UNIVERSIDADE FEDERAL DO RECÔNCAVO DA BAHIA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
ENGENHARIA DA COMPUTAÇÃO

LUCAS HENRIQUE COSTA ARAÚJO

APLICAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA A PREDIÇÃO DOS ÍNDICES DE NUTRIENTES NO SOLO

CRUZ DAS ALMAS - BA

2019

LUCAS HENRIQUE COSTA ARAÚJO

Aplicação de técnicas de aprendizado de máquina para a predição dos índices de nutrientes no solo

Trabalho de conclusão de curso apresentado como requisito para a formação no curso de Engenharia da Computação na Universidade Federal do Recôncavo da Bahia, sob orientação do Prof. MSc. Tiago Palma Pagano.

Cruz das Almas - BA
2019

UNIVERSIDADE FEDERAL DO RECÔNCAVO DA BAHIA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS

LUCAS HENRIQUE COSTA ARAÚJO

Esta monografia foi julgada adequada para a obtenção do Grau de Engenheiro da Computação, sendo aprovada pelo Coordenadoria do Centro de Ciências Exatas e Tecnológicas e pela Coordenadoria do curso de Engenharia da Computação do Campus de Cruz das Almas da Universidade Federal do Recôncavo da Bahia e pela banca examinadora:

Orientador: Prof. MSc. Tiago Palma Pagano
UFRB

Prof. DSc. Igor Dantas dos Santos Miranda
UFRB

Prof. DSc. Ramon Pereira Lopes
UFRB

Cruz das Almas, Bahia
06 de dezembro de 2019

Resumo

Este trabalho visa propor um método de recomendação dos nutrientes presentes no solo com base em informações conhecidas sobre a localidade do mesmo, o projeto foi desenvolvido com o intuito de ser integrado ao sistema AdubaTec, desenvolvido pelo autor deste documento e pela equipe de TI da Embrapa-CNPMF, este sistema é responsável por recomendar a adubação e a calagem do solo com base nas informações da cultura e da análise do solo, sendo que este resultado é obtido através de análises químicas de amostras de solo, os modelos de predição que foram obtidos neste trabalho serão úteis na substituição deste processo. Foram realizadas algumas abordagens para a escolha das características que serão utilizadas para realizar as predições de dados, a primeira abordagem consistiu na escolha da latitude e longitude, esta não apresentou bons resultados, a segunda foi adicionar a elevação ao conjunto de dados, também não foi possível apresentar bons resultados e por ultimo na terceira abordagem foi adicionado às entradas o valor do cálcio presente nas amostras, logo após foi possível obter bons resultados de predição em relação aos demais nutrientes, sendo que os algoritmo *Gradient Boosting Regressor* e o *Random Forest Regressor* apresentaram os melhores resultados, possuindo um coeficiente de determinação R^2 maior que 0,7 nos dados de teste para todos os modelos obtidos.

Agradecimentos

A realização deste trabalho e também a conclusão do curso de Engenharia da Computação não seria possível sem a ajuda de diversas pessoas que estiveram presentes em minha vida.

Primeiramente agradeço a Deus por me dar força para continuar, superar meus limites e me ajudar a passar por qualquer barreira que aparecer em minha vida.

Agradeço a meus pais Aderbal Araújo e Tânea Cristina, por me apoiarem em minhas decisões e sempre estarem dispostos a ajudar em qualquer que seja a situação, sendo presentes em todos os momentos de minha vida.

Agradeço a minha esposa Verônica Vieira por estar sempre ao meu lado me ajudando em todos os momentos difíceis e me incentivando a sempre seguir em frente buscando os melhores resultados independente do quão difícil seja.

Agradeço também ao professor e amigo Tiago Palma Pagano, por me orientar em todos esses anos de faculdade, sempre me proporcionando uma evolução tanto profissional quanto pessoal a cada diálogo.

Agradeço ao professor Ramon Lopes, por sempre estar disposto a me auxiliar em qualquer duvida que tive durante o curso e também por me aconselhar diversas vezes na minha tomada de decisão.

Agradeço ao professor Igor Dantas que me auxiliou no desenvolvimento deste trabalho de conclusão de curso.

Agradeço também a Luciano Pontes e Murilo Crespo analistas do NTI da Embrapa-CNPMF, que durante o meu período de estágio me auxiliaram com a minha formação profissional e pessoal, proporcionando para mim um aprendizado imensurável.

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito.
(Bíblia Sagrada, Romanos 12, 2)

Lista de ilustrações

Lista de abreviaturas e siglas

AM	Aprendizado de Máquina
CTC	Capacidade de Troca de Cátions
EMQ	*Erro médio quadrado*
GBR	*Gradient Boosting Regressor*
IA	Inteligência Artificial
K	Potássio
N	Nitrogênio
P	Fósforo
RFR	*Random Forest Regressor*
SB	Saturação das bases
SE	Sistema Especialista
SVM	*Suport Vector Machine*
SVR	*Suport Vector Regressor*
UFRB	Universidade Federal do Recôncavo da Bahia

Sumário

1 Introdução

A área da inteligência artificial (IA) é ampla e pode ser dividida em diversas categorias, cada uma com abordagens especificas para seus domínios de problemas, estas podem ser voltadas para o aprendizado de máquina (*machine learning*), aprendizagem profunda (*deep learning*), para a solução de problemas voltados para visão computacional, o processamento de linguagem natural e outras aplicações (RUSSELL; NORVIG, 2016).

As principais técnicas de IA, tais como o aprendizado de máquina e a aprendizagem profunda, necessitam de um mapeamento e extração de dados relacionados ao domínio que se deseja modelar (HASTIE et al., 2005).

Com essas técnicas e possível criar modelos de predição de dados que são capazes de aprender a características de um conjunto de dados, tanto para classificação quanto para regressão, e ao aprender estas características é possível prever novos dados com base em informações de entradas (HASTIE et al., 2005).

Este trabalho não utilizou técnicas de aprendizado profunda se limitou apenas a algoritmos conhecidos de aprendizado de máquina, isto para criar modelos de predição responsáveis por identificar as características do solo com base em informações de entradas, sendo que foram obtidos três modelos, um para cada nutriente, o magnésio, sódio e potássio.

Dos algoritmos utilizados para a obtenção dos modelos, os que apresentaram melhores resultados foram o *Gradient Booting*, *Random Forest* e o *Suport Vectors Machine*. Os modelos obtidos com estes, apresentaram resultados aceitáveis para as métricas do coeficiente de determinação (R^2) e do erro médio quadrado (EMQ), para o R^2 todos os modelos apresentaram o valor maior que 0,7 nos dados de teste e também um (EMQ) tendendo a 0.

Também, a utilização de técnicas de inteligência artificial se conectam com o desenvolvimento de sistemas especialistas (SE) como sistema AdubaTec, sendo um dos sistemas que se beneficiarão com os modelos de predição apresentados neste trabalho. Os SE podem ser definidos como sistemas que empregam o conhecimento de um especialista ou de uma fonte de conhecimento de determinado problema, proporcionando a tomada de

decisões similares a de seres humanos (HENDERSON, 2009).

Apesar de existir técnicas sofisticadas que consigam interpretar dados e entender os padrões dos mesmos, em diversas áreas isto é feito de forma manual. No setor agrícola existe a necessidade do produtor rural saber quais as quantidades corretas de calcário e adubo para realizar a calagem e adubação de sua plantação, estas informações são obtidas através de pesquisas manuais em livros sobre o assunto.

Para esta problemática especifica é possível modelar um sistema especialista com a finalidade de retornar as informações requisitadas pelo usuário. Para isso é necessário uma entrada de dados, esta consiste nos dados provenientes de um resultado da análise química do solo onde se deseja realizar o plantio de determinada cultura. Estes dados de entrada, quando armazenados podem ser utilizado para identificar as características do solo nas regiões já analisadas. Porém, mesmo com quantidades massivas de dados armazenados, haverá incertezas para locais nos quais não são associados a nenhum dado.

Com isto, este trabalho foi realizado para verificar a viabilidade de utilizar técnicas de IA em uma base de dados que contenha as informações dos nutrientes presentes no solo e outas características, tendo como finalidade prever os índices de nutrientes e eliminar a necessidade da análise do solo em determinadas situações. Estes modelos de predição, poderão ser utilizada em sistemas como o AdubaTec, adicionando um mecanismo que informe os dados de análise do solo com um custo nulo.

1.1 Problemática

Quais estratégias podem ser utilizadas para criar um modelo computacional de predição de dados relacionados às característica dos índices de nutrientes presentes no solo em uma determinada localidade, isso com base em dados de análise do solo preexistentes realizados por laboratórios específicos?

1.2 Justificativa

A necessidade de adubação e calagem do solo para a produção de suprimentos alimentícios agrícolas é universal, sendo que a recomendação do nutriente especifico para a fertilização é dependente das condições do solo em que a cultura será plantada.

O produtor rural necessita saber quais as quantidades corretas de calcário e adubo para realizar a calagem e adubação de sua plantação de determinada cultura, estes dados já existem, porém, estas respostas são obtidas através de pesquisas em livros sobre o assunto. As pesquisas são feitas de forma manual, geralmente por especialistas na área que retornam as recomendações básicas dos macronutrientes N, P e K, estes são os nutrientes mais impactantes na adubação do solo, com base em um resultado de análise do solo. (BORGES, 2009)

A análise do solo, consiste em processos laboratoriais que resultam nas características do solo analisado, este método informa os índices de determinado nutriente no solo, suas propriedades alcalinas entre outros dados. Esta verificação de solo muitas vezes são inacessíveis para a maioria dos produtores rurais, seja pelo seu custo ou pela dificuldade de acesso a laboratórios especializados que realizem tal tarefa.(CLAESSEN, 1997)

Com os resultados laboratoriais em mão, há a necessidade de um especialista da área de agronomia informar quais são as recomendações de nutrientes específicos, o método ideal para a sua aplicação, qual será a necessidade de calagem e como deverá ser realizado este processo, isto para cada cultura que o produtor rural queira plantar. Esta resposta por parte do especialista, geralmente não ocorre de forma imediata e os métodos utilizados para a obtenção dos valores de recomendação são omitidos.

O SE AdubaTec, visa a simplificação deste processo, tendo sua principal função a recomendação da adubação necessária para uma cultura. Este sistema substitui a função

do especialista agrônomo e informa os resultados da recomendação de nutrientes, as suas fontes e a quantidade exata que se deva obter para a compra, tudo de forma automatizada. Este SE utiliza a técnica de encadeamento para frente (GIARRATANO; RILEY, 1998), onde o usuário deverá informar atributos necessários para que o sistema forneça uma adubação precisa, um desses atributos é a análise do solo. Com isso, não há a necessidade de um especialista da área para informar as recomendações dos nutrientes, porém ainda há a necessidade da análise do solo.

Então, percebendo a necessidade de um mecanismo que consiga prever as características do solo, para que não seja sempre necessária a solicitação de análises laboratoriais constantes, o acoplamento de códigos que consiga mapear os dados e criar um modelo predições para identificar as tendências de comportamento do solo em uma região especifica, tornou-se necessário, possibilitando sempre que possível, fornecer recomendações de nutriente em regiões especificas sem a necessidade de uma previa análise do solo. Esse processo terá o intuito de auxiliar o pequeno produtor que não possua a disponibilidade de realizar o processo da análise do solo.

Com a aplicação de técnicas de inteligência artificial é possível reconhecer padrões e identificar tendências de dados (HENDERSON, 2009). Para o problema apresentado podem ser aplicadas técnicas voltadas para o aprendizado de máquina que realizem a regressão de dados, estas serão capazes de criar modelos computacionais responsáveis por predizer as características do solo com base em informações disponíveis sobre o mesmo. Neste trabalho serão abordadas as técnicas, *Random Forrest*, *Gradient Boosting* e *Support Vectors Machine*.

Com isso, um sistema que seja capaz de realizar a predição dos dados informados será útil para diminuir os custos tanto de análises do solo, quanto os de solicitações a especialistas para a interpretação destes resultados. O mesmo retornará os dados de forma imediata e também poderá ser acessível em qualquer momento que o produtor rural necessitar.

Alguns trabalhos correlatos podem ser citados neste documento como: (HELFER et al., 2019) utilizou métodos de regressão linear múltipla para predizer as características de matéria orgânica e argila vinculados aos dados históricos coletados do solo em uma região. Este trabalho apresentou um coeficiente de determinação de 0,9102 para o modelo

de predição da matéria orgânica; (DIAS et al., 2016) utilizou os algoritmos *Random Forest*, *Mult Layer Perceptron* e J48 para classificação do solo a partir de variáveis de relevo, geologia e sensoriamento remoto, sendo que o algoritmo *Random Forest* superou os outros nos resultados, possuindo uma acurácia de 66%; (CAMARGO et al., 2017) utilizou os algoritmos de aprendizado de máquina *Random Forest*, *Regression Trees* para classificar dados do solo para obter o nível produtividade de pastagem do mesmo.

2 Objetivo Geral e Específico

2.1 Objetivo Geral

Criar um sistema que utilize técnicas de aprendizado de máquina para obter modelos preditivos do comportamento dos índices dos nutrientes presentes no solo.

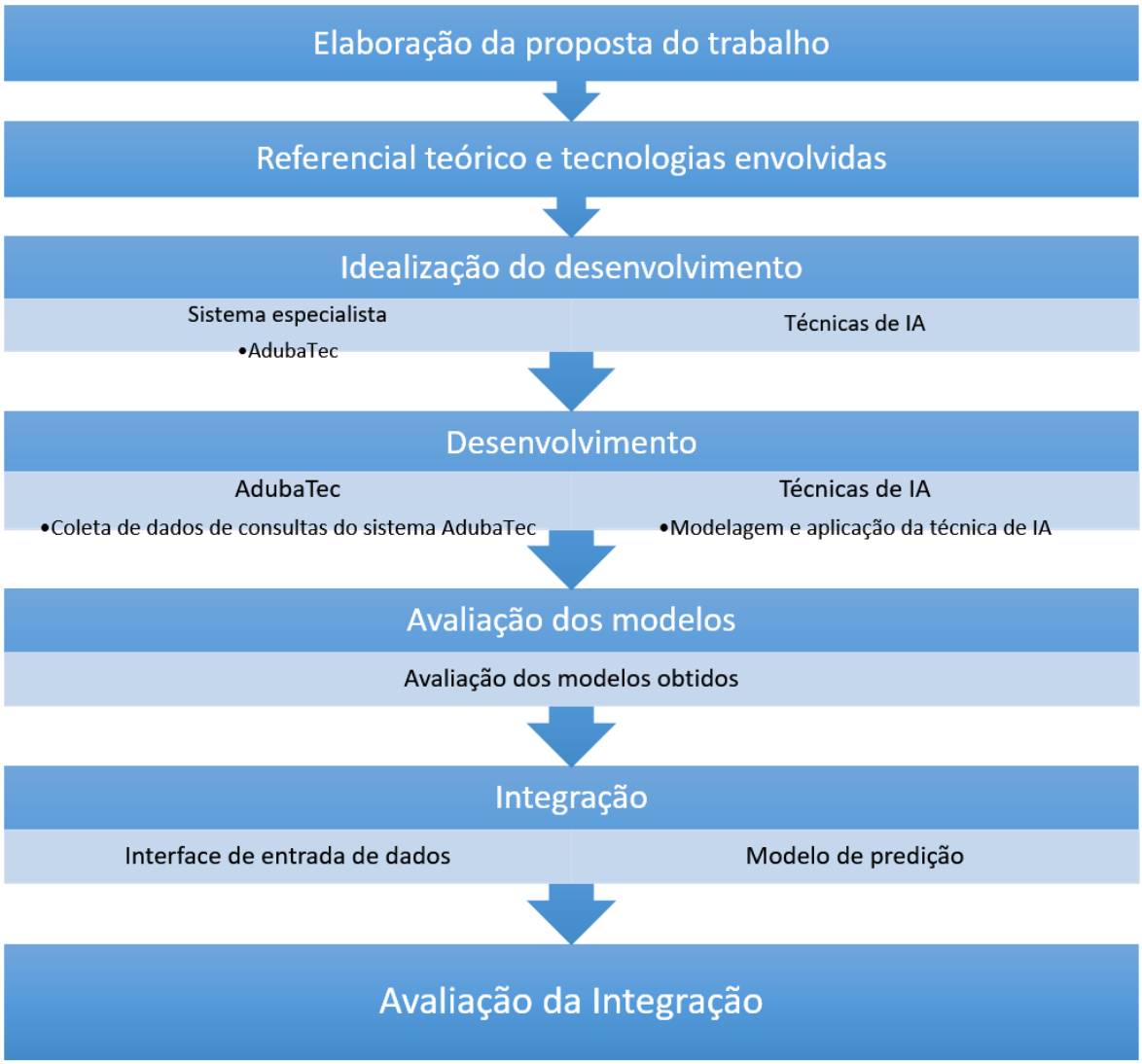
2.2 Objetivo Específico

- Criar uma sistema com os modelos de predição das características do solo, com base em um conjunto de dados existente;
- Criar um mecanismo de integração deste sistema com outros;
- Criar uma interface que seja possível selecionar uma coordenada no mapa e retornar de forma imediata as características desta coordenada;
- Analisar os resultados obtidos e verificar se são aceitáveis.

3 Metodologia

Para atender aos objetivos previstos para este trabalho foi utilizada a seguinte metodologia.

Figura 1 – Diagrama dos passos utilizados na metodologia de desenvolvimento do trabalho



O início do presente trabalho deu-se pela elaboração das propostas contendo seus objetivos e finalidades definindo o seu domínio de conhecimento e os assuntos envolvidos. Após foi realizado uma pesquisa, em bases de conhecimento diversas, para proporcionar o entendimento sobre os assuntos, assim facilitando a construção deste projeto. No decorrer

desta atividade foi definido que este trabalho seria associado ao sistema especialista de recomendação de calagem e adubação o AdubaTec, este sistemas foi desenvolvido pelo autor deste documento durante o período de outubro de 2017 a outubro de 2018 no estágio na Empresa Brasileira de Pesquisa Agropecuária (Embrapa)[1].

Sobre a criação do referencial teórico, durante esta atividade, foram realizadas pesquisas sobre os diversos pontos relacionados ao trabalho, como: Definições de sistemas especialistas e suas principais características e funcionalidades; Técnicas de inteligência artificial e suas aplicações; Arvores de decisão; Técnica *Random Forrest*; Técnica *Gradient Boosting Regressor*; Criação de sistemas Web utilizando a tecnologia Angular®; Conceitos sobre a adubação e calagem do solo para proporcionar o entendimento da necessidade de uma recomendação precisa; Conceito sobre o processo de análise do solo e seus resultados específicos; Entre outros temas abordados que serviram para o entendimento das motivações e necessidades.

O estudo das tecnologias concentrou-se em obter um nível de conhecimento necessário para criação dos softwares e também como aplicar corretamente técnicas de inteligência artificial para a criação de um modelo computacional que retorne as características do solo em função de informações conhecidas sobre o mesmo, isso com base em um conjunto de dados. Os resultados obtidos após a aplicação são apresentados no Capítulo 7.

Este trabalho possuiu a ideia inicial em utilizar os dados de consultas do sistema AdubaTec, este possuem todas as informações de entradas de dados que o sistema precisa para realizar uma recomendação, contendo a análise do solo do local onde a plantação será realizada, porém foi verificado que a quantidade de dados existentes no sistema não é suficiente para criar um modelo de predição. Com isso foi utilizado os dados extraídos do projeto RadamBrasil (Projeto Radar da Amazônia, 1975-1985), este foi uma pesquisa patrocinada pelo ministério de Minas e Energia responsável por analisar características minerais e botânicas presentes nos estados do Brasil. No arquivo Folha SD. 24 Salvador (LIMA et al., 1984), um dos resultado apresentados pelo projeto, conta com diversos dados de análise do solo do estado da Bahia, apresentando as informações da latitude e longitude e as características da amostra, como os índices de nutrientes.

No desenvolvimento foram criados um conjunto de algoritmos responsáveis por

[1] O estágio foi realizado sobre a supervisão do analista de sistema Luciano Vidal Pontes e do chefe do núcleo de tecnologia da informação Murilo Silva Crespo.

buscar os melhores modelos para diversas técnicas de aprendizado de máquina, treinando os mesmos com o conjunto de dados informado anteriormente. Logo após, estes foram avaliados com a finalidade de encontrar o melhor modelo que conseguisse representar a característica geral dos dados, sendo capaz de predizer novos dados com um nível de erro aceitável em relação ao domínio do problema e a sua utilização.

A integração se dará por meio de uma página web e uma *web service* que gerência os melhores modelos obtidos, o mesmo irá receber requisições com as informações de entrada dos modelos, realizar a predição dos dados com base nessas informações e retornar os resultados para a página web. O trabalho tentará identificar quais dados de entradas podem ser utilizadas nos modelos de forma que estes possam ser obtidos facilmente e que consigam apresentar uma relação com os nutrientes a serem preditos.

Neste trabalho foi utilizado bibliotecas com o *sklearn*, *pandas*, *numpy* entre outras, estas auxiliaram na aplicação dos modelos de inteligência artificial, no pre-processamento de dados e também com os mecanismos de avaliação dos dados.

4 Referencial Teórico

Este capítulo visa informar ao leitor as informações teóricas necessárias sobre todos os temas abordados neste projeto. Os temas abordados são:

4.1 Noções de Inteligências Artificial

A área da inteligência artificial visa compreender as entidades inteligentes para construir máquinas que possuam um nível de inteligência igual ou superior a estes seres. (RUSSELL; NORVIG, 2016) no seu livro define a IA como "A IA é estudo de agentes que recebem percepções do ambiente e executam ações". Um agente pode ser considerado como tudo aquilo que pode ser capaz de conhecer seu ambiente, para uma máquina, existe a necessidade de adição de sensores que possam captar as informações e logo após processa-las com a finalidade de obter um resultado especifico.

Diversas técnicas de inteligencia artificial são aplicadas diariamente nas instituições financeiras, bancarias e na indústria, para organizar e gerenciar os dados, também otimizar ações do cotidiano, um exemplo simples seria o corretor ortográfico.

As ações aplicadas a inteligência artificial geralmente envolvem a busca de dados e a análise dos mesmos, algorítimos que utilizam este processo podem ser sem informação ou informada. A busca sem informação consiste solução de um problema informando somente a sua definição, sem haver uma previa indicação de como soluciona-lo, já a busca informada, o domínio do problema já é bem definido e os métodos de como soluciona-los são esclarecidos no algorítimo, os que utilizam segundo método de busca possui uma maior taxa de eficiência na solução de problemas, porém há uma complexidade maior na hora de implementação (RUSSELL; NORVIG, 2016).

As principais atribuição a IA que são utilizados atualmente são: Detecção de fraudes, com técnicas de inteligência artificial é possível identificar clientes fraudulentos ou transações fraudulentas a partir de padrões de comportamentos existentes em um conjunto de dados; Classificação ou regressão de dados, conjuntos de dados podem possuir múltiplos atributos que estão relacionados a um tipo de classificação ou a um valor continuo, é possível utilizar aprendizado de máquina para entender os tipos de características que

estes atributos apresentam e realizar esta classificação ou a regressão. (FACELI et al., 2011).

4.2 Sistemas Especialistas

Um sistema especialista (SE) é um programa que utiliza o conhecimento do domínio de um problema especifico e faz o papel de um ente especialista na área emulando uma metodologia especifica para obter a solução do problema (GIARRATANO; RILEY, 1998). Eles possuem um mapeamento do conteúdo de uma área especifica com a finalidade de conseguir a partir de entradas especificas gerar respostas que sejam capazes de solucionar problemas específicos (TEIXEIRA, 2014).

A primeira introdução bem sucedida do primeiro sistema a utilizar técnicas de inteligência artificial, foi o sistema especialista R1 da empresa Digital Equipment Center que visava prover uma forma de gerenciar os pedidos de novos sistemas (MCDERMOTT, 1982). Este novo tipo de produto computacional se difundiu, fazendo com que quase todas as corporações importantes do Estados Unidos na época possuíssem seu próprio grupo de IA (RUSSELL; NORVIG, 2016).

Umas das definições de sistemas especialistas mais aceitas na comunidade acadêmica é a introduzida por (GIARRATANO; RILEY, 1998) que diz que "Os sistemas especialistas são sistemas computacionais que emulam a capacidade de tomada de decisão de um especialista humano em um domínio restrito".

Todo sistema especialista deve possuir alguns componentes que são essenciais para a sua funcionalidade, estes são: A base de conhecimento compostas pelas regras; O mecanismo de inferência e a interface com o usuário (GIARRATANO; RILEY, 1998).

A base de conhecimento comportas todos os objetos, atributos, regras e cenários possíveis que o sistema apresente e para cada cenário deve haver uma conclusão especifica que o mesmo possa chegar.

Os mecanismos de inferência podem ser encadeados para frente ou encadeamentos para traz. No encadeamento para frente o usuário deve selecionar os atributos necessários, até que seja caracterizado um cenário possível de um problema, desta forma o sistema conseguirá emitir uma resposta para esta definição de atributos. O encadeamento para

traz ocorre de forma contraria do método anterior, é necessário selecionar um cenário especifico com a solução desejada e com isso o sistema informa as possibilidades de atributos específicos que satisfaçam esta solução (GIARRATANO; RILEY, 1998).

Este tipo de sistemas visa fornecer um serviço similar ao de um especialista humano, porém apresenta algumas vantagens como a disponibilidade, o seu custo é reduzido e é permanente, também o SE pode explicar em detalhes a metodologia utilizada para alcançar determinada conclusão oferecendo uma resposta rápida e constante para a mesma problemática (MARCUS, 2013).

As possíveis aplicações de um SE podem variar em diversas situações, mas as principais delas se baseiam em sistemas de diagnostico, planejamento, previsão e controle. Neste documento foi utilizado o sistema especialista de diagnostico o AdubaTec para o auxilio da aplicação de técnicas de inteligência artificial.

4.3 Aprendizado de Máquina

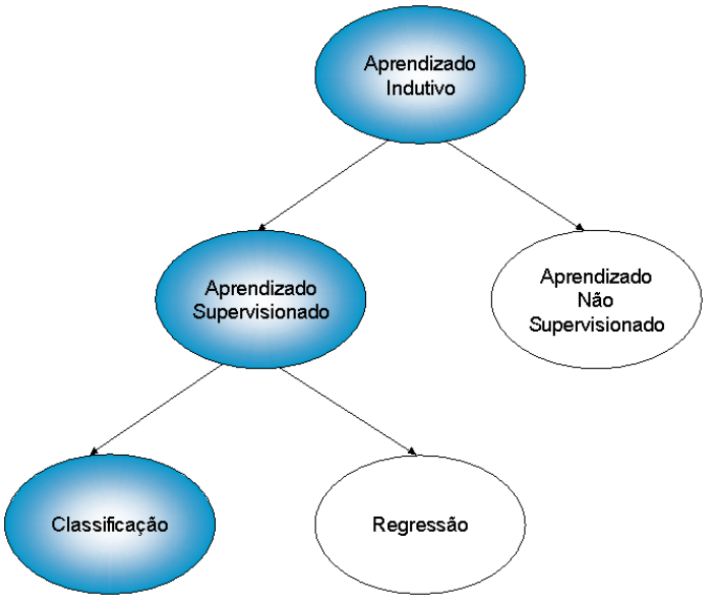
Um dos principais tópicos existentes na área da inteligência artificial é o aprendizado de máquina, este é baseado em princípios estatísticos, porém com algumas variações por possuírem diferentes objetivos como por exemplo as suposições para o aprendizado de máquia possuem o objetivo de proporcionar o entendimento de características existentes a partir de um conjunto de dados, já para a estatística é realizado o estudo da probabilidade de ocorrer ceto tipo de saída (FACELI et al., 2011). Este tópico possui uma característica importante para a área da IA, que é a capacidade de aprender de acordo com as entradas de dados (MACHADO, 2015).

Uma ponto de partida que temos para entender o que é o aprendizado de máquina(AM) seria o processo de indução lógica (MONARD; BARANAUSKAS, 2003). Este propicia entender características particulares a partir de um conjunto de exemplos, podendo ser capaz de identificar uma equação que caracteriza o comportamento de dados e com isso sendo possível identificar características de outros dados que não sejam previamente conhecidos, este método é aplicado a um conjunto de exemplos conhecidos, porém em alguns casos os resultados obtidos podem não representar o comportamento real de um problema.

A aprendizado indutivo pode ser classificado como supervisionado ou não supervisi-

onado, isto também ocorre com o aprendizado de máquina (MONARD; BARANAUSKAS, 2003). A figura apresentada a seguir exibe a hierarquia do aprendizado de máquina.

Figura 2 – Hierarquia do aprendizado de máquia. Fonte:(MONARD; BARANAUSKAS, 2003)



4.3.1 Conjunto de Dados

O conjunto de dados que é utilizado em *machine learnig* conta com exemplos de amostras no qual se deve aplicar as técnicas para extrair uma modelagem computacional. A tabela a seguir exibe um exemplo de como pode ser um conjunto de dados.

Tabela 1 – Exemplo de um conjunto de dados

Índice	Atributo 1	Atributo 2	...	Atributo N	Classe/Valor
1	X_{11}	X_{12}	...	X_{1N}	Y_1
2	X_{21}	X_{22}	...	X_{2N}	Y_2
...	...	...	...	...	...
N	X_{N1}	X_{N2}	...	X_{NN}	Y_N

Neste exemplo é possível verificar que cada dado possui seus atributos e a sua classificação especifica, desta forma é possível aplicar técnicas de aprendizado de máquina para realizar uma modelagem computacional para predição de uma classificação ou regressão em um novos tipo de dados não possua a mesma.

Segundo (MONARD; BARANAUSKAS, 2003), os atributos podem ser nominais, quando não existe uma ordem de valores, podendo ser dados do tipo texto ou podem ser

contínuos quando são valores numéricos pertencente aos domínios dos números reais.

Em alguns casos nem todos os atributos são necessários em um conjunto de dados, isso ocorre quando a classificação não esta diretamente relacionada a esta característica, então para criar uma modelagem eficiente é necessário remover os atributos desnecessários do *dataset*(conjunto de dados)(KAUFMAN; MICHALSKI, 1998).

Uma outra característica que pode existir no conjunto de dados é o que pode ser denominado ruído, isto ocorre quando os dados são imperfeitos ou apresentam alguns valores registrados de forma incorreta devido a um processo de geração, de coleta ou de classificação realizada com erros. Isto pode ocasionar um baixa precisão dos resultados obtidos pelo modelo de aprendizado de máquina.

No aprendizado supervisionado o conjunto inicial, responsável por treinar a IA deverá possuir as informações dos dados classificados para o método de classificação e valorados para o método de regressão. Neste conjunto os dados apresentam seus atributos, estes são responsáveis por descrever alguma característica inerente ao domínio de estudo que o conjunto representa.

A classe presente em cada dado em um *dataset* no aprendizado supervisionado representa o rótulo de interesse para o indutor, toda modelagem deve levar em conta quais características de atributos levam a uma rotulação especifica, ou seja, como os atributos se relacionam para obter uma classe especifica, sendo que estas já são informadas no caso do supervisionado, no não-supervisionado o algoritmo será responsável por agrupar os dados e classificá-los, sendo que logo após esse processo deve ser validado dando sentido a classificação dentro do contexto do problema.

Antes de utilizarmos o conjunto dados nos modelos de aprendizado de máquina é necessário aplicar técnicas para eliminar incertezas em relação aos mesmos, estas serão apresentadas na seção seguinte.

4.3.1.1 Pré-processamento de dados

Para a aplicação das técnicas de *machine learning* como, *Random Forrest*, *Gradient Boosting Regressor* em um conjunto de dados, antes deve ser realizado um pré-processamento de dados com a finalidade de diminuir erros durante a aprendizagem.

O objetivo do pré-processamento de dados é realizar um análise minuciosa no

conjunto de dados com a finalidade de remover informação incoerentes, com por exemplo classificações impossíveis ou valores improváveis, estes podem ocorrer durante a coleta dos dados (GARCÍA; LUENGO; HERRERA, 2015). Para realizar este procedimento é necessário um conhecimento básico sobre o domínio do contexto dos dados .

Nesta são feitas análises para a remoção de valores redundantes, atributos desnecessários para a problemática abordada, dados ausentes ou (*missing values*), ao final também podem ser utiliza uma técnica de normalização de dados quando os mesmos forem numéricos.

4.3.1.1.1 Dados ausentes e redundantes

Em relação aos dados ausentes a técnica de remoção pode ser utilizada, porém deve se levar em conta que existe a possibilidade de se perder características necessárias para a solução do problema, outros tipos de abordagem visam preencher estas lacunas com base nos dados existentes, para conjuntos de dados que contenham uma relação de dados ausentes baixa, possível completar estas lacunas utilizando a média ou a mediana dos dados existentes (PADILHA; CARVALHALHO, 2017).

Os dados redundantes em uma base dados, são aqueles considerados duplicados ou relativamente próximos, quando isto ocorre o modelo irá apresentar dificuldade de convergência, apresentando resultados que podem não ser aceitáveis (PADILHA; CARVALHALHO, 2017), para solucionar isto pode ser adotado a remoção destes dados.

Uma outra técnica que auxilia a criação de modelos computacionais que consigam prover resultados aceitáveis é a normalização dos dados.

4.3.1.1.2 Normalização de dados

Em diversas situações os valores que representam os atributos e as características dos dados são numéricos e contínuos, estes podem possuir escalas e faixas de valores diferentes, isto pode gerar incertezas na hora da criação do modelo de predição. Valores com escalas maiores tendem ser ponderados de forma desigual, isto pode ocasionar um certo nível de incerteza em relação os resultados gerados (FACELI et al., 2011).

A principal técnicas de normalização de dados e a reescala, esta consiste em alterar a escala dos valores do conjunto de dados para uma faixa de valores pré-definida, geralmente

estando dentro do intervalo $[0, 1]$, sendo 1 relacionado como o valor máximo da coluna a ser normalizada e 0 com o valor mínimo.

A equação 4.1 exibe como pode ser obtido os valores normalizados de uma coluna, com o índice i representado a linha do conjunto de dados e o índice j representado a coluna, com min_j e max_j representado os valores minimo e máximo das colunas, respectivamente.

$$x_{ij} = \frac{x_{ij} - min_j}{max_j - min_j} \tag{4.1}$$

A normalização é indicada quando os atributos dos dados são numéricos, também pode ser aplicado para os valores que representam a característica que se deseja predizer quando for utilizado em uma técnica de regressão.

4.3.2 Erro e Precisão

Um modelo obtido com o aprendizado de máquina é construído através de um indutor que utiliza um conjunto de dados similar ao apresentado na tabela 1 para realizar o treino, sendo que este *dataset* possuem seus atributos e suas classes ou valores, no caso da regressão, previamente informados (KAUFMAN; MICHALSKI, 1998).

Com o sistema treinado é possível utilizar outro conjunto de dados para testar o modelo obtido, este conjunto é similar ao utilizado para o treino, com seus atributos e classes ou valores bem definidos, porém nesta etapa o indutor irá predizer a classe ou valor em função dos seus atributos, com isso é possível analisar o erro e a precisão que a modelagem obteve em relação a problemática, comparando os valores conhecidos com os obtidos.

Existem diferentes metodologias para avaliar o erro e a precisão das modelagens de regressão e de classificação a seguir é fornecida uma introdução que como são utilizados estes mecanismo em cada tipo de técnica.

4.3.2.1 Classificação

O erro pode ser obtido através da equação $erro(h)$(4.2), onde h é a representação do classificador obtido, nesta temos o somatório do resultado da equação $g(i)$ (4.3), esta é a representação da verificação da classe retornada pelo algoritmo e da sua classe real presente do *dataset* de teste, retorna 1 quando o valor obtido do classificador for diferente

do valor original da amostra e 0 quando for igual (SAMMUT; WEBB, 2011). Com isso a equação do erro retorna quantas classificações foram feitas incorretas em relação ao total de dados n.

$$erro(h) = \frac{1}{n}\sum_{i=1}^{n} g(i) \tag{4.2}$$

$$g(i) = \begin{cases} 1, & y_i = h(x_i) \\ 0, & y_i \neq h(x_i) \end{cases} \tag{4.3}$$

Para realizar os cálculo da precisão, é importante entender um conceito chamado matriz de confusão (SAMMUT; WEBB, 2011), está uma tabela onde são apresentados algumas informações relacionadas ao modelo.

Suponha que a Tabela 2 representa a classificação real de clientes que podem realizar empréstimo (valor 1) e que não podem (valor 0), realizada com base em informações financeiras extraídas de um conjunto de dados, também apresenta a classificação predita por um modelo de classificação com base nos mesmos dados.

Tabela 2 – Exemplo de um conjunto de dados

Classe real	1	0	1	0	1	1	0	1	0	1	1	0
Classe predita	1	1	0	0	1	1	1	1	1	1	1	1

Através desta tabela podemos modelar a matriz de confusão, esta consiste em uma matriz 2x2 onde são informados os valores:

- **Verdadeiro Positivo(VP)**, este ocorre quando a classe que desejamos analisar foi classificada de forma correta. Um cliente pode realizar empréstimo e o modelo o classificou corretamente.
- **Falso Positivo(FP)**, este ocorre quando a classe que desejamos analisar foi classificada de forma incorreta.Um cliente pode realizar empréstimo e o modelo o classificou que não pode.
- **Verdadeiro Negativo(VN)**, este ocorre quando a classe que não desejamos analisar foi classificada de forma correta. Um cliente não pode realizar emprestimo e o modelo predisse corretamente.

- **Falso Negativo(FN)**, este ocorre quando a classe que não desejamos analisar foi classificada de forma incorreta. Um cliente não pode realizar empréstimo e o modelo predisse que poderia.

A Tabela 3 exibe o formato da matriz de confusão.

Tabela 3 – Modelo da matriz de confusão

		Valores Preditos	
		Empréstimo	Não Empréstimo
Valores Reais	Empréstimo	VP	FP
	Não Empréstimo	FN	VN

Verificando a Tabela 2 podemos completar os valores, a Tabela 4 exibe os resultados.

Tabela 4 – Exemplo de um conjunto de dados

		Valores Preditos	
		Empréstimo	Não Empréstimo
Valores Reais	Empréstimo	6	1
	Não Empréstimo	4	1

Com as informações da tabela, podemos aplicar diversas métricas de avaliação do modelo, a seguir serão exibidas algumas.

Pode ser obtido o cálculo da acurácia do modelo, esta indica uma quantas classificações o modelo acertou em relação ao conjunto total de dados testados. Esta pode ser obtida pela Equação 4.4.

$$Acuracia = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.4}$$

Também podemos verificar a precisão de acerto de uma classe, tomando a classe principal como a positiva, a precisão será dada pela Equação 4.5.

$$Precisao = \frac{VP}{VP + FP} \tag{4.5}$$

O *recall* ou sensibilidade é uma métrica que avalia quantos resultados corretos foram indicados em relação ao total que deveriam ser indicados, esta métrica é importante quando o peso de um falso negativo é crucial para o problema (PEDREGOSA et al., 2011). Para o exemplo apresentado anteriormente sobre empréstimo esta métrica é crucial, pois

quando, for negado uma operação que deveria ser aprovada, pode haver baixas a receita. A Equação 4.6 apresenta como pode ser obtida esta métrica.

$$Recall = \frac{VP}{VP + FN} \tag{4.6}$$

Também existe um outro mecanismo de avaliação quando se trata de um modelo de classificação, este é conhecido como o F1-*Score* esta é uma média harmônica entre os valores de obtidos na Equação 4.6 e na Equação 4.5, é uma maneira de observar uma métrica relacionada a precisão e o *recall* ou sensibilidade do modelo, quando esta métrica esta baixa significa uma das anteriores também esta (SAMMUT; WEBB, 2011).

$$F1 = 2\frac{Precisao * Recall}{Precisao + Recall} \tag{4.7}$$

4.3.2.2 Regressão

As equações referentes ao cálculo do erro apresentadas anteriormente são validas apenas para sistema de classificação, quando o objetivo é a regressão, ou seja, modelar um indutor que seja capaz de predizer um dado com base nos dados de entrada, é necessário realizar o calculo da distância do valor previsto com o valor real. De forma mais comum são utilizadas duas equações para este tipo de medição, o erro médio quadrado (*mean squared error*)(Equação 4.8) ou a distância absoluta média(*mean absolute distace*)(Equação 4.9), ambas as equações podem ser visualizadas a seguir.

$$mse(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h(x_i))^2 \tag{4.8}$$

$$mad(h) = \frac{1}{n}\sum_{i=1}^{n}|y_i - h(x_i)| \tag{4.9}$$

A Equação 4.8 representa o erro médio quadrado e a Equação 4.9 o erro médio absoluto, em ambas n é o tamanho total de dados existentes no conjunto utilizado na medição, y_i representa o valor conhecido para as entradas de dados x_i, h representa o modelo indutor utilizado para a regressão e $h(x_i)$ representa os valores preditos pelo modelo com base em dados de entrada.

Em algumas situações somente a verificação do erro médio quadrado pode ser utilizada como a métrica principal para avaliar se um modelo foi adequado para determinado de problema.

Outra métrica existente para avaliar um modelo é o coeficiente de determinação, também conhecido como R^2 *score*, este coeficiente indica o quanto um modelo consegue extrair ou explicar os comportamento de um conjunto de dados, ou seja, ele informa se o modelo é ideal ou não para o *dataset* escolhido (TJUR, 2009).

Esta é uma definição estatística utilizada como métrica principal em diversos problemas de *machine learnig*, segundo (NAGELKERKE et al., 1991) e importante para avaliar o quanto as variáveis dependentes (dado a ser predito), conseguem ser explicas pelas variáveis independentes (características de entrada do *dataset*). Valores maiores que 0.5 são consideráveis aceitáveis para o coeficiente de determinação (STARNES; YATES; MOORE, 2010).

Este resultado vária entre 0 e 1, sendo que quanto maior este valor melhor é a avaliação do modelo, um R2 de 0.67 indica que as variáveis de entrada conseguem explicar 67% das variações que ocorrem nos resultados que se desejam prever.

As equação a seguir descrevem o processo para o calculo do coeficiente de determinação de um modelo de regressão.

A equação 4.10 exibe como é feito o processo para encontrar a média dos dados observados, ou seja, aqueles que já estão presentes no conjunto de dados, a variável y_i indica isto, $i = 1..n$ com n sendo o tamanho do *dataset*.

$$\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{4.10}$$

A Equação 4.11 exibe como é feito o processo para obter a variança dos dados conhecidos, este processo é feito através o somatório das diferença dos quadrados do valor conhecido pela sua média Equação 4.10.

$$SQ_{tot} = \sum_{i=1}^{n}(y_i - \overline{y})^2 \tag{4.11}$$

A Equação 4.12 exibe como é feito o processo para obter a diferença quadrática entre o valor conhecido y_i e o valor predito $\hat{y}^2$, este processo também é conhecido como a

somatória dos erros residuais.

$$SQ_{res} = \sum_{i=1}^{n}(y_i - \hat{y})^2 \tag{4.12}$$

Com as equação apresentadas é possível obter o coeficiente de determinação R^2 *score* (Equação 4.13).

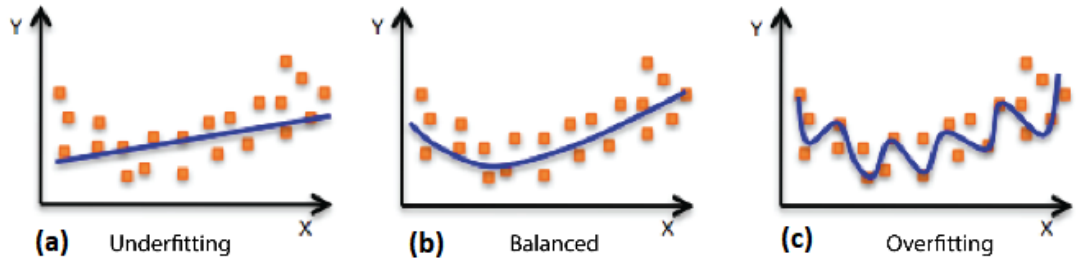
$$R^2 = 1 - \frac{SQ_{res}}{SQ_{tot}} \tag{4.13}$$

4.3.3 Problemas que podem ocorrer

Durante a aplicação das técnicas aprendizado de máquina alguns problemas podem ocorrer em relação aos modelos obtidos, estes podem apresentar duas características que podem invalidar a utilização deste na classificação ou regressão de novos resultados, estas são o ***underfitting*** (subajuste) e o ***overfitting*** (sobreajuste). Este serão descritos a seguir.

A Figura 3 exibem como as situação de *underfitting*(Figura 3 **(a)**) e *overfitting*(Figura 3 **(c)**) ocorrem em um modelo de regressão a curva azul exibe o comportamento das saídas de dados do modelo e os pontos em laranja são os dados de trino utilizados. A Figura 3 **(b)** exibe como deve ser o modelo ideal para o conjunto utilizado.

Figura 3 – (a) *Underfitting* em um conjunto de dados; (b) modelo de dados balanceado que consiga interpretar o conjunto passado e (c) *Overfitting* em um conjunto de dados. Fonte:(AWS, 2019)



4.3.3.1 Underfitting

O subajuste ocorre quando o modelo obtido após o treinamento não consegue descrever os dados apresentados, tendo um péssimo resultado tanto no conjunto de testes quanto no conjunto de treino. Esta é uma indicação que as características passadas como

a entrada do modelo, podem não estar diretamente relacionada as que se desejem predizer, talvez sendo necessário realizar um o novo estudo do domínio do problema abordado e alterando a modelagem do mesmo.

A Figura 3 **(a)** apresenta o comportamento de uma curva responsável por modelar um problema especifico, note que esta não é uma boa modelagem pois não consegue capturar a tendência da característica geral do problema.

4.3.3.2 Overfitting

O sobreajuste de dados ocorre quando o modelo obtido após o treinamento consegue descrever todo o conjunto de dados de treino, incluindo o ruído que o mesmo apresenta, desta maneira o aprendizado possui o foco apenas nos dados passados e em muitas situação não sendo capaz de modelar o sinal de interesse, ou seja, a característica geral do problema.

Para avaliar se uma modelagem apresenta *overfit*, devemos comparar o erro apresentado pelo modelo em relação aos dados de treino e aos dados de teste, quando o erro dos testes apresentam um valor muito superior aos dados de treino, isto indica que o modelo apresenta o sobreajuste, para realizar esta comparação de devemos utilizar a mesma métrica de erro para ambas as verificações.

Suponha que a métrica utilizada para o erro seja a informada na Equação 4.8, erro médio quadrado, com isso o erro de treino é representado pela Equação 4.14 com os dados de treino e o erro de teste é representado pela Equação 4.15 com os dados de teste, onde h é representado pelo modelo indutor obtido após o treino.

$$erro_{treino}(h) = mse(h) \tag{4.14}$$

$$erro_{teste}(h) = mse(h) \tag{4.15}$$

Quando a seguinte situação ocorre significa que o modelo apresenta *overfitting*.

$$erro_{teste}(h) >> erro_{treino}(h) \tag{4.16}$$

A Equação 4.16 informa quando o erro apresentado nos dados de treino é muito maior do que o erro apresentado nos dados de teste, quando este problema ocorre significa

que o modelo apresenta o sobre-ajuste, então ele consegue apreender todas as características dos ruídos apresentados no *dataset*.

A Figura 3 **(c)** exibe o comportamento de uma curva gerada por um modelo de regressão que apresenta *overfit* este apresenta bons resultados para os dados de treino porém aprende todas as características de ruido que o mesmo apresenta.

4.3.4 Aprendizado Supervisionado

O aprendizado supervisionado acontece quando o conjunto de dados apresentado ao sistema possui suas características rotuladas ou valoradas e com isso, o mesmo deverá ser capaz de aprender quais características levam aquele resultado, possibilitando a modelagem um mecanismo para recomendar os resultados de novos dados com base nas características extraídas do conjunto apresentado (MACHADO, 2015). A dificuldade principal para este tipo de aprendizado é encontrar um conjunto de dados que possuam comportamentos preditivos para ser possível criar um modelo computacional para mesmo(MONARD; BARANAUSKAS, 2003).

Nesta modalidade o algoritmo de treino, também conhecido com indutor, consiste em realizar a extração de um classificador ou regressor que seja capaz de entender as características de um conjunto de dados e com isso será possível utilizá-lo para predizer as informações para dados que ainda não possua uma classificação ou resultado conhecido.

Neste procedimento, os dados fornecidos para o aprendizado, podem ser descrito como um vetor de atributos juntamente com o seu rótulo, porém existem duas situações nesta problemática, os rótulos podem ser discretos ou contínuos, para os do primeiro tipo esta modelagem é conhecida como classificação de dados com os classificadores conhecidos e limitados, já para o segundo tipo a modelagem é conhecida como regressão de dados, sendo esta capaz de prever valores contínuos com base nos atributos do *datset* (MONARD; BARANAUSKAS, 2003).

4.3.5 Aprendizado Não-Supervisionado

O aprendizado não supervisionado acontece quando o conjunto de dados apresentado ao sistema não possui suas características rotuladas ou classificadas. Neste o algoritmo de aprendizado, também conhecido como indutor, processa o conjunto de dados com a

finalidade de formar agrupamentos, logo após é necessário verificar o significado de cada agrupamento obtido no contexto da problemática abordada.

4.4 Técnicas de *machine learnig*

4.4.1 Árvore de Decisão (Decision Trees)

Um exemplo de abordagem que utilizam o método do aprendizado supervisionado é a Árvore de Decisão. Este método constrói árvores que podem ser utilizadas para a classificação ou regressão a partir do conjunto de dados, dividindo espaços de entrada para encontrar fronteiras de decisões, estas divisões são feitas com base nos atributos existentes no conjunto de dados, ocorrendo de forma automática, esta técnica pode ser aplicada tanto para a classificação quanto para a regressão(QUINLAN, 1986). A seguir serão descritos alguns algoritmos que utilizam árvores de decisão.

Algoritmo ID3(*inductive decision tree*)(QUINLAN, 1986), um dos primeiro com a abordagem da árvore de decisão, consiste em um algoritmo guloso que visa buscar qual o melhor atributo que divide o conjunto de dados, ou seja identifica quais as principal característica do *dataset* que o divide em dois grupos, com isso são criados nós filhos representado cada grupo criado, sendo que neste será aplicada a mesma técnica porém no agrupamento que este pertence, este procedimento é realizado até que o *dataset* resultante só possuam uma classe. Um exemplo de árvore de decisão será apresentada na Figura 4 referente a tabela 5, está possui dados fictícios feitos com o intuito de exibir os tipos de *dataset* que o algoritmo ID3 têm a capacidade de tomar como dados de treinos para realizar a modelagem.

Tabela 5 – Histórico de crivo de empréstimo

Índice	Idade	Trabalho Fixo	Salário	Cancelou	Empréstimo
1	Média	Sim	Alto	Não	Sim
2	Baixa	Não	Baixo	Não	Não
3	Alta	Sim	Alto	Não	Sim
4	Alta	Não	Alto	Não	Sim
5	Baixa	Sim	Alto	Sim	Não
6	Baixa	Não	Baixo	Sim	Não
7	Média	Sim	Alto	Sim	Não
8	Alta	Não	Alto	Não	Sim
9	Média	Sim	Baixo	Não	Sim
10	Baixa	Sim	Alto	Não	Sim

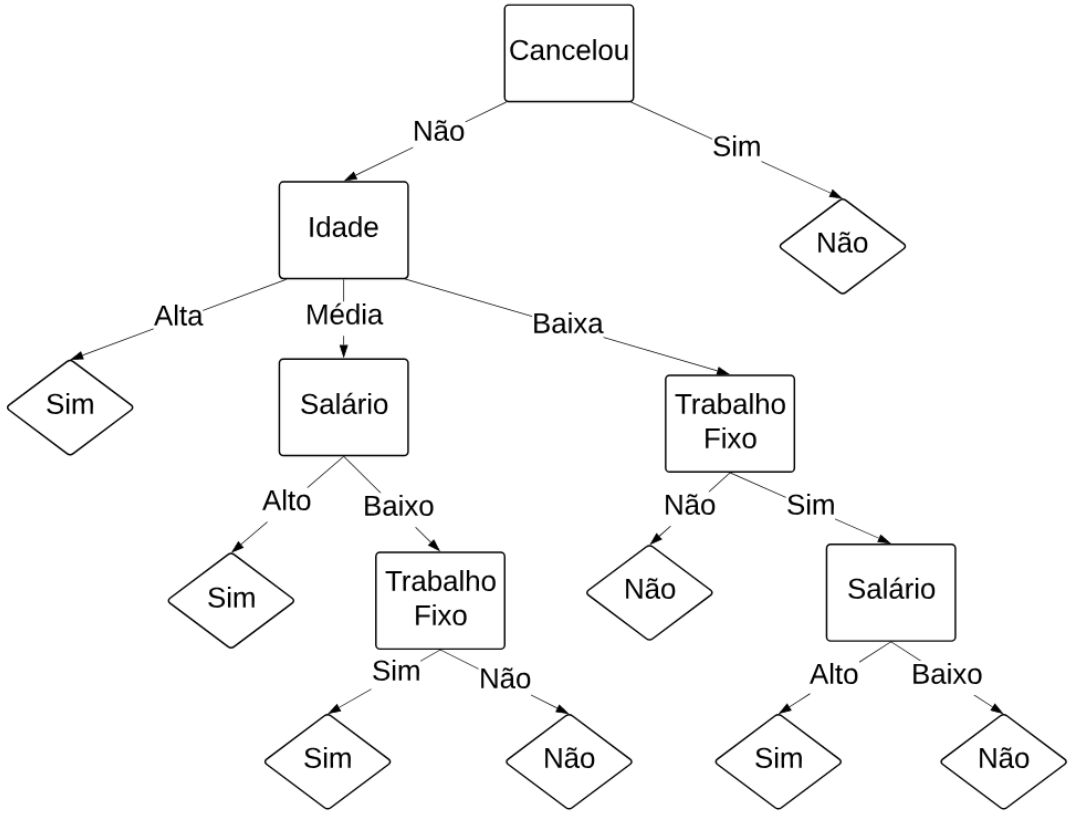


Figura 4 – Árvore de decisão para o problema de concessão de empréstimo do tabela 5.

O ID3 possui sua uma limitação por poder ser utilizado apenas com atributos categorizados (QUINLAN, 1986), ou seja, com este não é possível trabalhar com dados contínuos, somente com dados similares ao apresentado na tabela 5, porém em algumas situações estes podem ser modelados de forma discreta indicando faixas de valores.

4.4.2 Random Forrest

Este é um dos algoritmo de aprendizagem supervisionada utilizando tanto para classificação e quanto para regressão, isto por não apresentar dificuldade para encontrar uma configuração dos hiperparâmetros e também por apresentar bons resultados. No aprendizado de máquina o *Random Forrest* é categorizado com um método *ensemble* por possuir múltiplos algoritmos de aprendizado interno para obter um melhor modelo preditivo. Este algoritmo cria um conjunto de árvores de decisão, este processo ocorre de maneira aleatória, como o próprio nome ja diz.

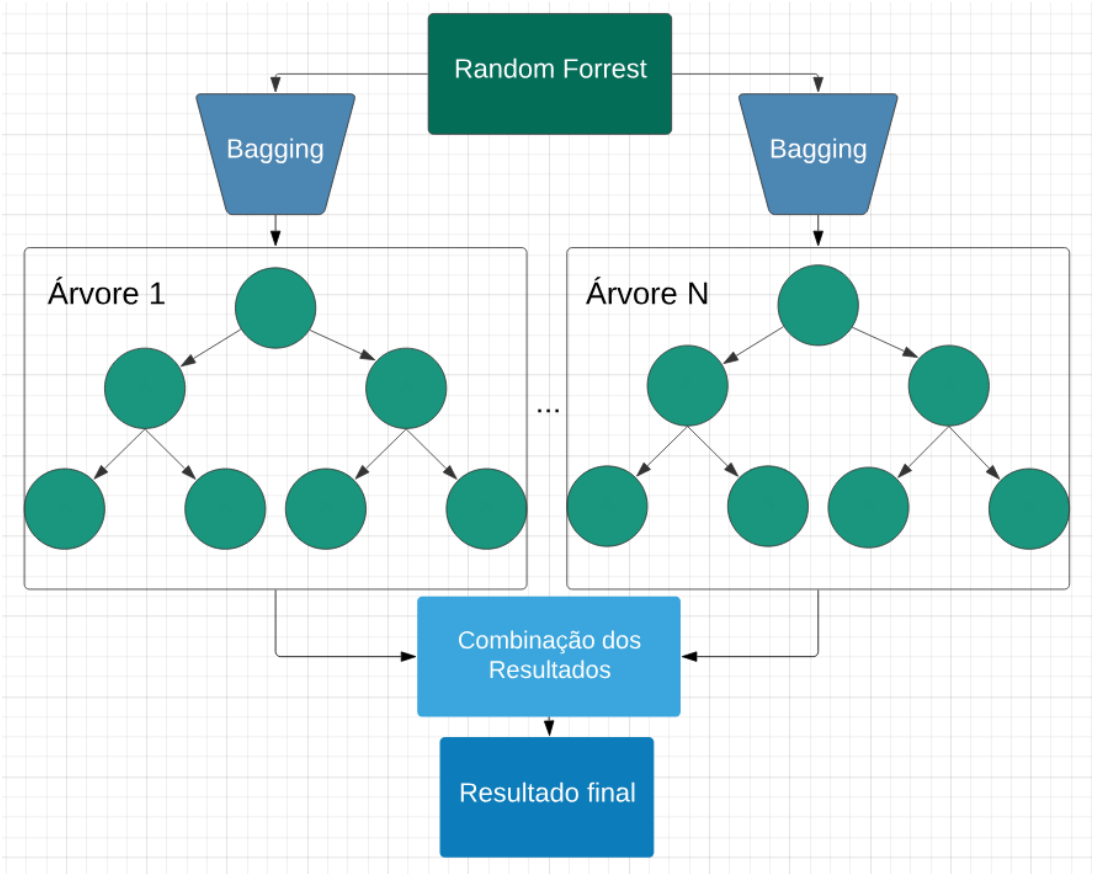
A execução de um algoritmo utiliza árvores de decisão, estas foram explicadas anteriormente, este cria diversas divisões no *dataset* conforme as restrições geradas pelas árvores, no fim é obtido um modelo classificação ou regressão. No algoritmo *Random Forrest* é criado um conjunto árvores de decisão, destas são computados quais resultados cada uma obtém e logo mais é verificado qual classificação ou regressão pode ser apresentada, este é considerado o resultado principal obtido pelo algoritmo.

O processo de treino deste algoritmo utiliza o método de *bagging*, uma acrônimo dos termos *bootstrap* e *aggregating*, este utiliza o *dataset* para criar combinação diferentes de dados em partes iguais e em seguida são passados como entrada para as árvores de decisões internas, no final do algoritmo são utilizados métodos de combinação apropriados para verificar qual será o dado final predito pelo modelo.

O resultado principal fornecido por este modelo ocorre de duas formas com métodos específicos para regressão e para classificação. Quando usado para classificação o resultado final apresentado será aquele que possuir a maior frequência de amostragem, ou seja aquele classificação informada pela maioria das árvores (BREIMAN, 2001). Quando usado para regressão o resultado final exibido será a média dos resultados informados para cada árvore de decisão interna na floresta, com isso é possível notar que quanto maior for a quantidade de árvores, maior será a precisão do resultado.

A figura 5 exibe um diagrama o que foi dito anteriormente.

Figura 5 – Exibição do algoritmo *Random forrest* com suas árvores de decisões.



Uma das principais vantagens do algoritmo *Random Forrest* em relação ao *Decision Three* é em com o *overfitting*, as árvores de decisões profundas tendem a sofrer este tipo de características, o que ocorre com menos frequência no algoritmo *Random Forrest*.

Os hiperparâmetros citados anteriormente são as variáveis que influenciam no processo de treino, para encontrar um bom modelo é necessário identificar uma configuração ideal destes parâmetros. Os hiperparâmetros mais importantes para uma árvore de decisão são:

- 1 - Profundidade da árvore, que define qual será o tamanho máximo das árvores de decisão presentes no algoritmo.
- 2 - Número de estimadores, este indica o número máximo de árvores de decisões existentes, este valor influência diretamente no resultado obtido pelo preditor, quanto maior o valor mais precisa será a predição porém o processo de computação será mais lento.

- 3 - Número de folhas, que indica o valor minimo possível de folhas existentes uma árvore.
- 4 - Máximo de características, que indica o valor máximo de caracteristicas de entradas que devem ser analisadas.

Este algoritmo pode ser aplicado em diversas situações, sendo utilizado em predição de dados relacionados o setor bancário, no mercado financeiro, na medicina e em outras áreas a fins.

4.5 Análise de Solo

A análise do solo é um processo necessário para a identificação de características do solo. Segundo o manual disponibilizado pela EMBRAPA, (CLAESSEN, 1997), este procedimento consiste em três etapas, a análise das propriedades físicas, químicas e mineralógicas.

Na análise física, primeiro são definidos os métodos de extração do solo e preparação da amostra a ser verificada, logo após diversos testes são aplicados para identificar a densidade, seu nível de plasticidade, a porosidade, grau de floculação, grau de contração, sua condutividade hidráulica e outras características.

Na análise química são identificados algumas características, fundamentais para uma recomendação de calagem e adubação precisa, nesta são informados os deficites e as riqueza de nutrientes. Neste passo é analisado o ph do solo, o carbono orgânico, o nitrogênio total, a acidez do solo, a capacidade de troca de cátions (CTC), o hidrogênio extraível, a saturação de bases (SB), a saturação por alumínio, saturação por sódio, os níveis de fósforo.

Na análise mineralógica, são realizados processos para a verificação de minerais que possam estar presentes no solo, este procedimento não possui influência na recomendação.

Este processo é realizado por laboratórios específicos, o cliente será o responsável por realizar coletas do solo de forma espaçada na região que se deseja conhecer as características do solo, logo após estas amostras deverão ser misturadas de forma uniforme e em seguida parte delas deverão ser enviadas para os laboratórios realizarem os procedimentos de análise informados anteriormente.

5 Sistema AdubaTec

Este é um sistema de recomendação de calagem e adubação, que visa fornecer ao produtor rural uma forma prática de obter as informações referentes às necessidades de nutrientes do solo e a maneira ideal para a calagem do mesmo. Estas informações são geradas a partir dos dados fornecidos pelo produtor, estes são inerentes à cultura e a análise do solo, sendo que esta deve ser realizada de forma prévia em um laboratório que forneça o serviço.(ARAÚJO; PONTES; CRESPO, 2018)

O sistema apresenta um formulário que o produtor deverá preencher com as informações referentes a sua cultura, variedade, sistema de cultivo, estágio a análise do solo e outras características, a partir destas é realizado uma busca na base de dados de recomendações de nutrientes e também de uma forma ideal para a calagem do solo que satisfaçam todas as características informadas.

A forma que o sistema se comporta em relação ao cadastro de novos dados é o diferencial que o mesmo apresenta em relação a sistemas que possuam a mesma finalidade. Este possui uma forma de cadastro parametrizável que visa a se modificar de acordo com as necessidades do usuário, sendo que podem ser adicionados modificadores específicos que contribuam como uma a melhor especificação uma recomendação para uma cultura especifica, estes podem ser variados como idade, espaçamento, clima, densidade de plantio entre outros.

Este sistema pode ser classificado como um sistema especialista, uma subárea da inteligência artificial, por ser capaz de possuir todas as características necessárias para solucionar uma problemática especifica, assim substituindo a necessidade constante de um especialista em agronomia para informar as recomendações de nutrientes com base em informações oferecidas.

5.1 Motivação

Com a demanda de informações precisas para as quantidades necessárias de nutrientes para adubação e também a forma ideal para a calagem do solo por parte do produtor rural, surgiu a necessidade de um sistema que fornecesse uma base de dados confiável e

fosse acessível para o produtor rural.

Também, pela necessidade por parte dos pesquisadores da área agrônoma de um sistema que mapeie os dados de recomendação de nutrientes existentes na literatura, este possibilita o acesso às informação de forma ágil e prática.

O sistema especialista AdubaTec, é uma ferramenta que irá auxiliar o produtor rural e o pesquisador, tendo como sua funcionalidade principal a recomendação da adubação necessária para determinada cultura, que conste em sua base de dados, em qualquer que seja os seus estágios.

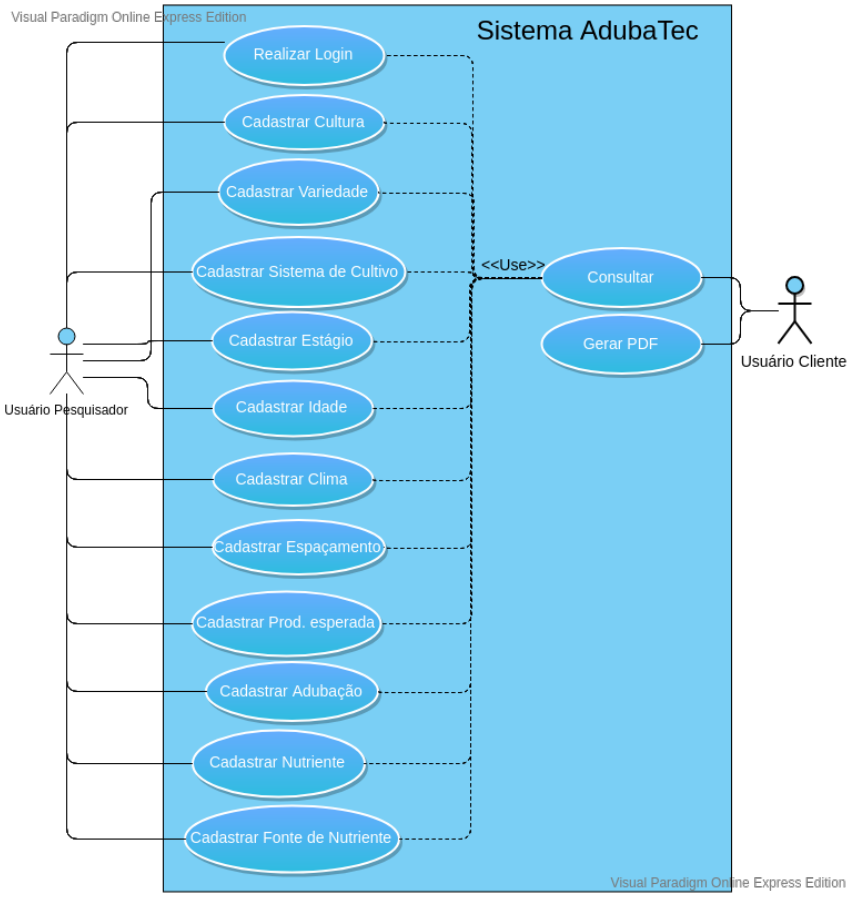
5.2 Desenvolvimento do projeto

Na idealização do projeto foi definido que o mesmo seria um sistema web com as funcionalidades que serão apresentadas a seguir:

- 1. Apresentar, a partir de dados informados pelo produtor rural inerentes a cultura a ser cultivada e a análise do solo, a forma ideal para a calagem do solo e as recomendações de nutrientes necessárias para a adubação do solo.
- 2. Fornecer um relatório em PDF que consiga informar de forma simplificada o produtor rural os resultados obtidos e também observações para que o mesmo realize, tanto a calagem como a adubação de forma correta para as características informada.
- 3. Fornecer um forma de cadastro que seja parametrizável, que vise englobar as recomendações de adubação existentes em diferentes bibliografias, sem que haja necessidade de um especialista em TI remodelar o sistema para que o mesmo seja capaz de aceita-la.
- 4. Armazenar todos os dados de consulta dos usuários, com a finalidade de futuramente utilizá-los para realizar um mapeamento do solo nas regiões em que haja uma análise cadastrada.

  A Figura 6 exibe o principal caso de uso do sistema.

Figura 6 – Caso de uso do sistema AdubaTec



5.2.1 Criação dos softwares do sistema

Para o sistema foi desenvolvido uma página web responsável pela interação com o cliente, sendo esta página o *front-end* do sistema, uma *web service* capaz de receber os dados do *front-end* e processa-los de acordo com as regras de negócio propostas nos requisitos do sistema, esta é responsável pela conexão com o banco de dados com a finalidade de realizar cadastros e consultas no mesmo. A *web service* junto com o banco de dados são caracterizados como o *back-end* do sistema.

A página web é responsável pela interação com o cliente sendo o mecanismo que irá receber os dados informados do usuário e repassá-los para o *back-end* e também é responsável por exibir as informações retornadas pelo mesmo. Este apresenta as funcionalidades que foram informadas anteriormente.

O *frontend* foi desenvolvido com a tecnologia Angular na versão 4, esta é uma plataforma para desenvolvimento web que utiliza a linguagem Typescript, foi desenvolvida pela Google, sendo uma das ferramentas de desenvolvimento web mais populares

atualmente.

5.2.1.1 Web Service

A *webservice* foi desenvolvida na linguagem de programação Java. Esta é responsável por implementar todas as regras de negócio, realizar todas as transações do sistema e também realiza as consultas e cadastros no banco de dados, a *web service* foi desenvolvida no parão REST, utilizando apenas os métodos *GET* e *POST*, possuindo *endpoints* que serão consumidos pelo *front-end*.

Também foi utilizado o *framework Hibernate*, que possibilitou a criação do banco de dados através das classes criadas na *web service*, sendo que somente as classes que eram pertinentes na modelagem foram refletidas como entidades no banco de dados com seus relacionamentos específicos, desta forma é possível utilizar todos os paradigmas de orientação objetos para a modelagem do banco de dados, realizar as consultas e os cadastros no mesmo.

5.2.1.2 Banco de dados

Para o sistema foi utilizado o banco de dados *PostgreSQL*, por ser uma ferramenta gratuita que utiliza a linguagem SQL. Este é responsável por armazenar os dados vindos da *web service*, tantos das consultas quanto os oriundos dos cadastros.

Para os dados cadastrados no sistema, foi utilizado o livro Recomendações de Calagem e Adubação para Abacaxi, Acerola, Banana, Laranja, Tangerina, Lima Ácida, Mamão, Mandioca, Manga e Maracujá dos autores Ana Lúcia Borges e Luciano da Silva Souza. Este apresentava dados necessários que foram utilizados para a modelagem do banco de dados do sistema (BORGES, 2009).

O livro contém tabelas com recomendações de calagem e adubação para as culturas informadas anteriormente, possuindo recomendações dos macronutrientes nitrogênio(N), fósforo(P), potássio(K) entre outros, com base nos nutrientes presente no solo.

Também foram analisados outros livros com a finalidade de realizar a modelagem do banco de dados de forma que fosse capaz de comportar o cadastro de recomendações oriundas de fontes de conhecimentos distintas da utilizada.

6 Sistema de predição

O sistema de predição proposto neste trabalho será responsável por fornecer ao produtor rural um mecanismo auxiliar de análise do solo, não necessitando de processo laboratoriais para realizar este procedimento, retornando os valores dos nutrientes de forma ágil e prática.

Este pode ser adicionado ao sistema de recomendação de calagem e de adubação do solo, o AdubaTec, com isto os usuários da plataforma só deverão passar as informações de quais serão as culturas a serem plantadas e qual o local da plantação. Os modelos obtidos neste trabalho são indicados para a utilização de sistemas como o AdubaTec, pois os valores de recomendação, são constantes em uma faixa, com isso, os erros apresentados pelo valor predito em relação ao valor real podem ser desconsiderados se as recomendações obtidas forem iguais. A seguir é apresentado um exemplo do que foi dito.

A recomendação de adubação no solo do com base no nutriente potássio para a cana de açúcar e apresentada na Tabela 6, suponha que o modelo de predição tenha predito o valor do nutriente 0,11 $cmol_c/dm^3$ e seu valor correto seria 0,06 $cmol_c/dm^3$, o sistema retornou um valor 83% maior que o valor real, porém para ambos os valores a recomendação será a mesma podendo ser $90, 110, 130, 150$ kg/ha, pois ambos são menores que 0,15 $cmol_c/dm^3$. Este será um dos testes realizados para validar o sistema.

Tabela 6 – Recomendação de adubação para o potássio. Fonte:(RAIJ, 1996)

Produtividade esperada	Potássio no Solo $cmol_c/dm^3$		
	< 0.15	0.15 - 0.30	> 0.30
t/ha	K_2O kg/ha		
< 60	90	60	30
60 - 80	110	80	50
80 - 100	130	100	70
> 100	150	120	90

O sistema de predição proposto neste trabalho será responsável por informar ao usuário os valores dos nutrientes cálcio, magnésio, sódio e potássio, isto com base em algumas informações de entradas, estas serão discutidas neste capítulo.

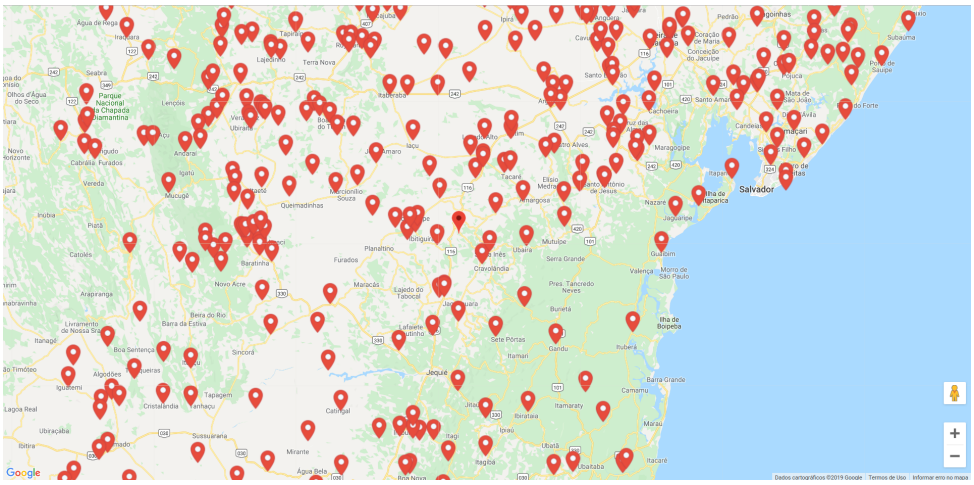
Este software apresentam dois subsistemas, um responsável por ser um mecanismo de interação com o usuário, a página web e outro por ser o mecanismo de predição das

características do solo, um servidor. O servidor será responsável por receber requisições que possuirão as características necessárias para a predição dos nutrientes, estas podendo ser enviadas por qualquer sistema, também por carregar os melhores modelos de predições obtidos para cada nutriente e predizer os valores destes com base nos dados passados pela requisição. O servidor foi desenvolvido na linguagem de programação *python* e pode ser integrado a outros sistemas, como no caso do sistema AdubaTec, logo o mesmo será capaz de usufruir das informações fornecidas pelo software de predição. Desta forma o sistema AdubaTec, possuirá um mecanismo que consiga sugerir as informações dos nutrientes presentes no solo para o usuário, de forma que o mesmo não necessita realizar processos de análise química do solo que podem levar um tempo considerável para fornecer os resultados.

A página web é responsável por ser um mecanismo de interação com o usuário, apresenta apenas um componente, sendo um mapa que utiliza a ferramenta Google Maps ®, o mesmo exibe pontos onde foram coletados amostras de solo e ao clicar em um ponto vazio do mapa o sistema criar uma nova marcação que exiba os dados preditos pelos modelos computacionais obtidos. A página web foi desenvolvida na tecnologia Angular e se comporta como um componente único, possui esta característica para facilitar a integração com outros sistemas.

A Figura 7 exibe a interface da pagina web com as marcações de todos as amostras obtidas pelo projeto Radam Brasil.

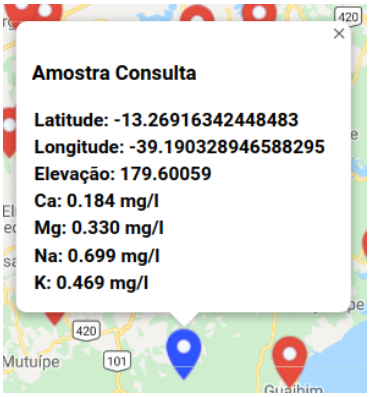
Figura 7 – Página web do sistema de predição



A Figura 8 exibe como são exibidas das informações dos nutrientes após a predição. Para este processo, só é necessário realizar o clique em uma posição no mapa para que a

página web solicite ao servidor as informações dos nutrientes na localidade.

Figura 8 – Página web do sistema de predição



A única restrição de uso deste sistema, consiste na limitação da região que o mesmo pode oferecer as predições dos nutrientes, este trabalho considerou apenas amostras de solo extraídas do estado da Bahia, logo é indicado a utilização apenas nas regiões do mapa que possua marcações de amostras.

A seguir serão apresentadas algumas informações sobre qual o conjunto de dados utilizado, quais estratégias foram adotadas para a obtenção dos modelos, como foram aplicadas as técnicas de aprendizado de máquina e como consistiu o processo de validação do modelos obtidos.

6.1 Conjunto de dados

Os *dataset* utilizado para a aplicação das técnicas de aprendizado de máquina foi o apresentado pelo projeto RADAM Brasil (LIMA et al., 1984), o projeto mapeou diversas características do solo do país, porém este trabalho ficou restrito apenas aos dados de análise do solo do estado da Bahia. Apesar do projeto possuir informações sobre diversas características do solo e a os pontos de latitude e longitude das amostras coletadas, nem todas são pertinentes para este trabalho, as informações utilizadas foram a Latitude, Longitude, Cálcio, Magnésio, Sódio e Potássio.

Mesmo que o projeto RADAM Brasil, possua diversas informações do solo o mesmo não apresenta os dados necessárias para obter a modelagem computacional de todas os nutrientes de entrada do sistema AdubaTec, entretanto, nem todos os atributos são necessários na hora da recomendação da calagem e da adubação, o sistema oferece uma

opção de despensa do dado de entrada quando não há informação necessária. Esta situação será sanado quando o sistema AdubaTec possuir dados suficientes de consultas de usuários, estas que irão conter dados de análise do solo, com isso poderá ser aplicado os algoritmos apresentados neste projeto para obter novos modelos preditivos, também sendo mais um critério de comparação com os resultados apresentados neste documento.

A seguir será aprestada a Tabela 7 que contém uma pequena amostragem dos dados utilizados. Os valores dos nutrientes apresentados na mesma possuem a unidade mg/l, os modelos são treinadas com esta unidade, porém após a predição o dado será convertido para $cmol_c/dm^3$, esta conversão é feita por um fator linear não propagando o erro obtido na predição. O procedimento de conversão é necessária pelo fato do sistema trabalhar com a unidade de $cmol_c/dm^3$ para estes nutrientes.

Tabela 7 – Amostra dos dados utilizados para a criação do modelo computacional de predição. Fonte (LIMA et al., 1984)

Amostra	(mg/l)					
	Latitude	Longitude	Cálcio(Ca)	Magnésio(Mg)	Sódio(Na)	Potássio(K)
CH-01	-14.87583	-39.90139	120	141	172	4.4
CH-26	-14.75666	-39.25444	150	83.7	280	8.2
CH-27	-14.34361	-39.16416	317	150	260	8.1

6.2 Abordagens

Para resolver a problemática apresentada neste trabalho foram realizadas diversas abordagens para obter os modelos de predições dos nutrientes. Estas consistiram em encontrar quais seriam as melhores informações que poderia ser utilizadas como características de entrada para o modelo.

6.2.1 1ª Abordagem

A primeira abordagem consistiu em utilizar os dados da latitude e longitude como os parâmetros de entrada para o sistema, estas informações já estão presentes no conjunto de dados do projeto RADAM Brasil, porém todos modelos obtidos com base nessas características apresentaram o *undefitting*, possuindo um elevado erro em todos os conjuntos, treino, teste e validação.

A principal indicativa do resultado falho desta abordagem é a falta de dados de entrada suficientes ou significativamente correlacionados pra criar modelos de predições com variáveis dependentes.

6.2.2 2ª Abordagem

A segunda abordagem consistiu em adicionar mais um atributo para compor as características de entrada dos modelos, o atributo escolhido foi a elevação do solo em relação ao nível do mar.

O conjunto de dados escolhido não apresenta este tipo de informação, então foi necessário o desenvolvimento de um mecanismo de requisições web que acessasse uma *api* responsável por retornar o valor da elevação em função da latitude e longitude e de forma automática preenchesse o conjunto de dados utilizados com estas informações.

O *script* de requisições foi desenvolvido com a tecnologia *NodeJS* e pode ser encontrada no Anexo I, a *api* utilizada foi a JWangMaps Elevation API[1], esta pode ser acessada de forma gratuita.

Apesar de ter adicionado esta nova característica os modelos obtidos não conseguiram fornecer bons resultados, para nenhum dos modelos utilizados, apresentando o *undefitting* em todos os resultados.

Também nesta abordagem, foi constatado como motivo principal dos resultados insatisfatórios a falta de dados de entrada suficientes ou significativamente correlacionados pra criar modelos de predições com variáveis dependentes.

6.2.3 3ª Abordagem

Na terceira abordagem foi adicionado junto com as características das abordagens anteriores o nutriente cálcio, com isso o sistema começou a convergir e conseguiu realizar a predição dos outros nutrientes, sódio, magnésio e potássio.

Mesmo que tenha sido possível encontrar modelos para os outros nutrientes, se tornou necessário o conhecimento do cálcio para realizar esta predição, isto tornaria inviável a proposta inicial, esta é a predição dos valores dos nutrientes do solo com base em

[1] Mais informações podem ser obtidas no seguinte endereço https://www.jawg.io/docs/apidocs/elevation/.

informações, como no caso da latitude, longitude e elevação. O cálcio só pode ser obtido com a previa análise do solo.

Para tentar contornar esta problemática, foi desenvolvido um algoritmo utilizando conceitos do *K-Nearest Neighbors* para regressão, porém com a diferenças. O algoritmo desenvolvido irá informar o valor do cálcio, utilizando K amostras próximas, gerando pesos para cada amostra de forma que estes sejam inversamente proporcionais a distância e que somados ficam igual a 1.

Este código foi desenvolvido diretamente na página web, na linguagem *typescript*, o trecho a seguir é responsável por encontrar os vizinhos próximos, com base na latitude e longitude utilizando a distância euclidiana para realizar este processo.

```
getKNeighbors(k, cord) {
  let neighbors = []
  for (let a of this.amostras) {
    const cord2 = a.cord
    const dist = this.calulateEuclidianDistance(cord, cord2)
    if (neighbors.length < k) {
      neighbors.push({ amostra: a, dist: dist })
    } else {
      for (let index in neighbors) {
        const d = neighbors[index].dist
        if (dist < d) {
          neighbors[index] = { amostra: a, dist: dist }
          break
        }
      }
    }
  }
  return neighbors
}
```

O cálculo da distância euclidiana pode ser obtido com a Equação 6.1, onde x e y serão as latitudes e longitudes das coordenadas um e dois.

$$dist = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{6.1}$$

Para encontrar os pesos foram utilizadas as estratégias apresentadas nos códigos a seguir, este será proporcional ao inverso da distância, com isso os vizinhos mais próximos

possuirão mais influência na amostra.

```
getWeights(neighbors) {
  let weights = []
  for (let n of neighbors) {
    weights.push(1 / n.dist)
  }
  let total = this.sumArray(weights)
  for (let i in weights) {
    weights[i] = weights[i] / total
  }
  return weights;
}
```

A Equação 6.2 exibe como é feito o processo para o cálculo do valor do cálcio predito, esta utiliza as informações do nutriente nas amostras vizinhas(Ca_i) e seus pesos (w_i).

$$Ca_pred = \sum_{i=0}^{k} Ca_i * w_i \tag{6.2}$$

O trecho de código a seguir exibe como é feito o o processo do cálculo da Equação 6.2.

```
getKPred(neighbors,weights){
  let caPred = 0;
  for (let i = 0; i < neighbors.length; i++) {
    caPred += neighbors[i].ca.value * weights[i]
  }
  return caPred
}
```

O valor do cálcio predito será próximo aos valores dos vizinhos, este tipo de predição se mostrou aceitável após a realização de testes onde foram comparados os valores reais e os preditos, sendo mais recomendado a predição deste quando existem amostras próximas. Com os testes foi notado que o valor ideal de vizinhos a serem analisados seriam 5, logo $k = 5$.

Inicialmente foi utilizado a mesma ideia para predizer os outros nutrientes, porém os resultados não se mostraram aceitáveis, possuindo um erro médio quadrado elevado, para os demais.

Com essa abordagem é fornecido uma forma de obter o valor do cálcio, logo o sistema apresentará todos as informações necessários para a entrada dos modelos de predição dos outros nutrientes.

Apesar de ser possível realizar a predição do cálcio pelo método informado existem duas situações que podem ocorrer na hora da predição, isto por ser utilizado uma média ponderada para obter os resultados, estas são exibidas a seguir.

A primeira situação ocorre quando é solicitada a predição de um dado que esteja em uma região circunvizinha de outras amostras existentes, nesta situação o sistema irá retornar o valor com uma tendência correta, seja de crescimento ou decrescimento. A Figura 9 exibe o que foi dito.

A segunda situação ocorre de maneira contrária da anterior, então é solicitado uma predição onde as amostras mais próximas não rodeiam o alvo, logo o resultado obtido não seguirá a tendência dos dados, e estará mais próximo do resultado médios. A Figura 10 exibe o que foi dito.

Figura 9 – Situação ideal para predição do cálcio.

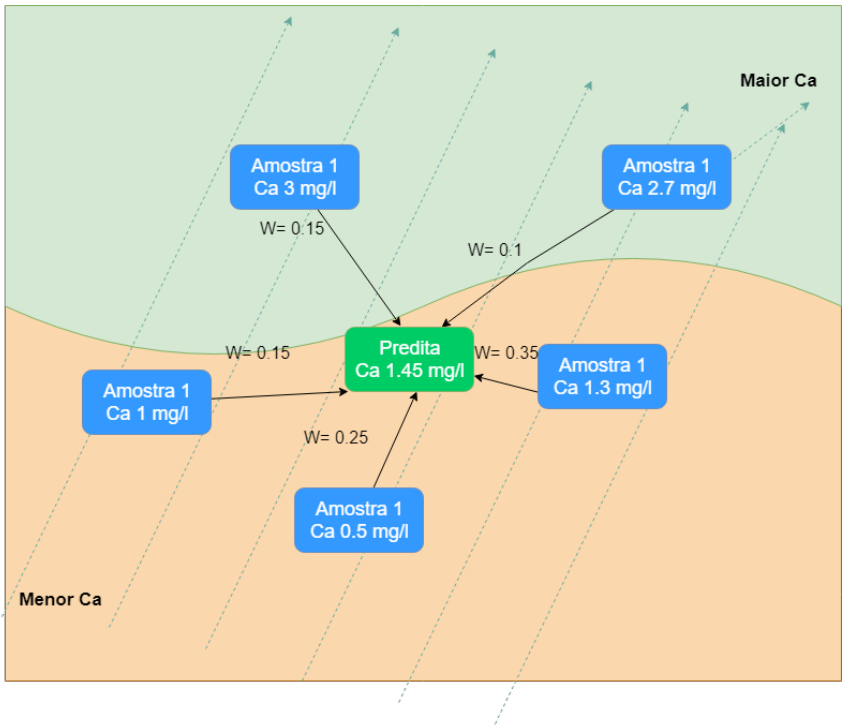
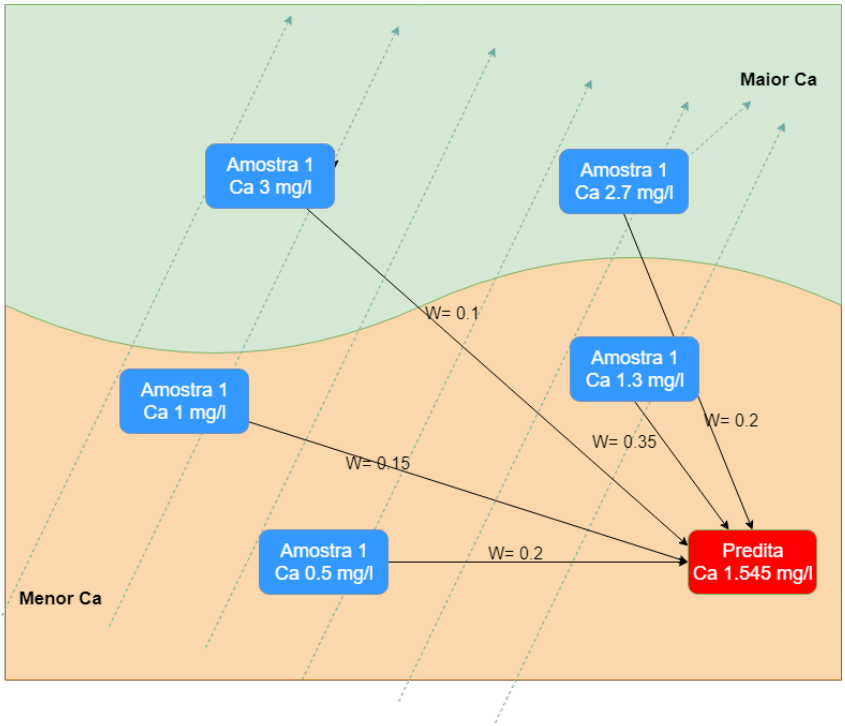


Figura 10 – Situação incorreta para a predição do cálcio.



6.3 Aplicação das Técnicas

No desenvolvimento dos códigos necessários para a aplicação das técnicas de inteligência artificial foi necessário o conhecimento de várias tecnologias que são comumente utilizadas na área de *data science* e *machine learning*. Neste projeto foi utilizado a linguagem de programação Python na versão 3.7 juntamente com diversas bibliotecas que facilitam o manuseio de dados e também a aplicação de técnicas de inteligência artificial, a seguir serão descritas algumas das bibliotecas utilizadas.

6.3.1 Ferramentas utilizadas

No momento de importação dos dados foi necessário realizar o manuseio de um arquivo para realizar a leitura do *dataset* para isto foi utilizado a biblioteca ***pandas***, com esta é possível abrir arquivos do formato CSV e manipula-los(MCKINNEY, 2011).

Também para manipulação dos dados, foi utilizado a biblioteca ***numpy***, com esta é possível realizar operações no conjunto de dados, trabalhando nos mesmos como se fossem matrizes (WALT; COLBERT; VAROQUAUX, 2011). Esta foi decisiva para realizar operações matemáticas no *dataset*.

Foi utilizado a biblioteca ***sklearn***, uma abreviação de *Scikit-Learn*, esta possui diversas ferramentas de *machine learning* em Python (PEDREGOSA et al., 2011). Desta biblioteca foi utilizado modelos de aprendizado supervisionado como, *Random Forrest Regressor*, *Decision Tree Regressor*, *Gradient Boosting Regressor*, *Support Vector Regressor* e também métricas de avaliação dos resultados como *R2 Score* e *Mean Squared Error*.

Para a exibição dos resultados foi utilizado a biblioteca ***matplotlib*** esta é responsável por criar um gráfico dos dados (HUNTER, 2007), este foi útil para verificar a curva de aprendizagem do modelo, comparar os dados preditos e os originais.

Após a obtenção dos melhores modelos para cada nutriente é necessário salvá-los para estes poderem ser utilizados a qualquer momento sem a necessidade realizar o treino novamente, para isto foi utilizado a biblioteca ***joblib***.

6.3.2 Desenvolvimento

6.3.2.1 Importação do conjunto de dados

O trecho a seguir é responsável por ler o arquivo ***DataSet.csv*** com as colunas colunas necessárias para o modelo e logo após criar a variável *dataset* que é uma matriz ***numpy*** dos dados importados, por fim os dados que serão utilizados na modelagem computacional são normalizados em uma faixa de 0 e 1.

```
1 import pandas as pd
2 import numpy as np
3 csvFile = pd.read_csv("DataSet.csv", usecols=[1, 2, 3, 4, 5, 6, 7])
4 dataSet = np.array(csvFile.values)
5 normalizeDataSet(dataSet)
```

6.3.2.2 Divisão dos dados

O trecho de código a seguir exibe como é feito o processo de separar o *dataset* em duas partes, uma para o treino e a outra para o teste do modelo, a função ***train_test_split***, é responsável por realizar esta divisão, a mesma tem como parâmetros os dados de entrada que serão a latitude, longitude, elevação e o cálcio (variável X no

algoritmo) e os dados a serem preditos que serão os nutrientes (variável Y no algoritmo), também o ***test_size*** que define o quanto do *dataset* será divido em dados de teste.

```
1 from sklearn.model_selection import train_test_split
2 X = dataSet[:,:4]
3 y = dataSet[:,:target_column]
4 X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2)
```

O conjunto de teste obtido no código anterior, só será utilizado após o obtenção dos parâmetros ideias para o modelo, sendo este útil para realizar uma avaliação final do modelo obtido, predizendo dados intocados pelo mesmo durante o processo de treino.

6.3.2.3 Criação do modelo

Podem ser criados modelos com os seus hiperparâmetros definidos da forma indicada pelo algoritmo, ou adicionadas de forma aleatória. O trecho de código a seguir exibe o processo que deve ser feito para realizar o treino e a predição dos dados em um modelo de aprendizado de máquina. Após serem obtidos os valores de treino e os de teste é possível criar o modelo treina-lo e predizer novos dados.

O modelo do exemplo foi o ***RandomForestRegressor***, neste exemplo apenas alguns dos hiperparâmetros foram utilizados, o ***n_estimators***, o ***max_depth*** e o ***random_state***.

```
1   from sklearn.ensemble import RandomForestRegressor
2   model = RandomForestRegressor(n_estimators=100,max_depth=50,random_state=0)
3   model.fit(xTrain,yTrain)
4   yPred = model.predict(xTest)
```

O processo de treino deve ser realizado com um conjunto de dados que apresente todas as características do problema, para isto deve ser realizado um técnica de balanceamento da dados.

O processo informado, foi apresentado com um intuito de ilustrar os passos básicos do treino e teste dos modelos de aprendizado de máquina utilizados, porém estratégias robustas foram utilizadas para encontrar os melhores modelos, como o *tuning*, este consiste

em tunar os hiper-parâmetros de um modelo com a finalidade de obter as melhores configurações.

6.3.2.4 Balanceamento de dados

Para realizar o balanceamento de dados, foi utilizado um estratégia que escolhe dados aleatórios no conjunto de dados, e os utiliza para o treino, logo após são verificados com um mecanismo de validação cruzada para identificar se o modelo apresenta bons resultados.

Uma situação importante, é que nesse procedimento são criados dados aleatórios de treino e de teste. Os dados de treino são passados para o modelo escolhido e nestes são aplicadas as técnicas de validação cruzada com a finalidade de identificar se aqueles dados de treino podem gerar modelos interessantes.

O código a seguir exibe como pode ser feito este procedimento. A função retorna uma semente randômica, com esta é possível replicar os dados de treino que apresentou bons resultados com a validação cruzada e obter os dados de teste, isto com a utilização da função ***train_test_split***. Note que este procedimento realiza um rodízio no conjunto de dados para verificar quais dados são os indicados para o treino, mesmo que em algum momento do rodízio os dados de teste resultantes tenha sido utilizados em procedimentos de validações anteriores, no final este modelo não será o utilizado. O modelo final a ser obtido será treinado e validado apenas com os dados de treino, e não terá conhecimento dos dados de teste até a avaliação final.

```
1  from sklearn.cross_validation import cross_val_score
2  def getBestSeed(X,y,faixa):
3    maior_score = 0
4    seed = 0
5    for i in faixa:
6      X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2,random_state=i)
7      model = RandomForestRegressor(random_state=0,max_deph=None,n_estimators=100)
8      scores =cross_val_score(model,X_train, y_train, scoring='r2', cv=10)
9      score = scores.mean()
10     if score>maior_score:
11       maior_score = best_score
12       seed = i
13   return seed,maior_score
```

Após serem encontrados os melhores dados para o treino, devemos tunar os parâmetros do modelo, para melhorar os resultados.

6.3.2.5 Tuning

Após a escolha de um modelo é necessário identificar a melhor configuração dos seus hiperparâmetros para o problema a ser resolvido. A função ***GridSearchCV*** da biblioteca ***sklearn***(PEDREGOSA et al., 2011), pode ser utilizada para encontrar estes parâmetros, esta é capaz de realizar todas as combinações de parâmetros informados pelo usuário e extrair o melhor modelo, também possui um mecanismo de validação cruzada, que divide o conjunto de dados de treino em k partes, separando uma delas para a validação e as outras para o treino do modelo, o algoritmo vária os conjuntos de validação e de teste de modo que o modelo consiga treinar com todos os dados e também testar todos os dados. A avaliação deste modelo é dada pela média dos resultados obtidos durante este processo, isto para a métrica escolhida.

O código a seguir exibe uma função genérica que utiliza o ***GridSearchCV*** para tunar os parâmetros de um modelo. Possui como entrada o modelo escolhido, os possíveis parâmetros e os dados de treino.

```
1 def tuningParameters(model,param_grid,X_train,y_train):
2     reg =GridSearchCV(model,cv=10,param_grid=param_grid,n_jobs=-1,scoring='r2')
3     reg.fit(X_train,y_train)
4     return reg.best_estimator_,reg.best_params_,reg.best_score_
```

O código a seguir exibe os parâmetros utilizados para os modelos *RandomForestRegressor*, *GradientBoostingRegressor* e *SupportVectorMachine*, estes apresentaram os melhores resultados.

```
1 # Parametros RandomForestRegressor
2 param_grid_full = {
3   'bootstrap': [True],
4   'max_depth': [None,10,50,100],
5   'max_features': ['auto','log2'],
6   'min_samples_leaf': [1,2,3,4],
7   'n_estimators':[10,50,100,200],
8 }
```

```
# Parametros GradientBoostingRegressor
param_grid_full = {
  'n_estimators':[200,300],
  'learning_rate': [0.05,0.01],
  'max_depth':[None,50,100],
  'min_samples_leaf':[3,5,8],
  'max_features':['auto','log2'],
  'loss':['ls','lad','huber','quantile']
}
# Parametros SupportVectorRegressor
param_grid_full = {
  'kernel':['linear','rbf','sigmoid'],
  'gamma': ['scale','auto',1e-7, 1e-4],
  'tol':[1e-3,1e-5,1e-4,1e-2],
  'epsilon':[1e-4],
  'max_iter':[-1],
  'C':[1.5, 10]
}
```

6.3.2.6 Avaliação do modelo

Após os dados serem preditos, é possível avaliar o modelo computacional. Existem algumas métricas que podem ser adotadas para verificar a viabilidade do modelo, podemos verificar o erro médio quadrado e o coeficiente de determinação (*R2 Score*) em relação aos dados corretos e os dados preditos pelo modelo.

Os trechos de códigos a seguir exibe como podem ser obtidos os erros e os coeficientes de determinação para os conjuntos de treino, validação e teste.

```
  from sklearn.metrics import mean_squared_error, r2_score
  X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=best_seed
      )
  X_train,X_val,y_train,y_val = train_test_split(X_train,y_train,test_size=0.2)

  model.fit(X_train,y_train)

  #Erro de treino
  y_train_pred = model.predict(X_train)
  r2_train = r2_score(y_train,y_train_pred)
  mse_train =  mean_squared_error(y_train,y_train_pred)

  #Erro de validacao
  y_val_pred = model.predict(X_val)
```

```
14  r2_validation = r2_score(y_val,y_val_pred)
15  mse_validation =  mean_squared_error(y_val,y_val_pred)
16
17  #Erro de teste
18  y_test_pred = model.predict(X_test)
19  r2_test = r2_score(y_test,y_test_pred)
20  mse_test =  mean_squared_error(y_test,y_test_pred)
```

O resultado de treino indica quando o modelo conseguiu aprender com os dados passados, quando um coeficiente de determinação tende a ser muito próximo 1 e um erro médio quadrado a 0, é uma indicação que o modelo conseguiu aprender todas as características dos dados, até os possíveis ruídos que os mesmos apresentam, em algumas situações isto é uma indicativa de *overfit*.

O resultado de validação é uma indicação se o modelo precisa ser alterado, este resultado é bastante utilizado nas etapa do *tuning*, ele realiza uma verificação se o modelo conseguiu extrair as características geral do problema. Os métodos de validação cruzada apresentados anteriormente, utilizam uma parte do conjunto de dados para realizar a validação.

O resultado dos dados de teste indicam se o modelo final obtido consegue realizar a predição de dados até então desconhecidos, de forma aceitável com um bom coeficiente de determinação, acima de 0.5, e um erro médio quadrado tendendo a 0.

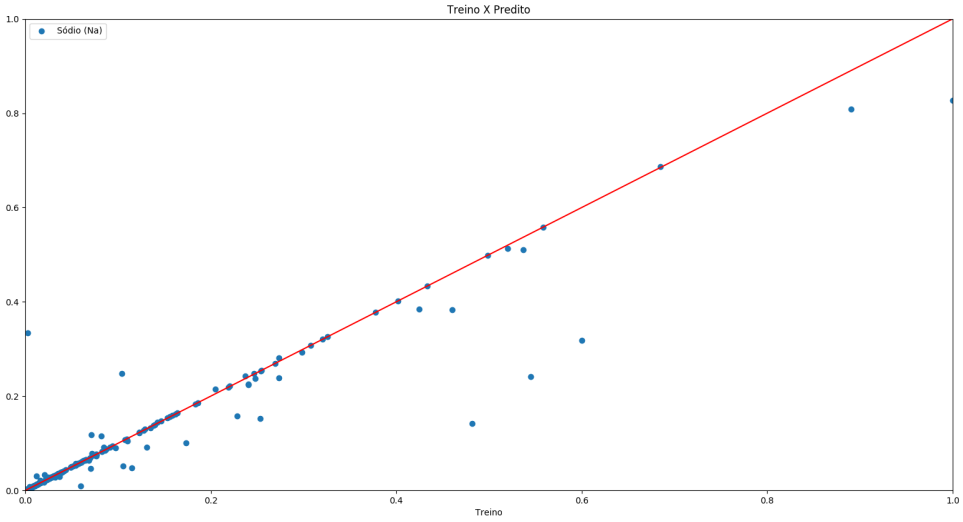
Também como um método de avaliação do modelo podemos plotar um gráfico dos valores corretos versus os preditos, para isso podemos utilizar a ferramenta ***matplotlib***. Com esta é possível traçar gráficos 2D e também 3D, os códigos a seguir exibem como pode ser feito estes procedimentos. Note que para o gráfico citado anteriormente, uma reta similar a função $f(x) = x$ é o melhor gráfico possível.

O código a seguir informa ao leitor o procedimento para plotar um gráficos 2D.

```
1  import matplotlib.pyplot as plt
2
3  plt.figure()
4  plt.title("Treino X Predito")
5  plt.ylabel("Predito")
6  plt.xlabel("Treino")
7  plt.scatter(y_train,y_train_pred)
```

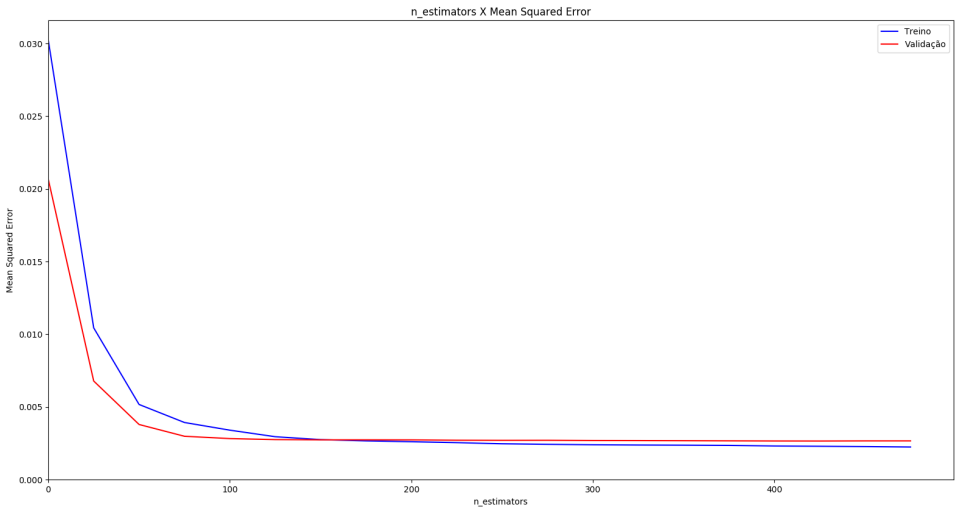
```
8   plt.plot([0,1],[0,1],'r')
9   plt.legend(loc=2)
10  plt.show()
```

Figura 11 – Gráfico 2D feito com a ferramenta ***matplotlib***



Outro tipo de gráfico que pode ser feito para verificara o comportamento do modelo durante o treino do mesmo, é a curva de aprendizagem, esta exibe o quanto o modelo conseguiu diminuir o seu erro ou aumentar o seu acerto, isto a depender da métrica utilizada, a Figura 12 exibe uma curva de aprendizagem para exibindo os erros para os dados de treino e de validação.

Figura 12 – Exemplo de curva de aprendizado. ***matplotlib***



No gráfico apresentado é possível verificar os erros de validação e de treino em função do número de estimadores presentes no modelo, como estes é possível verificar que o sistema convergiu e apresentou um erro constante. Quanto mais próximo for o valor do erro para cada conjunto e ambos tendendo a zero é uma indicação que o modelo conseguiu convergir de forma correta, porém é recomendado sempre observar os resultados apresentados no eixo.

6.3.2.7 Exportar modelo

Após serem encontrados os melhores modelos de predição para cada nutriente devemos salva-lós para utilizarmos novamente sem a necessidade de realizar o treino, isto é possível através do modulo ***joblib*** utilizando os métodos ***dump*** para persistir os dados no disco e ***load*** para carregar o modelo a partir de um arquivo. Com este é possível salvar estruturas de dados do Python em um arquivo. O código abaixo exibe como pode ser feito este tipo de procedimento.

```
1  from joblib import dump, load
2
3  model = RandomForestRegressor()
4  model.fit(x_train,y_train)
5
6  # Salvando o modelo treinado no arquivo model.joblib
7  dump(model, 'model.joblib')
8
9  # Carregando o modelo treinado a partir do arquivo model.joblib
10 model = load('model.joblib')
```

6.4 Integração

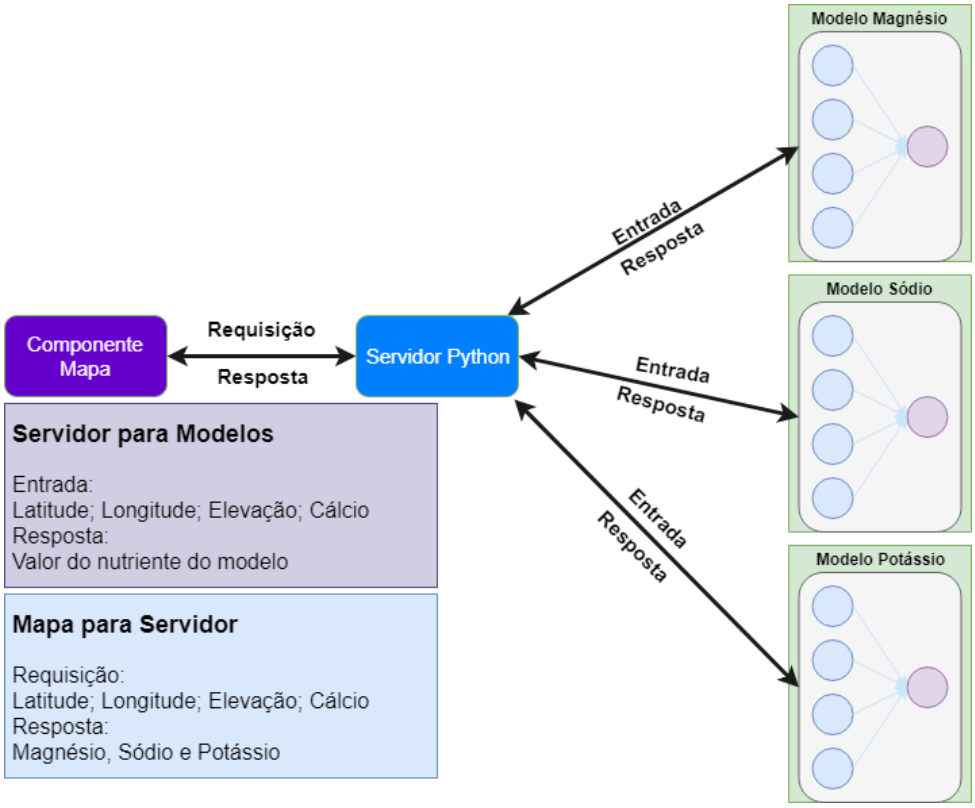
Com o sistema concluído é possível identificar que os módulos existentes são:

- A página web com o componente de mapa, responsável por receber os dados de entrada passado pelo usuário e enviar requisições com estes para o servidor;
- O servidor, responsável por receber os dados de entrada da página web, passar estes para os modelos de predição de dados e enviar os dados preditos para a página;

- Os modelos de aprendizado de máquina, estes são responsável por realizar a predição dos dados e enviar os dados preditos para o servidor.

O diagrama a seguir exibe a integração dos componentes do sistema.

Figura 13 – Diagrama de integração do sistema



7 Resultado e Discussões

Este trabalho realizou testes para verificar quais os melhores modelos de aprendizado de máquina que seriam utilizados e comparados, com estes foram escolhidos os algoritmos *RandomForestRegressor*(RFR), *GradientBoostingRegressor*(GBR) e o *Suport-VectorRegressor*(SVR), todos da biblioteca *sklearn* e utilizados para regressão, porém este capítulo se limitará à apresentar ao leitor apenas os melhores modelos obtidos, para cada nutriente.

Como já foi informado, este trabalho necessitou criar três modelos computacionais para os nutrientes magnésio, sódio e potássio com base nas informações de entrada, latitude, longitude, elevação e cálcio. Para o modelo do magnésio ambos os algoritmos apresentaram bons resultados, com o *Gradient Boosting* levando uma leve vantagem. Para o modelo do sódio o algoritmo *Random Forest* apresentou um bom resultado perante os demais. Para o modelo do potássio o algoritmo *Gradient Boosting* apresentou um bom resultado.

As tabelas a seguir exibem os resultados dos algoritmos para os modelos de predições de nutrientes, as métricas utilizadas para a avaliação dos modelos foram o coeficiente de determinação R^2, o erro médio quadrado (EMQ) e o resultado da validação cruzada (CV) que também utiliza a métrica R^2, porém em diversas combinações de dados de treino e validação, esta é a média dos resultados de cada combinação.

Tabela 8 – Resultados dos modelos do **magnésio**.

Algoritmo	Conjunto de dados						
	Treino			Validação		Teste	
	CV	R^2	EMQ	R^2	EMQ	R^2	EMQ
GBR	0.78	0,91	2,25x10^{-3}	0,86	2,67x10^{-3}	0,79	3,41x10^{-3}
RFR	0.74	0,89	2,74x10^{-3}	0,81	3,51x10^{-3}	0,74	4,21x10^{-3}
SVR	0.78	0,84	4,22x10^{-3}	0,82	3,27x10^{-3}	0,81	2,95x10^{-3}

Tabela 9 – Resultados dos modelos do **sódio**.

Algoritmo	Conjunto de dados						
	Treino			Validação		Teste	
	CV	R^2	EMQ	R^2	EMQ	R^2	EMQ
RFR	0.63	0,94	1,50x10^{-3}	0,63	7,40x10^{-3}	0,80	4,63x10^{-3}
GBR	0.63	0,89	2,37x10^{-3}	0,59	8,31x10^{-3}	0,69	9,82x10^{-3}
SVR	0.65	0,74	5,63x10^{-3}	0,87	2,82x10^{-3}	0,75	8,48x10^{-3}

Tabela 10 – Resultados dos modelos do **potássio**.

Algoritmo	Conjunto de dados						
	Treino			Validação		Teste	
	CV	R^2	EMQ	R^2	EMQ	R^2	EMQ
GBR	0.72	0,99	0,30x10^{-3}	0,74	9,35x10^{-3}	0,72	6,86x10^{-3}
RFR	0.63	0,94	1,3110^{-3}	0,59	8,72x10^{-3}	0,89	2,99x10^{-3}
SVR	0.74	0,88	2,41x10^{-3}	0,67	11,70x10^{-3}	0,75	6,17x10^{-3}

A seguir serão apresentadas a curva de aprendizado com o coeficiente de determinação e com o erro médio quadrado dos melhores modelos, para os dados de treino e validação, com esta é possível verificar que os algoritmos apresentaram um certo determinismo nos seus resultados.

Figura 14 – Curva com o EMQ para o **magnésio**

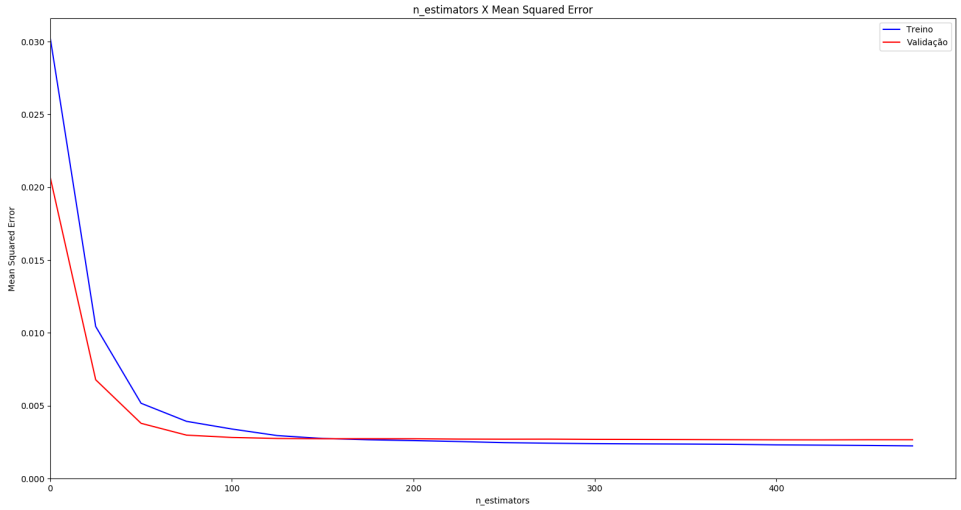


Figura 15 – Curva com o R^2 para o **magnésio**

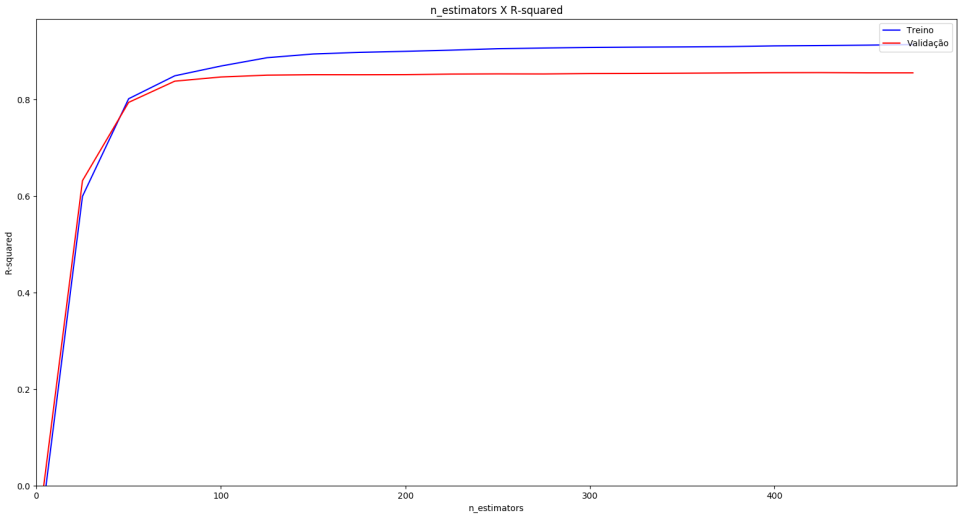


Figura 16 – Curva com o EMQ para o **sódio**

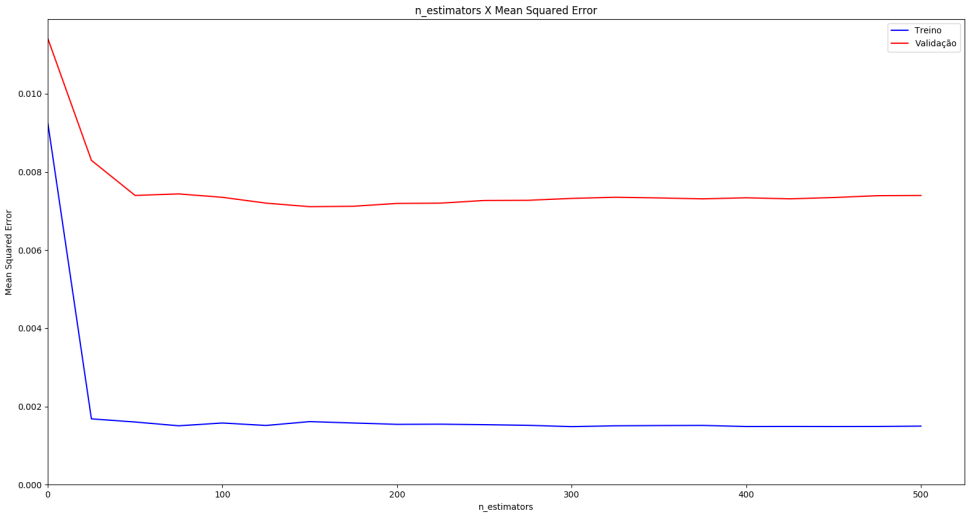


Figura 17 – Curva com o R^2 para o **sódio**

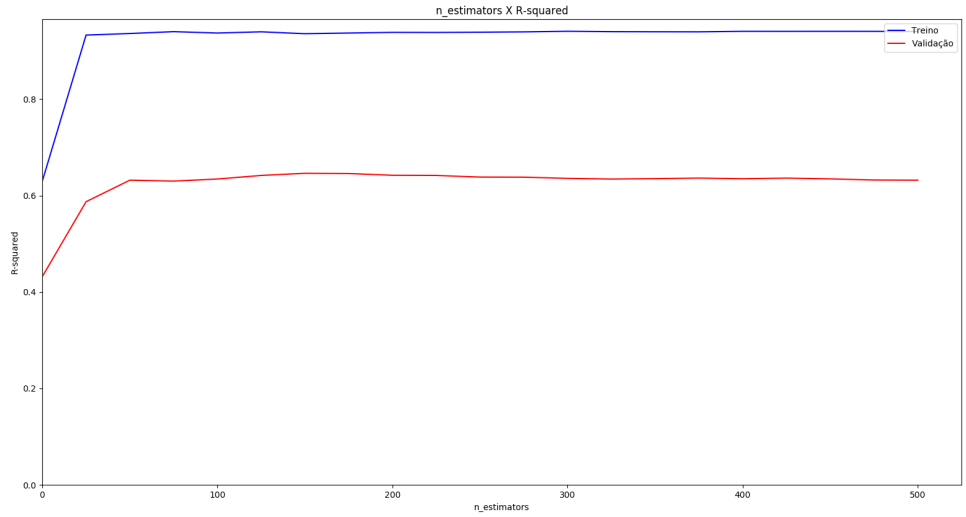


Figura 18 – Curva com o EMQ para o **potássio**

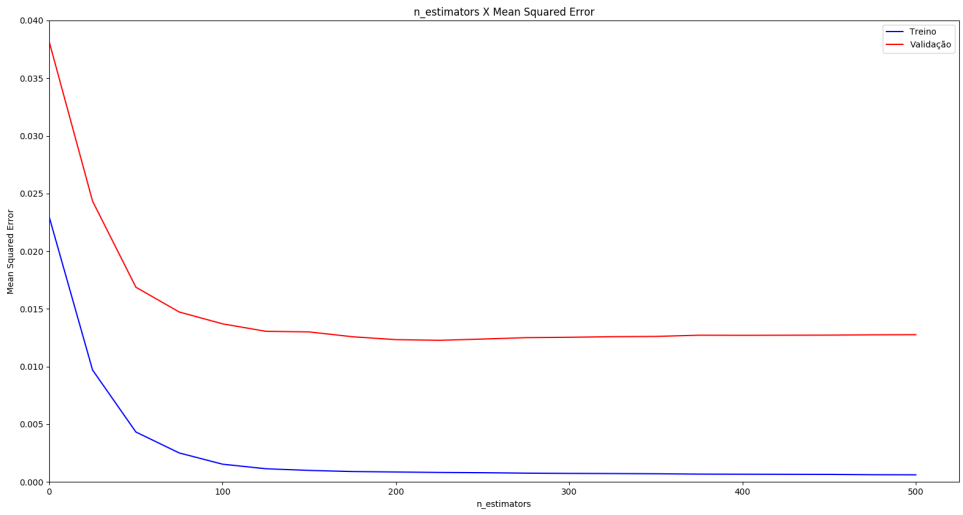
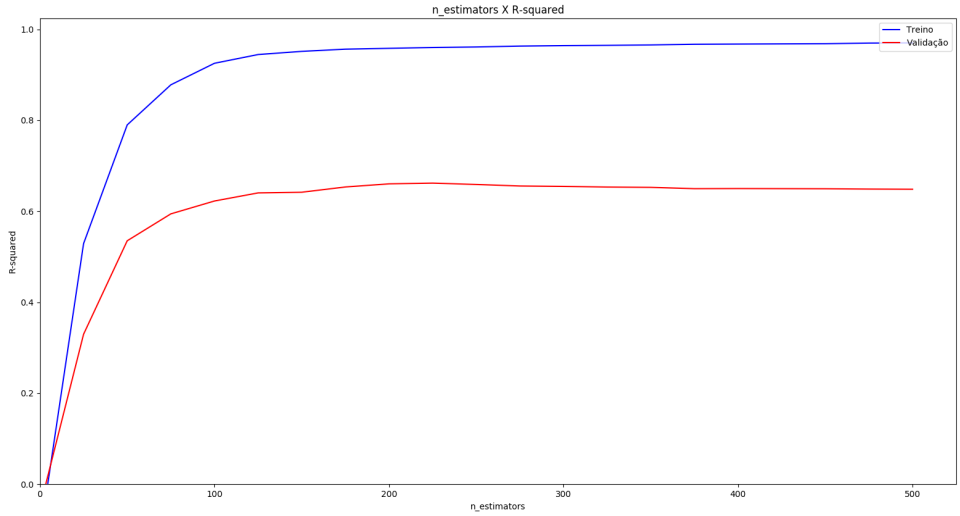
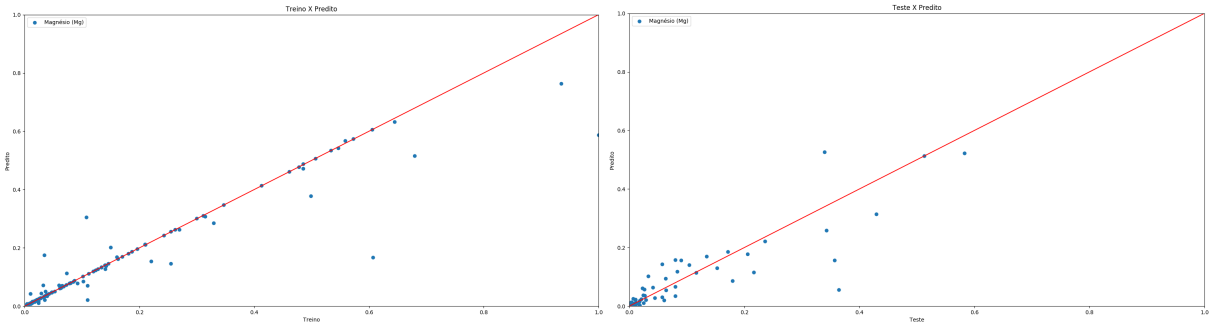


Figura 19 – Curva com o R^2 para o **potássio**



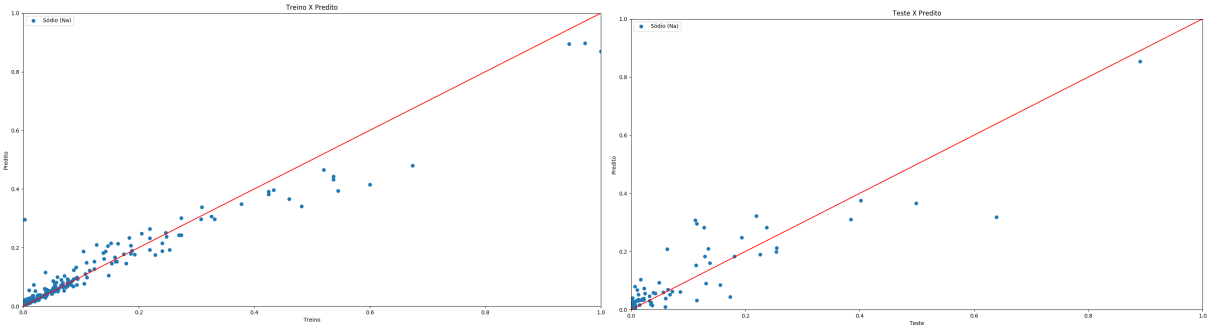
A seguir serão apresentados gráficos dos dados de treino e teste e as suas predições, neste o ideal é identificar um comportamento linear em ambos, pois será plotado o valor real pelo valor predito, quanto mais próximo da reta vermelha melhor.

Figura 20 – Dados reais e preditos do **magnésio**



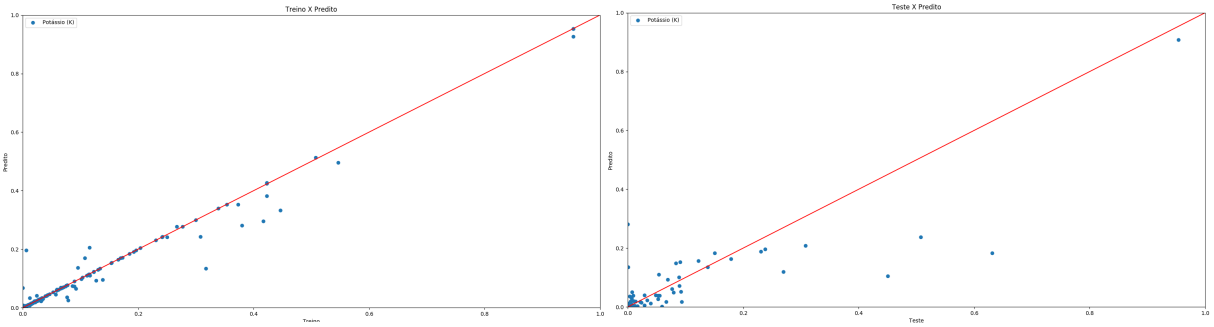
(a) Dados de treino (b) Dados de teste

Figura 21 – Dados reais e preditos do **sódio**



(a) Dados de treino (b) Dados de teste

Figura 22 – Dados reais e preditos do **potássio**



(a) Dados de treino (b) Dados de teste

A seguir serão exibidos os gráficos que indicam o índice do grau de importância das características de entrada dos modelos obtidos. Este apresenta um índice relativo ao maior valor, ou seja, a característica mais importante receberá o valor de 100 e os outros seu grau importância em relação a este. Nos modelos foi possível comprovar o que foi feito com a terceira abordagem informada anteriormente, o valor do cálcio do solo possui um grau de importância elevado para a problemática proposta.

Figura 23 – Grau de importância das variáveis de entrada **magnésio**.

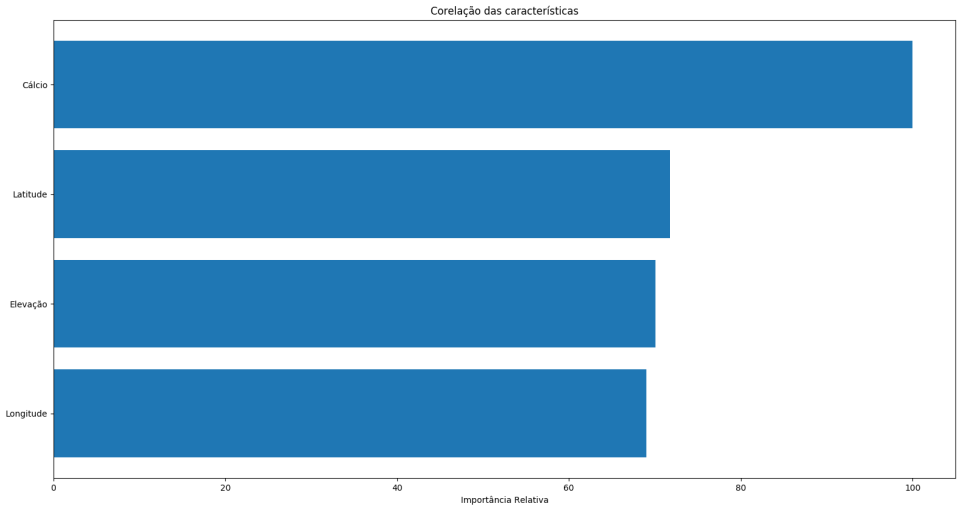


Figura 24 – Grau de importância das variáveis de entrada para o **sódio**.

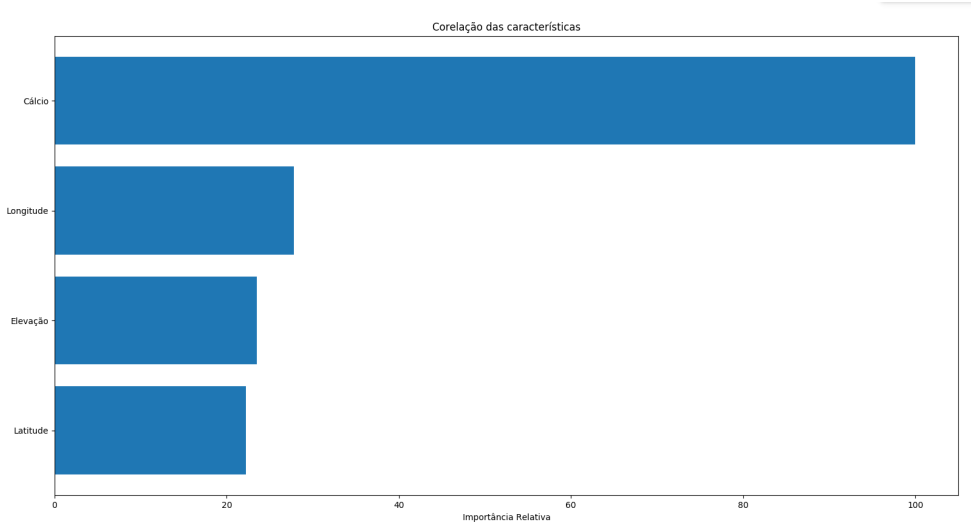
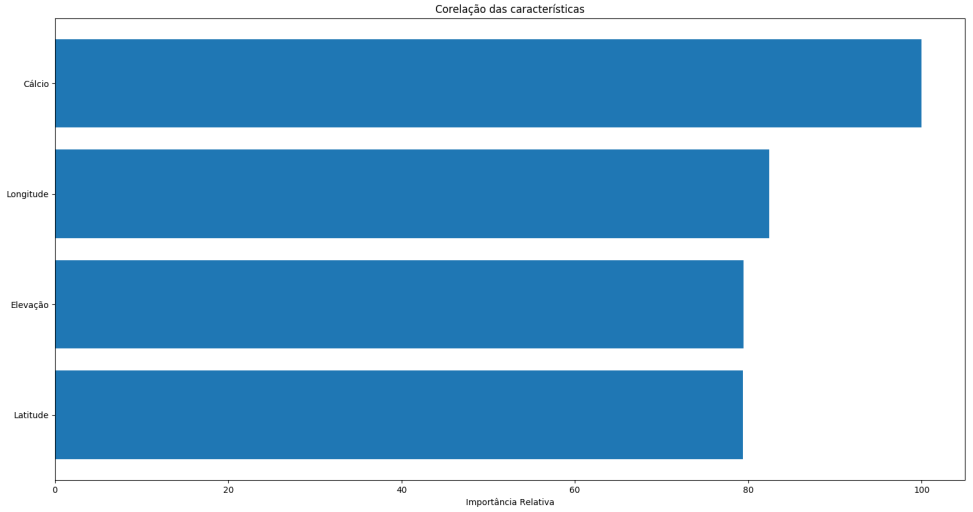


Figura 25 – Grau de importância das variáveis de entrada para o **potássio**.



7.1 Análise de Overfitting e Underfitting

Ao verificar e comparar os resultados obtidos do erro médio quadrado e do coeficiente de determinação para os conjunto de treino, teste e validação, juntamente com os gráficos é possível constatar que os modelos não apresentaram o *overfitting* e o *underfitting*.

O modelo do magnésio apresentou bons resultado em relação aos demais, possuindo um R^2 elevado e um EMQ tendendo a zero para os dados de teste, pelo gráfico também é possível verificar um bom comportamento no treinamento e validação.

Para modelo do sódio e potássio é possível verificar que apresentaram bons resultados para o R^2 e EMQ. Apesar dos gráficos destes modelos não apresentarem um resultado

similar do modelo do magnésio, é possível verificar que a diferença do EMQ de teste e validação, não é demasiada para invalidar o modelo o caracterizando com *overfitting*. No modelo do sódio é possível verificar que o erro inicial é menor do que o modelo do magnésio, ocasionando esse distanciamento na hora da exibição.

Outros modelos utilizados que apresentaram este problema ou foram alterados ou descartados. Apesar do modelo do potássio apresentar o coeficiente de determinação de 0,99 no treino, sendo esta uma indicativa que o mesmo aprendeu todas as características dos dados de treino incluindo o ruído, o mesmo apresentou resultados satisfatórios tanto no seu processo de validação quando no de treino.

Também, como teste para validação do sistema, foi retirada uma amostra de dados presente no conjunto de dados e passada como entrada para o sistema AdubaTec, este foi responsável por gerar um relatório indicando a recomendação de adubação e calagem necessária, logo após foram passados os parâmetros de latitude e longitude da amostra retirada na pagina web, sendo obtido a elevação e o valor do cálcio por estes em seguida todos dados foram passados para os modelos obtidos com finalidade de obter a predição dos nutriente, após este processo os dados preditos foram passados como entradas para o sistema AdubaTec que gerou o relatório de recomendação, ambos relatórios recomendaram os mesmos valores de recomendação de adubação e também de calagem. Este relatórios se encontram no Anexo II.

Todos os código desenvolvidos neste trabalho podem ser acessados no link [1], este também apresenta um notebook que contem os modelos obtidos para cada algoritmo e um mecanismo para replicar os resultados apresentados neste capítulo.

[1] https://github.com/lucashca/TCC

8 Considerações Finais

Com as técnicas existentes de aprendizado de máquina foi possível identificar os índices de nutrientes do solo com base em informações existentes sobre o mesmo. As características preditas foram os valores do magnésio, sódio e potássio as informações escolhidas como entrada foram a latitude, longitude, elevação e o valor do cálcio na localidade, apesar de ainda assim existir a dependência do cálcio, foi possível criar um mecanismo que conseguisse contornar este problema.

Os modelos de predição obtidos neste trabalho conseguiram apresentar um resultado satisfatório, possuindo um erro médio aceitável e principalmente um coeficiente de determinação maior que 0,7, em relação as dados de teste reais e os preditos, sendo que resultados acima de 0,5 são considerados aceitáveis, também apresentaram bons índices no treino com a validação cruzada utilizando um k igual a 10 neste processo.

Apesar dos resultados, este projeto possui uma limitação no seu uso devido ao mecanismo para obtenção do valor do cálcio e da região de funcionamento. Este sistema só é indicado a ser utilizado nas proximidades onde existam amostras marcadas no mapa, fora desta área os modelos de predição retornarão dados aleatórios e inconclusivos.

8.1 Trabalhos Futuros

Poderão ser aplicadas técnicas de aprendizagem profunda (*Deep Learning*) com o intuito de obter resultados melhores do que os apresentados neste trabalho.

Também, após o sistema AdubaTec apresentar uma quantidade de dados de consultas de usuário suficientes, estas técnicas poderão ser aplicadas nestes com a finalidade de obter modelos de predição dos índices de nutrientes com dados atualizados.

Referências

ARAÚJO, L. H. C.; PONTES, L. V.; CRESPO, M. S. *Sistema especialista AdubaTec.* [S.l.]: https://adubatec-h.cnpmf.embrapa.br, 2018. 41

AWS. *Ajuste do modelo: subajuste versus sobreajuste.* 2019. 7, 33

BORGES, A. L. B. *Recomendações de calagem e adubação para abacaxi, acerola, banana, laranja, tangerina, lima ácida, mamão, mandioca, manga e maracujá.* [S.l.]: Embrapa Mandioca e Fruticultura Tropical, 2009. 15, 44

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001. 38

CAMARGO, L. da S. et al. Previsão da produtividade de pastagens a partir de indicadores de solo. 2017. 17

CLAESSEN, M. E. C. Manual de métodos de análise de solo. *Embrapa Solos-Documentos (INFOTECA-E)*, Rio de Janeiro: EMBRAPA-CNPS, 1997., 1997. 15, 40

DIAS, L. M. da S. et al. Predição de classes de solo por mineração de dados em área da bacia sedimentar do são francisco. *Pesquisa Agropecuária Brasileira*, v. 51, n. 9, p. 1396–1404, 2016. 17

FACELI, K. et al. Inteligência artificial: Uma abordagem de aprendizado de máquina. 2011. 23, 24, 27

GARCÍA, S.; LUENGO, J.; HERRERA, F. *Data preprocessing in data mining.* [S.l.]: Springer, 2015. 27

GIARRATANO, J. C.; RILEY, G. *Expert systems.* [S.l.]: PWS publishing co., 1998. 16, 23, 24

HASTIE, T. et al. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, Springer, v. 27, n. 2, p. 83–85, 2005. 13

HELFER, G. et al. Tellus: um modelo computacional para a predição da fertilidade do solo na agricultura de precisão. 2019. 16

HENDERSON, H. *Encyclopedia of computer science and technology.* [S.l.]: Infobase Publishing, 2009. 14, 16

HUNTER, J. D. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, IEEE Computer Society, v. 9, n. 3, p. 90, 2007. 54

KAUFMAN, K. A.; MICHALSKI, R. S. Discovery planning: Multistrategy learning in data mining. 1998. 26, 28

LIMA, M. et al. Projeto radambrasil–folha sd. 24 salvador. *Brasil. Departamento Nacional da Produção Mineral, Rio de Janeiro. Levantamento de Recursos Naturais*, n. 24, 1984. 20, 47, 48

MACHADO, V. P. Inteligência artificial. 2015. 24, 35

MARCUS, S. *Automating knowledge acquisition for expert systems*. [S.l.]: Springer Science & Business Media, 2013. v. 57. 24

MCDERMOTT, J. R1: A rule-based configurer of computer systems. *Artificial intelligence*, Elsevier, v. 19, n. 1, p. 39–88, 1982. 23

MCKINNEY, W. pandas: a foundational python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, v. 14, 2011. 53

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas inteligentes-Fundamentos e aplicações*, v. 1, n. 1, p. 32, 2003. 7, 24, 25, 35

NAGELKERKE, N. J. et al. A note on a general definition of the coefficient of determination. *Biometrika*, Oxford University Press, v. 78, n. 3, p. 691–692, 1991. 32

PADILHA, V. A.; CARVALHALHO, A. C. P. d. L. *Mineração de Dados em Python*. [S.l.]: Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, 2017. v. 1. 27

PEDREGOSA, F. et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, v. 12, n. Oct, p. 2825–2830, 2011. 30, 54, 57

QUINLAN, J. R. Induction of decision trees. *Machine learning*, Springer, v. 1, n. 1, p. 81–106, 1986. 36, 37

RAIJ, B. van. *Recomendações de adubação e calagem para o Estado de São Paulo*. [S.l.]: IAC Campinas, 1996. 45

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. [S.l.]: Malaysia; Pearson Education Limited„ 2016. 13, 22, 23

SAMMUT, C.; WEBB, G. I. *Encyclopedia of machine learning*. [S.l.]: Springer Science & Business Media, 2011. 29, 31

STARNES, D. S.; YATES, D.; MOORE, D. S. *The practice of statistics*. [S.l.]: Macmillan, 2010. 32

TEIXEIRA, J. de F. *Inteligência artificial*. [S.l.]: Pia Sociedade de São Paulo-Editora Paulus, 2014. 23

TJUR, T. Coefficients of determination in logistic regression models—a new proposal: The coefficient of discrimination. *The American Statistician*, Taylor & Francis, v. 63, n. 4, p. 366–372, 2009. 32

WALT, S. V. D.; COLBERT, S. C.; VAROQUAUX, G. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, IEEE Computer Society, v. 13, n. 2, p. 22, 2011. 53

Anexo I

Script para as requisições da elevação.

```
const csv = require('csv-parser');
const fs = require('fs');
const https = require('https');
const createCsvWriter = require('csv-writer').createObjectCsvWriter;
let csvFile = []
const csvWriter = createCsvWriter({
  path: 'DataSetWithElevation.csv',
  header: [
  { id: 'Amostra', title: 'Amostra' },
  { id: 'Latitude', title: 'Latitude' },
  { id: 'Longitude', title: 'Longitude' },
  { id: 'Elevation', title: 'Elevation' },
  { id: 'CA', title: 'Ca' },
  { id: 'MG', title: 'Mg' },
  { id: 'Na', title: 'Na' },]});
fs.createReadStream('mainDataSet1Original.csv')
.pipe(csv()).on('data', (row) => {csvFile.push(row)})
.on('end', () => {getAllLocation(csvFile, 1)});
function parseLocation(data) {
  return data.Latitude + ',' + data.Longitude
}
let rowWithErros = []
async function getAllLocation(csvData, index) {
  if (csvData.length > index) {
    location = parseLocation(csvData[index])
  } else {
    csvWriter.writeRecords(csvFile)
    .then(() => console.log('The CSV file was written successfully'));
    return
  }
  https.get('https://api.jawg.io/elevations?locations=' + location + '&access-token=[YOU-
      TOKEN]'
  , (resp) => {
    let data = '';
    resp.on('data', (chunk) => {data += chunk;});
    resp.on('end', async () => {
      let elevation = JSON.parse(data)[0].elevation
      csvFile[index].Elevation = elevation
      await resolveAfterXMileSeconds(300)
      getAllLocation(csvData, index)
    });
  }).on("error", (err) => {
    getAllLocation(csvData, index)
```

```
43      rowWithErros.push(index)
44    });
45  }
46  function resolveAfterXMileSeconds(x) {
47    return new Promise(resolve => {
48      setTimeout(() => {resolve(x);}, x);
49    });
50  }
51  console.log(rowWithErros)
```

Anexo II

A seguir são apresentados os relatórios fornecidos pelo sistema AdubaTec referente ás entradas de dados corretas e preditas pelo modelo, respectivamente.

Embrapa
Mandioca e Fruticultura

Dados Cadastrados

- Nome: Lucas Henrique Costa Araújo
- E-mail: lucashenriquecaraujo@gmail.com
- Estado: Bahia
- Cidade: Cruz das Almas
- Cultura: Lima ácida Tahiti
- Variedade: Sem especificação
- Estágio: Formação
- Sistema de Cultivo: Sem especificação
- Clima: sem especificação
- Idade: 1° Ano
- Espaçamento: 7 x 4 m

Recomendação de calagem

Necessidade de Calagem (t/ha)

Dispensa

Recomendação de adubação

Nitrogênio

Nitrogênio - N (kg/ha)	Fosfato de amônio (kg/ha)
40	133,33

Observações

- Toda a consulta foi feita levando em consideração os seguintes extratores:

Mehlich-1: (P, K, Na, Cu, Mn, Zn) **Cloreto de Potássio/1M**: (Al, Ca, Mg) **Acetato de Calcio/0,5M**: (H+Al)

Dados da Análise

Nutriente	Resultado da análise	Unidade
Potássio (K)	5,40	(cmol $_c$/dm³)
Cálcio (Ca)	6,55	(cmol $_c$/dm³)
Magnésio (Mg)	2,78	(cmol $_c$/dm³)
Sódio (Na)	2,51	(cmol $_c$/dm³)

Dados Calculados

Variáveis	Resultado	Descrição
CTC	17,24	Capacidade de troca de cátions
CTCe	17,24	Capacidade de troca de cátions efetiva
SBases	17,24	Soma de bases
Ca/Mg	2,36	Relação cálcio por magnésio
K/Mg	1,94	Relação potássio por magnésio
K/Ca	0,82	Relação potássio por cálcio
%K	32	Saturação por potássio
%Ca	38	Saturação por cálcio
%Mg	17	Saturação por magnésio
m%	0	Saturação por aluminío
V%	101	Saturação por bases

Embrapa
Mandioca e Fruticultura

Dados Cadastrados

- Nome: Lucas Henrique Costa Araújo
- E-mail: lucashenriquecaraujo@gmail.com
- Estado: Bahia
- Cidade: Cruz das Almas
- Cultura: Lima ácida Tahiti
- Variedade: Sem especificação
- Estágio: Formação
- Sistema de Cultivo: Sem especificação
- Clima: sem especificação
- Idade: 1° Ano
- Espaçamento: 7 x 4 m

Recomendação de calagem

Necessidade de Calagem (t/ha)

Dispensa

Recomendação de adubação

Nitrogênio

Nitrogênio - N (kg/ha)	Fosfato de amônio (kg/ha)
40	133,33

Observações

- Toda a consulta foi feita levando em consideração os seguintes extratores:

Mehlich-1: (P, K, Na, Cu, Mn, Zn) **Cloreto de Potássio/1M**: (Al, Ca, Mg) **Acetato de Calcio/0,5M**: (H+Al)

Dados da Análise

Nutriente	Resultado da análise	Unidade
Potássio (K)	4,48	(cmol $_c$/dm³)
Cálcio (Ca)	0,89	(cmol $_c$/dm³)
Magnésio (Mg)	1,84	(cmol $_c$/dm³)
Sódio (Na)	1,83	(cmol $_c$/dm³)

Dados Calculados

Variáveis	Resultado	Descrição
CTC	9,04	Capacidade de troca de cátions
CTCe	9,04	Capacidade de troca de cátions efetiva
SBases	9,04	Soma de bases
Ca/Mg	0,48	Relação cálcio por magnésio
K/Mg	2,43	Relação potássio por magnésio
K/Ca	5,03	Relação potássio por cálcio
%K	50	Saturação por potássio
%Ca	10	Saturação por cálcio
%Mg	21	Saturação por magnésio
m%	0	Saturação por aluminío
V%	101	Saturação por bases