoritmo AutoML para todas as empresas juntas onde foi gerado um modelo e o segundo ciclo no qual foi executado para cada empresa separadamente a execução do algoritmo de AutoML gerando um modelo específico para cada empresa. Os dois ciclos foram realizados na tentativa de melhoramento dos resultados, pois na execução individual, o algoritmo de AutoML irá gerar um algoritmo de regressão com os melhores resultados apenas para a empresa em questão, diferente da execução global em que é gerado o algoritmo de regressão com os melhores resultados para todas as empresas, assim beneficiando umas mais que as outras.

5.3.1 Ciclo 1: Execução do AutoML global

A seguir, é possível observar os dois algoritmos encontrados pelos dois AutoML e seus respectivos parâmetros. Na Tabela 3, é apresentado os melhores parâmetros encontrados pela ferramenta HyperOpt-Sklearn.

O Random Forest Regressor(RFE) foi o resultado gerado pelo algoritmo AutoML HyperOpt-Sklearn em conjunto com seus hiperparâmetros. A esquerda da Tabela 3, é demonstrado todos os hiperparâmetros e a direita, os melhores valores de entrada encontrados pelo HyperOpt-Sklearn por meio de seu algoritmo de busca.

Na Tabela 4 é possível observar os valores de entrada dos hiperparâmetros gerados pelo TPOT que obtiveram os melhores resultados, bem como o algoritmo de Regressão LinearSVR, o melhor algoritmo encontrado.

É possível observar que foram encontrados dois algoritmos de Regressão, o primeiro Random Forest Regressor através da Tabela 3, e o segundo o LinearSVR por meio da Tabela 4, com cada Regressor tem os próprios valores de entrada dos hiperparâmetros e, embora ambos tenham utilizado o mesmo banco de dados, cada algoritmo

Hiperparâmetros do RFE	Valores gerados pelo HyperOpt-Sklearn
n_estimators	11
criterion	mse
max_depth	None
min_samples_split	2
min_samples_leaf	22
min_weigh_fraction_leaf	0.0
max_features	0.4131
max_leaf_nodes	None
min_impurity_decrease	0.0
min_impurity_split	None
bootstrap	False
oob_score	False
n_jobs	1
random_state	2
verbose	False
warm_start	False
ccp_alpha	0.0
max_samples	None

Tabela 3 – Algoritmo e parâmetros encontrados pelo HyperOpt-Sklearn. Fonte - Autor

Hiperparâmetros do LinearSVR	Valores gerados pelo TPOT
epsilon	0.001
tol	0.01
C	0.01
loss	squared_epsilon_insensitive
fit_intercept	True
intercept_scaling	1
dual	True
verbose	0
random_state	None
max_iter	1000

Tabela 4 – Algoritmo e parâmetros encontrados pelo TPOT. Fonte - Autor.

de AutoML geraram como seu resultado da busca de algoritmos diferentes pois cada AutoML realiza heurísticas de busca diversos para encontrar o melhor algoritmo de regressão.

Após encontrar os algoritmos de regressão gerados pelos algoritmos de AutoML, foi realizada a comparação do TPOT com o HyperOpt-Sklearn. Os dois algoritmos foram testados com o mesmo banco de dados e mesmo ambiente de execuçao. Os resultados da comparação entre os dois AutoMLs são apresentados através da Tabela 5. A coluna Diferença informa a comparação do RMSE do TPOT com o RMSE do Hyperopt-Sklearn, no caso negativo o TPOT foi o melhor em relação ao Hyperopt-Sklearn.

Levando em consideração as empresas, 14 apresentam o TPOT como o melhor algoritmo, e 15 o HyperOpt-Sklearn, resultado bem equilibrado com uma pequena vantagem para o HyperOpt-Sklearn. A diferença negativa representa que o TPOT alcançou

um melhor resultado em comparação ao HyperOpt-Sklearn, e no caso positivo, significa que o HyperOpt-Sklearn obteve um melhor resultado. Para algumas empresas, como a 28, o TPOT apresentou resultado bem relevante, com RMSE -28,00 (-40,07%) e, em outras, como a Empresa 16 com 47,20 (28,67%) em que o HyperOpt-Sklearn obteve o resultado de RMSE bem melhor em relação ao TPOT. Porém o resultado da diferença média de todas as empresas, aponta uma pequena vantagem para o TPOT, com uma diferença de 2,16 RMSE. Dito isso, o modelo do TPOT será considerado o modelo inicial, onde a partir dele serão feitas as análises de seleção de *features*.

Empresa	RMSE TPOT	RMSE Hyperopt	Diferença	% Diferença
0	166,78	173,80	-7,03	-4,04%
1	60,18	54,85	5,34	9,73%
2	52,13	51,46	0,67	1,30%
3	19,36	19,12	0,25	1,28%
4	178,80	187,02	-8,22	-4,40%
5	44,17	46,44	-2,27	-4,88%
6	45,91	46,73	-0,82	-1,75%
7	71,36	62,58	8,78	14,03%
8	29,76	28,28	1,48	5,24%
9	262,75	253,12	9,63	3,81%
10	249,76	250,30	-0,54	-0,22%
11	99,69	92,39	7,30	7,90%
12	172,84	163,85	8,98	5,48%
13	365,14	373,87	-8,73	-2,34%
14	29,89	24,29	5,61	23,09%
15	55,05	54,05	1,01	1,86%
16	211,81	164,62	47,20	28,67%
17	293,13	299,85	-6,72	-2,24%
18	256,09	275,15	-19,06	-6,93%
19	28,67	27,44	1,23	4,48%
20	53,84	67,25	-13,41	-19,94%
21	125,58	142,43	-16,85	-11,83%
22	253,51	248,82	4,69	1,89%
23	101,96	129,97	-28,02	-21,56%
24	273,77	283,77	-10,01	-3,53%
25	49,35	42,80	6,55	15,31%
26	220,26	244,61	-24,35	-9,95%
27	40,26	37,55	2,71	7,21%
28	41,87	69,87	-28,00	-40,07%

Tabela 5 – Resultados dos modelos encontrados pelo TPOT e Hyperopt e a diferença entre eles. Fonte - Autor.

A Tabela 6 reproduz os resultados obtidos pela análise do Intervalo de Confiança, Desvio Padrão do Erro e a Média de tempo das tarefas. Nos resultados do intervalo de confiança, empregado para realizar uma verificação se os dados apresentados representam ou não para 1 desvio padrão a aleatoriedade confiável para os resultados, apenas as empresas 13 e 17 apresentaram um resultado abaixo do ideal (68%). Porém, desvio padrão do erro demonstrou que algumas empresas com grande média de tempo das atividades apresentaram um valor considerável do desvio padrão do erro

em relação a média de tempo. Por exemplo, a Empresa 0 apresenta uma média de tempo das atividades de 194 minutos e o desvio padrão do erro é de 156 minutos de diferença em relação a média em minutos das tarefas de 194 minutos, ou seja, para uma empresa que apresenta em média 3 horas e 10 minutos o algoritmo apresenta uma margem de erro de 2 horas e 36 minutos para mais ou para menos. Até esta etapa, leva-se a conclusão parcial de que os resultados de quase todas as empresas estão dentro da variabilidade aceitável por meio da análise do intervalo de confiança, mas em contrapartida o desvio padrão do erro está perto da média de tempo encontrada o que sifnifica que o erro em minutos está pouco abaixo da média de tempo em ninutos. Como pode-se observar na Empresa) em que com Média de Tempo de 194, 89 minutos o erro pode variar para mais ou para menos aproximadamente 156,10 minutos.

Empresa	Intervalo de Confiança	Desvio Padrão	Media Tempo
0	75,99%	156,10	194,89
1	81,83%	58,02	80,91
2	81,84%	50,43	53,23
3	70,83%	17,10	32,12
4	80,62%	178,88	190,37
5	73,90%	43,58	59,28
6	76,71%	42,09	57,08
7	81,26%	65,17	64,30
8	73,65%	29,03	37,44
9	79,92%	234,65	255,18
10	70,15%	248,28	308,37
11	80,62%	95,15	96,82
12	76,32%	154,89	195,96
13	60,54%	338,78	601,75
14	83,70%	25,53	24,97
15	79,34%	48,78	57,49
16	80,13%	194,21	214,15
17	67,55%	294,26	419,45
18	77,86%	266,41	309,32
19	75,09%	26,16	58,67
20	76,26%	58,12	95,32
21	80,56%	141,07	162,23
22	70,09%	256,94	364,18
23	78,68%	116,33	141,18
24	77,59%	263,04	306,36
25	74,83%	44,90	80,62
26	71,53%	31,79	56,33
27	76,39%	232,08	302,64
28	73,42%	51,57	70,18

Tabela 6 – Intervalo de confiança, Desvio Padrão do Erro e Média de Tempo das Tarefas. Fonte - Autor;

5.3.1.1 Seleção de *Features*

Na tentativa de encontrar um melhor resultado e para averiguar a importância e necessidade de retirar uma ou mais *features* com informações menos relevantes para

o banco de dados. Com os resultados das combinações é possível analisar se é viável ou não retirar duas *features*, como por exemplo: *título* e *descrição*, do banco de dados para realizar a predição em algumas empresas, as que apresentam valor negativo na coluna '% Diferença' podem ser considerados como nova versão simplificada do banco de dados. Conduziu-se uma experimentação da combinação entre todas as colunas de 2 a 2 e combinação de 3 a 3. De um total de 4 colunas(título, descrição, tag e projeto), todas as possíveis combinações foram analisadas. A Tabela 7 e a Tabela 8 apresentaram os resultados da combinação 2 a 2.

Empresa	('ti', 'de')	% Diferença	('ti', 'ta')	% Diferença	('ti', 'pr')	% Diferença
0	163,14	-2,18%	167,89	0,66%	171,66	2,93%
1	56,21	-6,60%	70,74	17,54%	68,00	12,98%
2	52,54	0,79%	50,38	-3,36%	51,30	-1,59%
3	18,63	-3,80%	19,83	2,43%	20,26	4,64%
4	186,70	4,42%	187,53	4,89%	181,46	1,49%
5	46,04	4,22%	45,76	3,60%	45,39	2,76%
6	45,68	-0,49%	42,46	-7,50%	44,80	-2,42%
7	66,02	-7,49%	70,22	-1,60%	65,00	-8,92%
8	32,73	9,97%	30,37	2,04%	29,35	-1,37%
9	220,09	-16,24%	247,30	-5,88%	255,84	-2,63%
10	266,57	6,73%	273,81	9,63%	265,54	6,32%
11	97,33	-2,36%	95,43	-4,27%	99,17	-0,52%
12	157,01	-9,16%	165,06	-4,50%	165,15	-4,45%
13	359,51	-1,54%	365,02	-0,03%	355,64	-2,60%
14	27,75	-7,16%	29,40	-1,65%	27,68	-7,41%
15	49,57	-9,97%	49,00	-10,99%	51,32	-6,78%
16	195,36	-7,77%	164,80	-22,19%	199,50	-5,81%
17	355,16	21,16%	294,53	0,48%	329,41	12,38%
18	274,03	7,01%	236,79	-7,54%	272,16	6,28%
19	28,51	-0,54%	29,15	1,66%	32,14	12,12%
20	56,43	4,81%	75,12	39,53%	64,96	20,66%
21	144,50	15,06%	210,02	67,24%	163,06	29,84%
22	291,88	15,14%	240,62	-5,09%	282,01	11,24%
23	124,15	21,76%	126,79	24,35%	121,90	19,56%
24	295,88	8,08%	299,02	9,23%	284,40	3,88%
25	39,90	-19,15%	58,82	19,19%	54,92	11,29%
26	243,54	10,57%	268,50	21,90%	207,64	-5,73%
27	36,80	-8,58%	32,09	-20,30%	37,96	-5,71%
28	85,49	104,18%	65,14	55,58%	63,11	50,73%

Tabela 7 – Combinação de 2 a 2 e sua comparação com o modelo inicial gerado pelo TPOT. Parte 1. Fonte - Autor.

Empresa	('de', 'ta')	% Diferença	('de', 'pr')	% Diferença	('ta', 'pr')	% Diferença
0	171,62	2,90%	154,66	-7,27%	170,65	2,32%
1	63,37	5,29%	65,05	8,09%	66,07	9,78%
2	50,47	-3,18%	51,58	-1,04%	53,23	2,12%
3	19,40	0,20%	19,32	-0,22%	20,93	8,12%
4	195,67	9,43%	180,55	0,98%	190,20	6,38%
5	45,99	4,11%	43,76	-0,94%	46,29	4,78%
6	43,11	-6,09%	46,63	1,58%	46,87	2,09%
7	67,59	-5,28%	69,98	-1,93%	71,05	-0,44%
8	31,39	5,47%	28,90	-2,89%	28,48	-4,31%
9	243,65	-7,27%	260,80	-0,74%	238,71	-9,15%
10	289,16	15,78%	242,75	-2,81%	261,57	4,73%
11	102,54	2,86%	85,71	-14,02%	89,10	-10,62%
12	165,08	-4,49%	172,98	0,09%	163,33	-5,50%
13	365,30	0,04%	362,90	-0,61%	363,93	-0,33%
14	25,94	-13,21%	30,91	3,41%	28,76	-3,80%
15	44,89	-18,46%	54,32	-1,34%	56,28	2,24%
16	202,32	-4,48%	225,93	6,66%	191,94	-9,38%
17	332,62	13,47%	350,22	19,48%	354,97	21,09%
18	310,67	21,31%	278,34	8,69%	269,41	5,20%
19	25,79	-10,05%	28,92	0,87%	33,22	15,86%
20	66,40	23,33%	63,86	18,61%	74,99	39,28%
21	127,42	1,47%	153,24	22,03%	183,24	45,91%
22	257,21	1,46%	300,98	18,72%	297,86	17,49%
23	130,45	27,95%	112,82	10,65%	120,79	18,47%
24	318,71	16,42%	259,10	-5,36%	234,89	-14,20%
25	50,07	1,47%	58,97	19,50%	47,25	-4,25%
26	202,47	-8,08%	331,39	50,45%	191,62	-13,00%
27	45,21	12,30%	33,78	-16,10%	42,51	5,60%
28	28,21	-32,63%	60,99	45,65%	63,47	51,59%

Tabela 8 – Combinação de 2 a 2 e sua comparação com o modelo inicialgerado pelo TPOT. Parte 2. Fonte - Autor.

De acordo com os resultados dispostos acima, a combinação descrição e tags foi a combinação que mais se aproximou da média dos valores de RMSE em relação aos resultados obtidos com uso do banco de dados completo, resultado 0,18 RMSE maior em relação ao executado por todos os dados. Isso indica que estas duas tabelas são mais importantes em relação as demais, mas não são mais relevantes em relação ao conjunto total de *features*. Porém, levando em consideração as médias de todas as empresas, nenhuma combinação de colunas obteve resultados melhores em relação ao modelo utilizando todas as *features*.

As Tabelas 9 e 10 representam os resultados da combinção 3 a 3. Entre todas elas a combinação que obteve a melhor média geral foi a combinação que retirou a coluna título; a combinação descrição, tags e produto dispuseram de média geral de 4 RMSE de valores RMSE maior em relação aos resultados encontrados com a versão completa do banco de dados.

Empresa	('ti', 'de', 'ta')	% Diferença	('ti', 'de', 'pr')	% Diferença
0	163,68	-1,86%	160,57	-3,72%
1	63,29	5,16%	58,75	-2,38%
2	52,66	1,03%	44,41	-14,80%
3	19,47	0,55%	19,98	3,18%
4	174,69	-2,30%	184,42	3,15%
5	43,67	-1,15%	46,70	5,71%
6	47,15	2,70%	48,26	5,11%
7	73,56	3,07%	76,24	6,84%
8	28,78	-3,29%	28,09	-5,60%
9	201,79	-23,20%	251,60	-4,24%
10	281,08	12,54%	248,10	-0,66%
11	89,22	-10,50%	95,08	-4,62%
12	162,48	-5,99%	163,34	-5,49%
13	370,33	1,42%	375,79	2,92%
14	27,51	-7,98%	26,75	-10,50%
15	54,33	-1,30%	51,04	-7,29%
16	210,44	-0,65%	176,17	-16,83%
17	302,04	3,04%	316,07	7,83%
18	301,05	17,56%	280,06	9,36%
19	27,02	-5,76%	26,05	-9,15%
20	78,16	45,17%	66,73	23,96%
21	176,35	40,42%	137,11	9,18%
22	266,16	4,99%	310,10	22,32%
23	120,58	18,26%	142,83	40,09%
24	297,88	8,81%	333,22	21,72%
25	70,07	41,99%	49,90	1,13%
26	279,58	26,93%	289,02	31,22%
27	29,18	-27,51%	43,74	8,64%
28	66,40	58,59%	63,02	50,51%

Tabela 9 – Combinação de 3 a 3 e sua comparação com o modelo inicial gerado pelo TPOT. Parte 1. Fonte - Autor.

Empresa	('ti', 'ta', 'pr')	% Diferença	('de', 'ta', 'pr')	% Diferença
0	163,49	-1,97%	168,10	0,79%
1	64,95	7,93%	62,68	4,16%
2	53,10	1,87%	50,95	-2,26%
3	21,17	9,33%	17,80	-8,08%
4	185,30	3,63%	180,72	1,07%
5	47,09	6,61%	45,26	2,46%
6	42,26	-7,95%	46,02	0,23%
7	72,21	1,18%	74,03	3,74%
8	28,88	-2,96%	29,29	-1,57%
9	242,70	-7,63%	231,46	-11,91%
10	242,68	-2,83%	250,32	0,23%
11	91,95	-7,76%	107,87	8,20%
12	167,34	-3,18%	170,69	-1,24%
13	377,74	3,45%	387,28	6,07%
14	27,40	-8,32%	25,70	-14,03%
15	49,16	-10,70%	45,61	-17,14%
16	172,41	-18,60%	220,78	4,23%
17	300,52	2,52%	291,22	-0,65%
18	228,37	-10,82%	305,58	19,33%
19	30,69	7,04%	30,20	5,32%
20	68,52	27,28%	58,83	9,27%
21	155,42	23,76%	137,16	9,22%
22	273,04	7,70%	285,17	12,49%
23	126,19	23,77%	134,20	31,62%
24	401,28	46,58%	286,63	4,70%
25	49,98	1,28%	52,30	5,97%
26	251,30	14,09%	192,60	-12,56%
27	40,60	0,85%	38,49	-4,40%
28	50,34	20,22%	49,37	17,92%

Tabela 10 – Combinação de 3 a 3 e sua comparação com o modelo inicial gerado pelo TPOT. Parte 2. Fonte - Autor.

Nos dois tipos de combinação, 2 a 2 e 3 a 3, é possível observar diferentes resultados ao comparar diferentes empresas. Algumas empresas apresentaram uma melhora considerável dos resultados, enquanto outras apresentaram piora. Na média dos resultados de todas as empresas, a combinação apresenta um resultado RMSE 0,18 RMSE a 7,76 RMSE de média entre todas as empresas.

Assim como na tentativa de gerar um banco de dados mais eficiente para execução do algoritmo de regressão, também na combinação de colunas houve um percentual de perda dos resultados de RMSE, ou seja, a diferença entre a predição realizada pelo algoritmo de AM inicial e os valores reais ficaram mais próximos, porém na média geral houve aumento do RMSE. A Tabela 11 apresenta os resultados referentes ao processo de utilizar *frameworks*, oferecidos pelo scikit-learn para realizar a seleção de *features*.

Empresa	Chi-Square	% Diferença	MutualInfo	% Diferença	RFE	% Diferença
0	167,05	0,16%	168,10	0,79%	172,05	3,16%
1	61,42	2,06%	62,68	4,16%	61,24	1,75%
2	53,24	2,14%	50,95	-2,26%	54,39	4,33%
3	19,31	-0,25%	17,80	-8,08%	18,86	-2,61%
4	182,16	1,88%	180,72	1,07%	191,96	7,36%
5	46,62	5,55%	45,26	2,46%	47,42	7,35%
6	42,86	-6,64%	46,02	0,23%	43,63	-4,96%
7	77,07	8,00%	74,03	3,74%	64,23	-10,00%
8	34,52	15,98%	29,29	-1,57%	29,71	-0,18%
9	235,07	-10,53%	231,46	-11,91%	244,72	-6,86%
10	269,22	7,79%	250,32	0,23%	272,02	8,92%
11	111,53	11,87%	107,87	8,20%	98,68	-1,01%
12	171,45	-0,80%	170,69	-1,24%	163,88	-5,18%
13	354,72	-2,85%	387,28	6,07%	337,00	-7,71%
14	29,06	-2,78%	25,70	-9,56%	26,72	-10,62%
15	53,70	-2,46%	45,61	-11,82%	51,09	-7,20%
16	188,61	-10,96%	220,78	4,23%	209,51	-1,09%
17	339,78	15,91%	291,22	-0,65%	312,78	6,70%
18	268,70	4,92%	305,58	19,33%	227,89	-11,01%
19	24,69	-13,89%	30,20	5,32%	27,54	-3,96%
20	62,24	15,60%	58,83	9,27%	68,59	27,40%
21	173,26	37,96%	137,16	9,22%	173,23	37,94%
22	322,07	27,04%	285,17	12,49%	293,83	15,90%
23	146,00	43,20%	134,20	31,62%	128,58	26,11%
24	330,59	20,75%	286,63	4,70%	235,35	-14,03%
25	48,49	-1,73%	52,30	5,97%	58,88	19,32%
26	194,97	-11,48%	243,61	10,60%	261,44	18,69%
27	31,17	-22,57%	38,49	-12,56%	32,92	18,69%
28	45,80	9,39%	49,37	17,92%	44,89	7,21%

Tabela 11 – Seleção de *Features* e a comparação com o modelo inicial gerado pelo TPOT. Fonte - Autor.

A seguir é demonstrado através da Tabela 12, o resumo dos resultados do primeiro ciclo para a Empresa 26 e como é possível melhorar os resultados com diferentes métodos. É possível observar o melhor resultado é gerado a partir da combinação de colunas *tags e projeto* e eliminando as colunas *título e descrição* com RMSE de 191,62, redução de 13% em relação ao resultado base (apenas utilizando o TPOT). O pior resultado é obtido também em uma combinação de colunas *descrição e projeto* com RMSE de 331,39, acréscimo de 50,45% em relação ao resultado base.

Método	RMSE
TPOT	220,26
TPOT + Combinação ”ti, de”	243,54
TPOT + Combinação ”ti, ta”	268,50
TPOT + Combinação ”ti, pr”	207,64
TPOT + Combinação ”de, ta”	202,47
TPOT + Combinação ”de, pr”	331,39
TPOT + Combinação ”ta, pr”	191,62
TPOT + Combinação ”ti, de, ta”	279,58
TPOT + Combinação ”ti, de, pr”	289,02
TPOT + Combinação ”ti, ta, pr”	251,30
TPOT + Combinação ”de, ta, pr”	192,60
TPOT + Seleção de Features Chi Squared	194,97
TPOT + Seleção de Features Mutual Info	249,61
TPOT + Seleção de Features RFE	261,44

Tabela 12 – Resultados das avaliações do primeiro ciclo para a Empresa 26. Fonte - Autor.

5.3.2 Ciclo 2: Execução do AutoML por empresa

Diante dos resultados obtidos no Ciclo 1, considerou-se a possibilidade de que a geração de um modelo específico por empresa poderia trazer melhores resultados. Foi realizado mais uma verificação no intuito de alcançar os melhores resultados. O algoritmo TPOT foi executado para cada empresa, onde foi gerado pelo TPOT um algoritmo e hiperparâmetros diferentes para cada. Através da Tabela 13 é possível visualizar que houve uma melhora em 9 empresas, com destaque para as Empresas 13, 16 e 19 com melhoras de resultado RMSE em torno dos 15% em relação aos resultados com algoritmo e hiperparâmetros gerados apenas pelo TPOT. Por outro lado se destacam negativamente as Empresas 23, 25 e 28 com piores dos resultados em torno dos 35%. Levando em perspectiva valores gerais, um aumento médio de RMSE de 6,14 e, das 29 empresas, 9 apresentaram melhora dos resultados enquanto 20 piora nos resultados.

No ciclo 1, a seleção de features e combinação das colunas não obteve melhores resultados considerando a média de todas as empresas. Visto que em que para cada ciclo executado informações são levadas em consideração no próximo ciclo exe-

cução, não foi realizada novamente para o ciclo 2.

Empresa	RMSE	Diferença	% Diferença
0	168,90	2,12	1,27%
1	54,84	-5,34	-8,88%
2	52,68	0,55	1,06%
3	20,01	0,64	3,32%
4	179,95	1,15	0,64%
5	47,93	3,75	8,50%
6	47,71	1,80	3,91%
7	74,37	3,01	4,22%
8	30,52	0,76	2,55%
9	278,05	15,30	5,82%
10	243,70	-6,05	-2,42%
11	117,05	17,36	17,42%
12	169,63	-3,21	-1,86%
13	326,91	-38,23	-10,47%
14	25,31	-4,58	-15,33%
15	55,19	0,14	0,25%
16	183,92	-27,89	-13,17%
17	319,68	26,55	9,06%
18	291,51	35,42	13,83%
19	23,07	-5,60	-19,55%
20	58,92	5,08	9,44%
21	161,04	35,45	28,23%
22	248,60	-4,91	-1,94%
23	142,36	40,40	39,63%
24	282,21	8,44	3,08%
25	64,37	15,02	30,43%
26	268,88	48,62	22,07%
27	36,43	-3,83	-9,50%
28	57,99	16,12	38,49%

Tabela 13 – Resultado da execução individual do TPOT para cada empresa. Fonte - Autor.

O algoritmo TPOT foi executado novamente para a Empresa 26, com o objetivo de realizar um teste, na tentativa de gerar um melhor modelo com a alteração de parâmetros do TPOT. Foram alterados os parâmetros de *population_size* de 20 para 50 e alterando a configuração de 'TPOT light' para configuração padrão, considerando que o TPOT utiliza um algoritmo de programação genética. Este procedimento foi realizado para aumentar o espaço de procura do algoritmo de busca do TPOT e não convergir tão rapidamente em resultados (algoritmos de regressão) não tão efetivos. Com a mudança houve um decaimento de 58,69 RMSE em relação à execução do TPOT anterior citado na Tabela 13 e 10,07 menor em relação à execução inicial demonstrado na Tabela 5.

É possível identificar a variabilidade do resultado ao tratamento individual de uma empresa através da Figura 19, onde os principais resultados de RMSE estão dispostos para cada empresa. Além dos valores RMSE encontrados por empresa, linhas

transversais indicando a linha de tendência dos valores RMSE indicam a variabilidade dos resultados, no qual uma linha visto na Empresa 0 está mais abaixo e ao seguir para a Empresa 28 está mais acima.

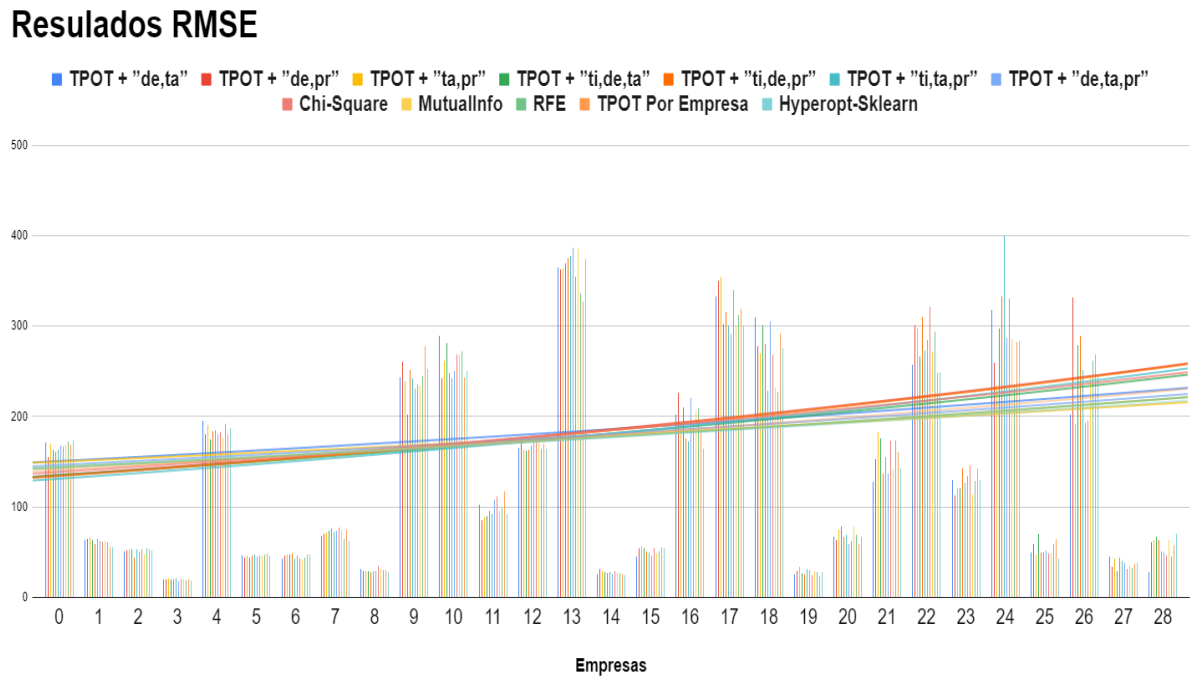

Figura 19 – Comparação de todos os resultados da medida de RMSE. Fonte: Autor

Este trabalho entra em concordância com os resultados do artigo (SHUKLA; KUMAR, 2021) onde é afirmado que os diferentes modelos baseados em conjuntos de empilhamento apresentam bom desempenho para diferentes conjuntos de dados, de modo que, em seu trabalho, para o primeiro e o terceiro banco de dados testados, o SVR se apresentou com o melhor modelo e para o segundo e o quarto banco de dados testados, Stacked-GLM e Stacked-MLP. De mesmo modo, neste trabalho para as diferentes fontes de dados cada uma apresentou resultados diferentes para cada modelo encontrado.

6 Conclusões

Este trabalho teve como objetivo realizar um estudo sobre a realização da predição com o uso do Aprendizado de Máquina Supervisionado baseado em um determinado banco de dados de empresas reais. Para construção do projeto, a metodologia CRISP-DM foi utilizada como base e desta, as 5 primeiras etapas foram aplicadas, como é possível observar através dos Capítulos 4 e 5. Ao todo foram realizados dois ciclos de desenvolvimento: no primeiro, com a execução do AutoML para todas as empresas de uma única vez e, no segundo, houve a execução do algoritmo AutoML para cada empresa. No primeiro ciclo, dois algoritmos de *Auto Machine Learning* para realização da busca pelo melhor algoritmo de regressão e dos hiperparâmetros. Os dois algoritmos apresentaram resultados próximos e o modelo de regressão com o melhor resultado obtido pelo TPOT, o *LinearSVR* foi utilizado. Em seguida, com o objetivo de melhorar os resultados, foi aplicado um processo de otimização. A combinação de colunas e a seleção de *features* foram aplicadas e, segundo análise, a média geral apresentou uma piora nos resultados de RMSE. No segundo ciclo, ao aplicar uma execução individual do algoritmo TPOT para cada empresa, foi percebido nos resultados gerais, uma variação média de RMSE de 6,14 de aumento do RMSE, e das 29 empresas, 9 apresentaram melhora, enquanto 20 piora dos resultados. Isso indica uma queda de qualidade ao executar o TPOT em frações do banco de dados. Tal fator não exclui a importância de realizar uma análise individual para cada empresa. A Empresa 19 apresentou uma queda no RMSE de 19,55% ao realizar a execução individual do algoritmo o que indica que a execução do ciclo 2 apresentou uma melhora nos resultados para algumas empresas. Porém, o exemplo da Empresa 26 demonstra uma piora ao executar o algoritmo TPOT com os mesmos parâmetros, mas ao alterar os parâmetros houve uma melhora de 21,82% em relação à execução individual com os parâmetros não alterados. Estes fatores embasam a importância de um estudo mais detalhado na seleção de um modelo mesmo com o uso de um algoritmo AutoML.

Alguns desafios foram encontrados durante o desenvolvimento do projeto como a qualidade das informações, diferença de resultados apresentados pelas empresas e no tempo e poder computacional. Dados importantes como *descrição*, *tags*, *cliente* e *responsável* apresentaram um percentual de dados ausentes que interferiram no resultado dos algoritmos. Após analisar todo projeto e os resultados obtidos, concluímos que cada empresa apresenta características diferentes. Tais como grau de importância de determinadas informações contidas nas *features*: *título, descrição, tags*, pois ao realizar a combinação de colunas e seleção de *features* como apresentado no capítulo anterior algumas empresas apresentaram bons resultados enquanto em outras

o contrário, o valor de RMSE aumenta com a retirada de algumas colunas através dos métodos de Combinação de Colunas ou Seleção de *Features*. Foi observado, na versão inicial do base de dados, na qual poderíam conter informações relevantes na identificação das tarefas, porém não foram utilizados por haver muitos dados ausentes.

Como limitação do trabalho, o método de avaliação é um fator a ser considerado. O RMSE prevaleceu como medida de avaliação entre algoritmos de AutoML e seleção de *features*. Apenas uma métrica não deve constar todas as verificações necessárias para um modelo. O *Correlation Coefficient Square* (R^2) ou chamado de Coeficiente de Determinação e o *Mean Absolute Error* (MAE) ou Erro Absoluto Médio são exemplos de métricas bastante utilizadas para avaliar modelos de regressão que permitem identificar falhas específicas do modelo. O projeto apresentou necessidade de maior tempo e poder computacional para realização da busca pela melhor seleção de *features* e execução do algoritmo de AutoML deve ser considerado em trabalhos futuros. A experimentação impactará positivamente na busca pelo melhor modelo, visto que, a preparação do banco de dados na realização do pré-processamento, a *tokenização* e a verificação do código de desenvolvimento são atividades importantes e necessitam de diversas análises através de mais ciclos para alcançar melhores resultados. Apesar de o método de seleção de features e combinação de colunas, de modo geral, não apresentar melhora, em certas empresas apresentou bons resultados, como no exemplo da Empresa 26 em que a combinação de colunas *tags e projeto* obteve os melhores resultados no ciclo 1.

Para os trabalhos futuros, também é possível apontar a possibilidade de utilizar outros métodos além de métodos de Regressão como os algoritmos de Rede Neural e Clustering, como por exemplo um novo projeto que teria o objetivo de indicar qual nova tarefa inserir de acordo com o tempo total. Para o ciclo 2 em diante o processo de seleção de features também indicado para ser executado e avaliado em trabalhos posteriores.

Após a apresentação destas conclusões, os resultados demonstram a necessidade de cada empresa buscar o aprimoramento da coleta e organização dos seus dados, bem como prover todas as informações importantes das atividades, desenvolvimento do melhor modelo e avaliação contínua dos resultados. E através desta visualização de um exemplo de como se encontra o estado de uma base de dados real e da aplicação em um modelo de Regressão.

Referências

AGARWAL, R. et al. Estimating software projects. *ACM SIGSOFT Softw. Eng. Notes*, v. 26, n. 4, p. 60–67, 2001. Citado na página 12.

AVILA, A. *Tudo que você precisa saber sobre Estimativa e Métricas de Software*. 2019. Disponível em: <https://www.supera.com.br/tudo-que-voce-precisa-saber-sobre-estimativa-e-metricas-de-software/>. Citado na página 11.

BALAJI, A.; ALLEN, A. *Choosing the best AutoML Framework*: A head to head comparison of four automatic machine learning frameworks on 87 datasets. 2018. Disponível em: <https://medium.com/georgian-impact-blog/choosing-the-best-automl-framework-4f2a90cb1826>. Citado 2 vezes nas páginas 20 e 21.

BARAIK, G. *Major Kernel Functions in Support Vector Machine (SVM)*. 2020. Disponível em: <https://www.geeksforgeeks.org/major-kernel-functions-in-support-vector-machine-svm/>. Citado na página 18.

BASKELES, B.; TURHAN, B.; BENER, A. Software effort estimation using machine learning methods. In: IEEE. *2007 22nd international symposium on computer and information sciences*. [S.l.], 2007. p. 1–6. Citado na página 12.

BERGSTRA, J.; YAMINS, D.; COX, D. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In: DASGUPTA, S.; MCALLESTER, D. (Ed.). *Proceedings of the 30th International Conference on Machine Learning*. Atlanta, Georgia, USA: PMLR, 2013. (Proceedings of Machine Learning Research, 1), p. 115–123. Disponível em: <http://proceedings.mlr.press/v28/bergstra13.html>. Citado na página 22.

BROWNLEE, J. *Difference Between Classification and Regression in Machine Learning*. 2017. Disponível em: <https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/>. Citado na página 17.

BROWNLEE, J. *TPOT for Automated Machine Learning in Python*. 2020. Disponível em: <https://machinelearningmastery.com/tpot-for-automated-machine-learning-in-python/>. Citado na página 21.

CHAKURE, A. *Random Forest Regression*. 2019. Disponível em: <https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f>. Citado na página 20.

Charette, R. N. Why software fails [software failure]. *IEEE Spectrum*, v. 42, n. 9, p. 42–49, 2005. Citado na página 9.

CHARETTE, R. N. Why software fails [software failure]. *IEEE spectrum*, IEEE, v. 42, n. 9, p. 42–49, 2005. Citado na página 11.

COHN, M. *Agile estimating and planning*. [S.l.]: Pearson Education, 2005. Citado na página 11.

COMPARISON of Hyperparameter Tuning algorithms: Grid search, Random search, Bayesian optimization. 2020. <https://medium.com/analytics-vidhya/comparison-of-hyperparameter-tuning-algorithms-grid-search-random-search-bayesian-optimiza Accessed: 2021-12-15. Citado na página 22.

CORTES, C.; VAPNIK, V. Support vector machine. *Machine learning*, v. 20, n. 3, p. 273–297, 1995. Citado na página 17.

CUMMINGS, B. A. J.(1987). natural language understanding. *Journal Computational Linguistics Colume*, v. 14, p. 96–97. Citado na página 16.

DAVID, D. *Alternative Hyperparameter Optimization Technique You need to Know – Hyperopt*. 2018. Disponível em: <https://www.analyticsvidhya.com/blog/2020/09/alternative-hyperparameter-optimization-technique-you-need-to-know-hyperopt/>. Citado na página 22.

DOBILAS, S. *Support Vector Regression (SVR) — One of the Most Flexible Yet Robust Prediction Algorithms*: Aa visual explanation of svr with python implementation examples. 2020. Disponível em: <https://towardsdatascience.com/support-vector-regression-svr-one-of-the-most-flexible-yet-robust-prediction-algorithms-4d25fbd Citado 2 vezes nas páginas 18 e 19.

DRUCKER, H. et al. Support vector regression machines. *Advances in neural information processing systems*, Morgan Kaufmann Publishers, v. 9, p. 155–161, 1997. Citado na página 25.

ESCOVEDO, T. Machine learning: Conceitos e modelos — parte i: Aprendizado supervisionado*. In: . [s.n.], 2020. Disponível em: <https://tatianaesc.medium.com/machine-learning-conceitos-e-modelos-f0373bf4f445>. Citado na página 22.

FEURER, M. et al. Efficient and robust automated machine learning. In: CORTES, C. et al. (Ed.). *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc., 2015. p. 2962–2970. Disponível em: <https://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf>. Citado na página 20.

FILHO, A. J. F. D. *Análise de Métodos de regressão para previsão de demanda de curto prazo*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2020. Citado na página 17.

FLOWUP. *Tenha todas as tarefas do seu time em uma única tela*. 2021. Disponível em: <https://www.flowup.me/empresas-tecnologia>. Citado 2 vezes nas páginas 14 e 15.

GUIMARãES, A. *Técnicas de Comitês para a Estimação de Esforço na Correção de Software*. 2019. Citado na página 26.

GUPTA, S. 90 percentile response time. In: . [s.n.], 2021. Disponível em: <http://www.quotium.com/performance/90-percentile-response-time/>. Citado na página 31.

HARDIN, J. W. et al. *Generalized linear models and extensions*. [S.l.]: Stata press, 2007. Citado na página 25.

HARRIS, Z. S. Distributional structure. *Word*, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954. Citado na página 15.

HAWKINS, D. M. *Identification of outliers*. [S.l.]: Springer, 1980. v. 11. Citado na página 25.

IDRI, A.; HOSNI, M.; ABRAN, A. Systematic literature review of ensemble effort estimation. *Journal of Systems and Software*, Elsevier, v. 118, p. 151–175, 2016. Citado na página 12.

INSTITUTE, P. M. *UM Guia Do Conhecimento Em Gerenciamento de Projetos*. Project Management Institute, 2018. ISBN 9781628251920. Disponível em: <https://books.google.com.br/books?id=jgSMnQAACAAJ>. Citado na página 9.

JAIN, R. *Simple Tutorial on SVM and Parameter Tuning in Python and R*. 2017. Disponível em: <https://www.hackerearth.com/blog/developers/simple-tutorial-svm-parameter-tuning-python-r/>. Citado na página 18.

KOJI, F.; RODRIGUES. *Introdução a Bag of Words e TF-IDF*. 2020. <https://medium.com/turing-talks/introdução-a-bag-of-words-e-tf-idf-43a128151ce9>. Accessed: 2021-12-16. Citado na página 15.

KUMAR, P. S. et al. Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades. In: . [s.n.], 2020. v. 38, p. 100288. ISSN 1574-0137. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1574013720303889>. Citado na página 11.

LEDELL, E.; POIRIER, S. H2O AutoML: Scalable automatic machine learning. *7th ICML Workshop on Automated Machine Learning (AutoML)*, July 2020. Disponível em: <https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf>. Citado na página 20.

LIMA, M.; AMORIM, F. *Random Forest*. 2020. Disponível em: <https://lamfo-unb.github.io/2020/07/08/Random-Forest/>. Citado na página 19.

LóPEZ, F. *HyperOpt: Hyperparameter Tuning based on Bayesian Optimization*: An alternative for the optimization of functions as well as hyperparameters of machine learning pipelines. 2021. Disponível em: <https://towardsdatascience.com/hyperopt-hyperparameter-tuning-based-on-bayesian-optimization-7fa32dffaf29>. Citado na página 22.

MCKINNEY, W. et al. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, Dallas, TX, v. 14, n. 9, p. 1–9, 2011. Citado na página 29.

MCKINNEY Wes. Data Structures for Statistical Computing in Python. In: WALT Stéfan van der; MILLMAN Jarrod (Ed.). *Proceedings of the 9th Python in Science Conference*. [S.l.: s.n.], 2010. p. 56 – 61. Citado na página 28.

MISRA, S.; LI, H.; HE, J. *Machine learning for subsurface characterization*. [S.l.]: Gulf Professional Publishing, 2019. Citado na página 20.

MOHAMMED, M.; KHAN, M. B.; BASHIER, E. B. M. *Machine learning: algorithms and applications*. [S.l.]: Crc Press, 2016. Citado na página 16.

MOORE, J. H. *Information about Automated Machine Learning (AutoML)*. 2021. Disponível em: <http://automl.info/>. Citado na página 20.

MOURA, K. Ciclo de vida dos dados #2. In: . [s.n.], 2019. Disponível em: <https://medium.com/@kvmoura/crisp-dm-79580b0d3ac4>. Citado na página 13.

MURTAGH, F. Multilayer perceptrons for classification and regression. *Neurocomputing*, Elsevier, v. 2, n. 5-6, p. 183–197, 1991. Citado na página 25.

NAJADAT, H.; ALSMADI, I.; SHBOUL, Y. Predicting software projects cost estimation based on mining historical data. *International Scholarly Research Notices*, Hindawi, v. 2012, 2012. Citado 4 vezes nas páginas 9, 11, 12 e 14.

NAQA, I. E.; MURPHY, M. J. What is machine learning? In: *machine learning in radiation oncology*. [S.l.]: Springer, 2015. p. 3–11. Citado na página 16.

OLSON, R. S. et al. Evaluation of a tree-based pipeline optimization tool for automating data science. In: *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. New York, NY, USA: ACM, 2016. (GECCO '16), p. 485–492. ISBN 978-1-4503-4206-3. Disponível em: <http://doi.acm.org/10.1145/2908812.2908918>. Citado na página 21.

ONO, K. et al. Influence of outliers on estimation accuracy of software development effort. *IEICE Transactions on Information and Systems*, The Institute of Electronics, Information and Communication Engineers, v. 104, n. 1, p. 91–105, 2021. Citado na página 25.

PARRY, P. *Automated machine learning for production and analytics*. 2017. Disponível em: <https://github.com/ClimbsRocks/auto_ml>. Citado na página 20.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado 2 vezes nas páginas 18 e 21.

PIANUCCI, M.; PITOMBO, C. Uso de árvore de decisão para previsão de geração de viagens como alternativa ao método de classificação cruzada. *Engenharia Civil*, n. 56, p. 5–13, 2019. Citado na página 19.

PIETRO, M. Machine learning with python: Regression (complete tutorial): Data analysis & visualization, feature engineering & selection, model design & testing, evaluation & explainability. In: . [s.n.], 2020. Disponível em: <https://towardsdatascience.com/machine-learning-with-python-regression-complete-tutorial-47268e546cea>. Citado na página 35.

SALTZ, J.; HOTZ, N. *What is CRISP DM?* 2021. Disponível em: <https://www.datascience-pm.com/crisp-dm-2/>. Citado na página 12.

SARKER, I. H. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, Springer, v. 2, n. 3, p. 1–21, 2021. Citado na página 17.

SASEENDRAN, A. et al. Impact of noise in dataset on machine learning algorithms. 2019. Citado na página 30.

SEHRA, S. K. et al. Research patterns and trends in software effort estimation. *Information and Software Technology*, v. 91, p. 1–21, 2017. ISSN 0950-5849. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0950584917304317>. Citado na página 11.

SHEARER, C. The crisp-dm model: the new blueprint for data mining. *Journal of data warehousing*, THE DATA WAREHOUSE INSTITUTE, v. 5, n. 4, p. 13–22, 2000. Citado 3 vezes nas páginas 12, 14 e 29.

SHUKLA, S.; KUMAR, S. A stacking ensemble-based approach for software effort estimation. In: *ENASE*. [S.l.: s.n.], 2021. p. 205–212. Citado 3 vezes nas páginas 10, 25 e 50.

SOUZA, H. F. Estatística em testes para não matemáticos - p-valor, alfa, beta e tamanho do efeito. In: . [s.n.], 2020. Disponível em: <http://henriquefreitas.com.br/matem%C3%A1tica/2017/11/27/estatistica-em-testes-para-nao-matematicos-parte-5.html>. Citado na página 24.

THORNTON, C. et al. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2013. (KDD '13), p. 847–855. ISBN 9781450321747. Disponível em: <https://doi.org/10.1145/2487575.2487629>. Citado na página 32.

USMAN, M.; MENDES, E.; BÖRSTLER, J. Effort estimation in agile software development: a survey on the state of the practice. In: *Proceedings of the 19th international conference on Evaluation and Assessment in Software Engineering*. [S.l.: s.n.], 2015. p. 1–10. Citado na página 11.

VANSCHOREN, J. et al. Openml: Networked science in machine learning. *SIGKDD Explorations*, ACM, New York, NY, USA, v. 15, n. 2, p. 49–60, 2013. Disponível em: <http://doi.acm.org/10.1145/2641190.2641198>. Citado na página 21.

WASKOM, M. L. seaborn: statistical data visualization. *Journal of Open Source Software*, The Open Journal, v. 6, n. 60, p. 3021, 2021. Disponível em: <https://doi.org/10.21105/joss.03021>. Citado na página 35.

WEBSTER, J. J.; KIT, C. Tokenization as the initial phase in nlp. In: *COLING 1992 Volume 4: The 14th International Conference on Computational Linguistics*. [S.l.: s.n.], 1992. Citado 2 vezes nas páginas 31 e 38.

WEN, J. et al. Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, v. 54, n. 1, p. 41–59, 2012. ISSN 0950-5849. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0950584911001832>. Citado 2 vezes nas páginas 11 e 26.

ZANETTI, D. *A importância do Gerenciamento do Tempo em projetos*. 2017. Disponível em: <https://promovesolucoes.com/

a-importancia-do-gerenciamento-do-tempo-em-projetos>. Citado na página 9.