UNIVERSIDADE federal fluminense

mba em ciência de dados

André Luis de Souza

predição de chuvas: utilizando machine learning

rio de janeiro

2023

Resumo

A previsão de chuvas tem um importante papel na sociedade, podendo evitar grandes catástrofes e até mesmo perdas humanas, isto acontece tanto nas grandes cidades grandes centros comerciais do nosso País como Rio de Janeiro e São Paulo quanto em pequenas cidades do interior de cada estado. É de suma importância salientar o uso desta prática para o agronegócio que representa cerca de 26% do PIB nacional (CNABrasil), ou seja, um pouco mais de um quarto de toda riqueza gerada no País, em termos do agronegócio esta previsão é fundamental para evitar perdas e dar a oportunidade as grandes empresas do ramo de se organizarem em períodos de fortes tempestades. A previsão de fortes chuvas pode exercer um papel importante na matriz energética brasileira já que as hidroelétricas têm presença em cerca de 62% da energia produzida, segundo site governamental EPE (Empresa Pesquisa Elétrica), prever um fenômeno como este é uma árdua tarefa, visto que por mais que existam aparelhos capazes de medir: temperatura, pressão, densidade de nuvens, volume de água etc. Mesmo com todos estes métodos não foi possível produzir resultados precisos de previsão segundo Roopa (2019), usando técnicas de aprendizado de máquina, podemos produzir resultados mais aproximados. Nós podemos apenas fazer isso tendo como base a análise de dados históricos de precipitação e assim poder “treinar” a máquina utilizando padrões obtidos e assim obter melhores resultados.

Palavras-chave: Aprendizado de máquina; models preditivos; chuvas.

Abstract

The forecast of rains has an important role in society, being able to avoid major catastrophes and even human losses, this happens both in large cities large commercial centers in our country such as Rio de Janeiro and São Paulo and in small cities in the interior of each state. It is extremely important to emphasize the use of this practice for agrobusiness, which represents around 26% of the national PIB (CNABrasil), that is, a little more than a quarter of all the wealth generated in the country. to avoid losses and give the opportunity to big compannies in the field to organize themselves in periods of strong storms. according to the government website EPE (Empresa Pesquisa Elétrica), predicting a phenomenon like this is an arduous task, since even though there are devices capable of measuring : temperature, pressure, cloud density, water volume, etc. Even with all these methods, it was not possible to produce accurate prediction results according to Roopa (2019), using machine learning techniques, we can produce more approximate results. We can only do this based on the analysis of historical precipitation data and thus be able to “train” the machine using patterns obtained and thus obtain better results..

Keywords: Machine learning; predictive models; rainfall

Capítulo 1

1. Introdução

## Contextualização

A humanidade já havia tentado prever o fenômeno natural da chuva desde quando estavam reunidas em tribos observando pássaros no céu, como o caso das andorinhas que voam em grupos e alturas maiores do que normalmente quando uma grande tempestade está para acontecer Vieira, W. G., & Silva, J. B. (2005). A importância da previsão do tempo na agricultura, a uma distância de três a quatro quilômetros, uma corrente de ar quente e úmido empurra os insetos para cima, levando as andorinhas para camadas superiores.

Claramente este é o modo mais arcaico de prever chuvas, após o conhecimento obtido pela física sabemos que quando está para iniciar chuvas é desencadeado uma série de acontecimentos: a pressão atmosférica cai, a temperatura do ar diminui, a umidade do ar aumenta, nuvens se tornam mais volumosas e por meio de ondas mecânicas é possível descobrirmos se está aumento de volume foi causado pela formação de água (ou até mesmo gelo) em seu interior (Cielo, 12(8), 634-641).

Porém mesmo diante de tantas opções dispostas e do aumento tecnológico ainda não é possível prever com muita precisão quando uma chuva irá acontecer e ainda onde, por isso o presente trabalho visa demonstrar e diante do exposto responder a seguinte questão problema: É possível utilizarmos aprendizagem de máquina para prever chuvas baseados em dados históricos?

Para tanto, foi utilizada a metodologia segundo Gil (2002), baseada em estudo de casos documental com uso de dados primários (obtidos por sites oficiais governamentais).

A abrangência utilizada no presente trabalho está baseada no: Estudo de casos do uso de predição de chuvas utilizando a técnica chamada de: “Machine Learning”.

Tema geral: Predição de chuvas utilizando Machine Learning

Delimitação: Estudo das principais técnicas que tornam possível realizar uma previsão do clima utilizando aprendizagem de máquina.

## 1.2 Descrição do problema

O principal problema a ser abordado neste trabalho é a dificuldade de prever com precisão as chuvas em uma determinada região. Isso se deve ao fato de que as chuvas são influenciadas por muitos fatores, como temperatura, pressão atmosférica, umidade, vento, entre outros. Além disso, a ocorrência de eventos extremos, como secas prolongadas ou chuvas intensas, torna ainda mais desafiadora a tarefa de prever as chuvas com antecedência (YASEEN et al., 2019).

Capítulo 2

2. Fundamentação Teórica

Basicamente A aprendizagem de Máquina pode ser dividida em dois grupos:

Aprendizado Generativo:

O aprendizado generativo se concentra em modelar a distribuição conjunta dos dados de entrada e de saída. Em outras palavras, o objetivo é aprender o padrão subjacente dos dados para gerar novos exemplos que se assemelhem aos dados de treinamento. Esse tipo de abordagem é usado em problemas de modelagem de probabilidade, como reconhecimento de imagens e síntese de dados. Os modelos generativos podem ser usados para gerar novos dados que se assemelhem ao conjunto original, tornando-os úteis em tarefas como geração de imagens, texto e áudio.

Aprendizado Preditivo: Por outro lado, o aprendizado preditivo está focado na construção de modelos que se concentram em aprender a mapear os dados de entrada para os dados de saída. Em vez de se preocupar com a distribuição conjunta dos dados, os modelos preditivos se concentram em encontrar relações e padrões que permitem prever com precisão os resultados de novos dados não vistos. Esse tipo de abordagem é amplamente utilizado em problemas de classificação, regressão e outras tarefas de previsão. (Bishop, C. M. 2006).

A exemplo da diferença das duas abordagens acima citadas, temos o Chat GPT que faz uso do tipo de aprendizado generativo utilizando redes neurais, fazendo uso de uma ampla massa de dados e sempre imprimindo uma resposta dada uma pergunta podendo estar certa ou errada.

Já o presente trabalho fará uso da técnica Preditiva, utilizando classificação e regressão para executar a tarefa de treinar e prever.

Atualmente, os algoritmos de aprendizado de máquina são utilizados em diversas aplicações que modelam problemas como: classificação, regressão ou agrupamento, a fim de reconhecer padrões ocultos ou difíceis para a percepção humana (Christopher M, 2006).

No presente trabalho, irá fazer uso de algoritmos de classificação para realizar previsões a partir de uma base de dados fornecida pelo Instituto Nacional de Meteorologia. Não apenas usar algoritmo de classificação, mas, provar qual o método melhor se adequa ao caso proposto. Após tratamento da base de dados será necessário alcançar um modelo, pois um modelo é uma especificação de uma relação matemática entre diferentes variáveis (Grus Joel, 2020). E tratar a relação destas variáveis para que melhor se adaptem ao algoritmo escolhido.

A relação que as variáveis apresentam entre si, será fundamental na escolha do que é chamado de “Features” (Alice Zeng et al., 2018). Por fim processar a massa de dados e analisar seus resultados e performance do modelo por meio de dados estatísticos checando o quanto a máquina foi capaz de aprender com os dados históricos fornecidos.

## 2.1 Algoritmos Supervisionados e Não supervisionados

Existem de forma básica dois tipos de algoritmos de aprendizados de máquina: supervisionados e não supervisionados, a principal diferença entre algoritmos supervisionados e não supervisionados está na presença ou ausência de dados rotulados durante o treinamento do modelo. Os algoritmos supervisionados utilizam um conjunto de dados rotulados para treinar o modelo, ou seja, o conjunto de dados contém exemplos de entrada e as respectivas saídas corretas esperadas. O objetivo do modelo é aprender a relação entre os dados de entrada e as saídas esperadas, de forma que possa generalizar essa relação para novos dados de entrada e fazer previsões precisas. Exemplos de algoritmos supervisionados incluem regressão linear, regressão logística, árvores de decisão, redes neurais, entre outros (Sonia Sehra et al 2021).

Já os algoritmos não supervisionados são utilizados quando não se tem dados rotulados, ou seja, o conjunto de dados contém apenas as entradas, sem as respectivas saídas esperadas. O objetivo do modelo é encontrar padrões e estruturas ocultas nos dados de entrada, sem a necessidade de se ter conhecimento prévio sobre as saídas esperadas. Exemplos de algoritmos não supervisionados incluem clustering (agrupamento), análise de componentes principais (PCA), redução de dimensionalidade, entre outros (Hastie T. et al., 2009).

## 2.2 Modelos Sobreajustados e Sub-ajustados

Um perigo comum em aprendizado de máquina é o sobreajuste — produzir um modelo de bom desempenho com os dados que você treina, mas que não lide muito bem com novos dados. Isso pode implicar o aprender com base no ruído dos dados. Ou, pode implicar em aprender a identificar entradas específicas em vez de qualquer fator que sejam de fato preditivos da saída desejada.

O outro lado é o sub-ajuste, produzindo um modelo que não desempenha bem nem com os dados usados no treino, apesar de que, quando acontece isso, você decide que seu modelo não é bom o suficiente e continua a procurar por melhores (Grus Joel, 2020):

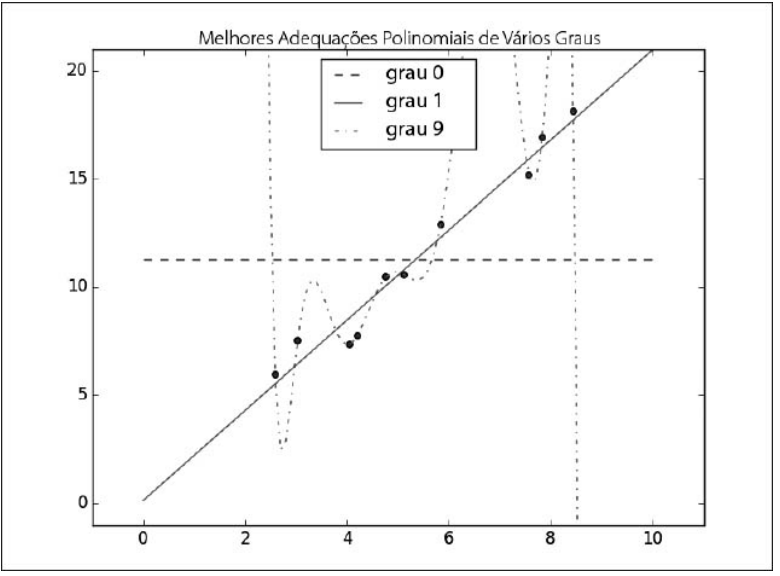


Figura 1 – Sobreajuste e Subajuste

A linha horizontal mostra o melhor grau adequado polinomial 0 (isto é, constante). Ele sub-ajusta o dado em treinamento intensamente. O melhor ajuste por um polinômio do 9º grau (isto é, parâmetro 10) passa por todos os pontos de dados em treinamento, mas sobreajusta gravemente — se fossemos adquirir um pouco mais de pontos de dados, provavelmente sairiam bem errados. E a linha de 1º grau tem um bom equilíbrio — é bem próximo a cada ponto, e (se esses dados são representativos) a linha estará próxima dos novos pontos de dados também. Evidentemente, os modelos que são muito complexos tendem ao sobreajuste e não lidam bem com dados além daqueles com os quais foram treinados. Então, como temos certeza que nossos modelos não são muito complexos? O método mais fundamental envolve o uso de dados diferentes para treinar e testar o modelo. A maneira mais fácil de fazer isso é dividir seu conjunto de dados e teste.

Modelos complexos demais causam sobreajuste e não generalizam quase nada. Então como criar modelos sem complexidade excessiva? A abordagem mais básica consiste em usar diferentes dados para treinar e testar o modelo (Grus Joel, 2020).

Por este motivo os dados são separados no momento de execução do modelo de Machine learning.

## 2.3 Balanceamento de dados

O balanceamento de dados é uma técnica utilizada em análise de dados e Machine Learning para garantir que as amostras usadas no treinamento do modelo estejam distribuídas de maneira uniforme entre as diferentes classes ou categorias.

Quando os dados estão desbalanceados, isso significa que uma classe tem muito mais amostras do que as outras, o que pode levar a problemas na performance do modelo. Por exemplo, se o modelo for treinado com dados desbalanceados, ele pode ter uma tendência a classificar todas as amostras na classe majoritária, ignorando completamente as outras classes.

Para resolver esse problema, é comum aplicar técnicas de balanceamento de dados, que podem ser divididas em duas categorias principais: oversampling (sobreamostragem) e undersampling (subamostragem).

Na sobreamostragem, as amostras da classe minoritária são replicadas ou geradas artificialmente para aumentar o número de amostras nessa classe, tornando-a mais equilibrada em relação às outras classes. Já na subamostragem, algumas amostras da classe majoritária são removidas aleatoriamente para diminuir o número de amostras nessa classe e torná-la mais equilibrada em relação às outras classes.

Existem diversas técnicas específicas para cada uma dessas abordagens, como por exemplo: SMOTE, ADASYN e Random Oversampling/Undersampling. É importante lembrar que o balanceamento de dados deve ser feito de forma cuidadosa e sempre considerando o contexto do problema, pois em alguns casos a desigualdade na distribuição das classes pode ser justificada ou mesmo necessária (Y. Mohana, 2019).

Lidar com um conjunto de Dados desbalanceados pode fazer com que o modelo não performe de maneira correta ou que retorne um resultado fora de realidade, não existindo um padrão que se adeque melhor a base de dados, isto faz com que seja necessário experimentar diferentes métodos e técnicas.

Para evitar esse viés, técnicas de pré-processamento dos dados são utilizadas para alterar as distribuições das classes na base de treinamento, visando reduzir desbalanceamento. Uma abordagem bastante comum é o uso de métodos de amostragem. Tais métodos consistem na eliminação de casos da classe majoritária (undersampling) e replicação (ou geração sintética) de casos da classe minoritária (oversampling) dos dados de treinamento visando obter classificadores melhores do que os obtidos a partir da distribuição original. Segundo Jo e Japkowicz (2004), não há garantia de que a distribuição original dos dados de treinamento seja a mais adequada para a construção de classificadores.

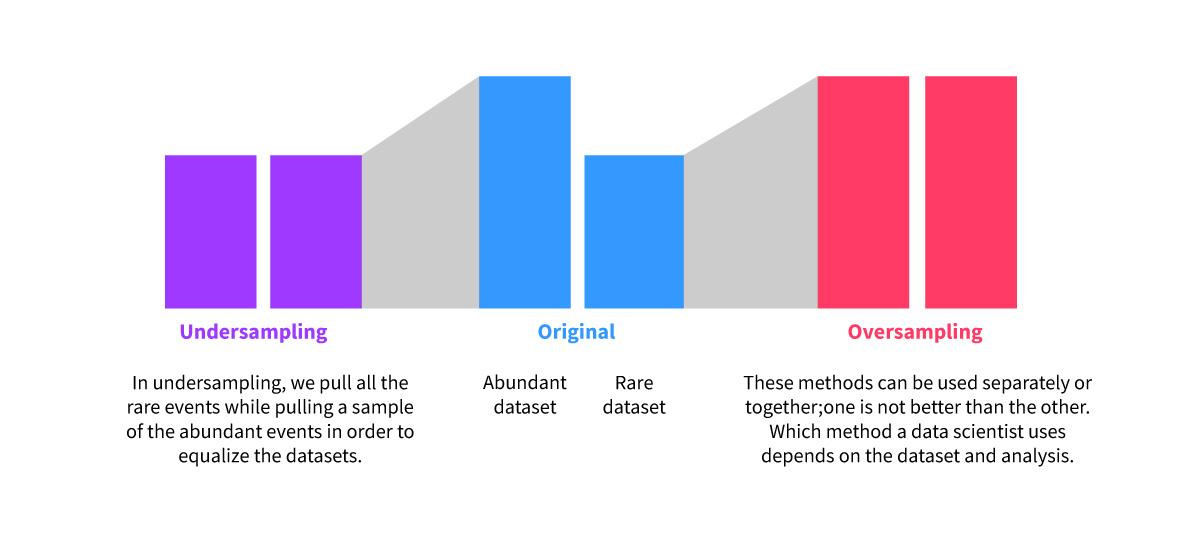


Figura 2 – Diferenças entre Undersampling e Oversampling

Basicamente em ciência de dados existem duas técnicas principais: Undersampling e Oversampling. Undersampling é uma técnica para equilibrar conjuntos de dados desiguais, mantendo todos os dados na classe minoritária e diminuindo o tamanho da classe majoritária. Embora tenha desvantagens, como a perda de informações potencialmente importantes, ainda é uma habilidade comum e importante para os cientistas de dados.

Principio del formulário

## 2.4 Normalização

Técnica de manipulação amplamente utilizada em grandes bases, pois resolve e evita problemas em observações, muitas delas encontrados na natureza que causam uma discrepância dos dados coletados.

A normalização é usada para remover os efeitos de variações em escala, que podem ser causados por fatores como a unidade de medida utilizada para coletar os dados, o tamanho do conjunto de dados ou a variação natural nos valores observados. Isso ajuda a garantir que os dados estejam em uma escala comparável, tornando mais fácil identificar tendências, padrões e anomalias.

Existem várias técnicas de normalização disponíveis, sendo a mais comum a normalização min-max. Nessa técnica, os valores de um conjunto de dados são transformados em uma escala entre 0 e 1, com base no valor mínimo e máximo do conjunto de dados. Outras técnicas incluem a normalização por média e desvio padrão, a normalização por escala logarítmica e a normalização por rescaling (Géron, A. 2019).

Final del formulário

## 2.5 Acurácia

A acurácia é uma métrica amplamente utilizada para avaliar o desempenho de modelos de classificação. Ela representa a proporção de previsões corretas em relação ao total de previsões feitas pelo modelo. A fórmula matemática da acurácia é simplesmente obtida através da divisão do número de previsões corretas pelo número total de previsões realizadas pelo modelo. Essa métrica é expressa geralmente como uma porcentagem multiplicando o resultado por 100. Quanto mais próxima de 100%, maior é a precisão do modelo em fazer previsões corretas. Vale ressaltar que, embora a acurácia seja uma métrica importante, é essencial considerar outras métricas em cenários com classes desbalanceadas ou quando os custos dos erros de classificação variam (Christopher M, 2006).

Acurácia = Número de Previsões corretas

Total de Previsões

Em geral, é expressa como uma porcentagem multiplicando o resultado por 100:

Acurácia (%) = Número de Previsões corretas x 100

Total de Previsões

## 2.6 Correlação

O grau de dispersão conjunta e de assiciação linear entre duas variáveis aleatórias podem ser avaliados pela covariância e coeficiente de correlação (Simon da Fonseca 2006).

Defini-se covariância entre x e y como:

Desenvolvendo-se os parênteses, obtém-se a fórmula prática:

3. Proposta

A proposta realizada no presente trabalho é a de criar um modelo preditivo em Python coletando dados reais do instituto de meteorologia do estado do Rio de Janeiro, da região de Copacabana, contendo todos os dias de chuva do ano de 2022, imputar os dados no modelo e com isto realizar análises, extraindo os seus resultados e observando sua performance e viabilidade.

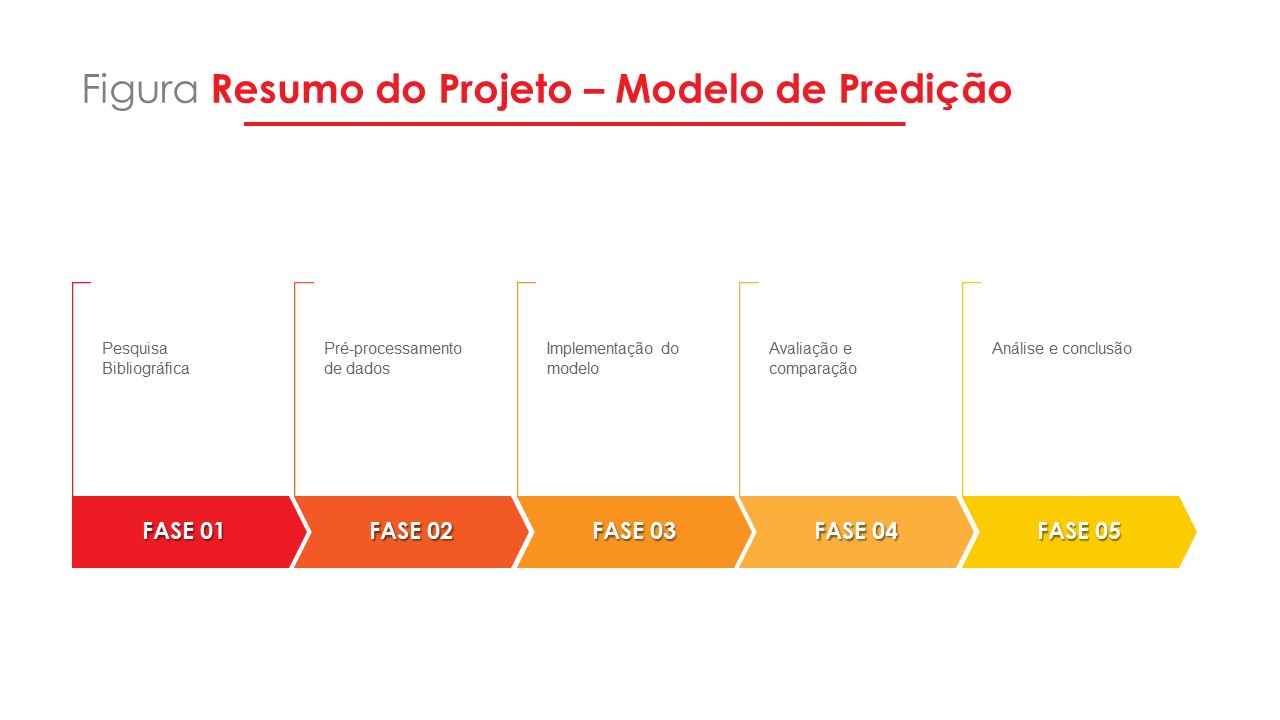


Figura 3 – Fases do Projeto

As fases são:

* Pesquisa Bibliográfica: Coleta de dados: Obtenção de dados históricos de fontes confiáveis e relevantes para a região de estudo.
* Pré-processamento de dados: Limpeza, transformação e tratamento de dados para prepará-los para o treinamento do modelo.
* Implementação do modelo: Desenvolvimento e codificação do modelo de previsão de chuvas (Python).
* Avaliação e comparação: Medição do desempenho do modelo usando métricas apropriadas.
* Análise e conclusão: Discussão dos resultados obtidos, identificação de pontos fortes e limitações do modelo proposto.

4. Implementação

## 4.1 O conjunto de dados

Ao iniciarmos a Fase 1, pesquisa bibliográfica, é realizado ampla pesquisa que busca conceitos para resolução da questão problema além de procurar um conjunto de dados apropriado, e o conjunto que fora obtido foi descarregado diretamente do site do Instituto Nacional de Meteorologia, obtendo informações meteorológicas da região do estado do Rio de janeiro. fonte utilizada: <https://portal.inmet.gov.br/> estas informações podem ser obtidas indo em dados meteorológicos e depois em histórico de dados meteorológicos. Um arquivo em zip foi baixado contendo informações de todas as estações meteorológicas no ano de 2022 de todos os estados brasileiros em csv.

O estado do Rio de Janeiro contém 31 estações meteorológicas sendo 11 convencionais e 20 automáticas disponível em <https://oglobo.globo.com/rio/em-meio-onda-de-calor-no-verao-medicao-de-temperatura-no-rio-encontra-percalcos-entenda-24866630> acessado em 25 de jul. 2023. A diferença entre as estações convencionais e automáticas é que as convencionais se tratam de equipamentos com diversos instrumentos diferentes e sendo necessário para a coleta de dados a ida de profissional ao local, uma vez lá irá de maneira visual coletar os dados, já as ditas automáticas consiste em modernos aparelhos que enviam dados de forma “on-line” para o sistema do Instituto, segundo material divulgado da própria página da internet do instituto de Meteorologia, chamado “ A estação meteorológica e o seu Observador”, um informativo que se encontra disponível em: <https://portal.inmet.gov.br/uploads/publicacoesDigitais/aestacaometeorologicaeseuobservador.pdf> acessado em 25 de jul. 2023.

A região de Copacabana sob código da estação A652, latitude -22,98 e longitude -43,19 foi selecionada, se trata de um amplo Dataset contendo 8.769 linhas de observação iniciais, com as seguintes colunas:

* Data
* Hora UTC
* Precipitação Total, Horário (mm)
* Pressão Atmosférica Ao nível da Estação, Horária (mB)
* Pressão Atmosférica Máxima na Hora Ant. (AUT) (mB)
* Pressão Atmosférica Mínima na Hora Ant. (AUT) (mB)
* Radiação Global (kj/m2)
* Temperatura do Ar Bulbo seco, Horária (0C)
* Temperatura do Ponto de Orvalho
* Temperatura Máxima na Hora ANT. (AUT) (0C)
* Temperatura Mínima na Hora ANT. (AUT) (0C)
* Temperatura de Orvalho Máxima na Hora ANT. (AUT) (0C)
* Temperatura de Orvalho Mínima na Hora ANT. (AUT) (0C)
* Umidade Rel. Max. Na Hora (AUT) (%)
* Umidade Rel. Min. Na Hora (AUT) (%)
* Umidade Relativa do Ar, Horaria (%)
* Vento direção Horaria (gr) (0(gr))
* Vento rajada Máxima (m/s)
* Vento velocidade horária (m/s)

Ao total de 19 colunas contendo informações específicas enviadas da estação. Originalmente em um único dia o Observador, coleta dados 24 vezes em 24 horários diferentes, muitas vezes entrando no abrigo onde estará a estação meteorológica.

Varanda de uma casa

Descrição gerada automaticamente

Figura 4 – Estação Meteorológica convencional

Diagrama

Descrição gerada automaticamente

Figura 5 – Estação Meteorológica automática

Outras formas de medições também são amplamente utilizadas para checar as variáveis que podem dar origem a chuvas como: Satélites, Balões meteorológicos, barógrafo, radares etc. A figura 6 mostra em uma visão geral como e quais instrumentos também podem ser usados:

Diagrama

Descrição gerada automaticamente

Figura 6 – Visão geral de como é feita a medição das variáveis meteorológicas

## 4.2 Tratamento do conjunto de dados

Para começar a tratar o dado é necessário sabermos algumas definições, como por exemplo; dado meteorológico, segundo Mário Adelmo varejão Silva “trata-se do valor assumido (qualitativo ou quantitativo) por cada variável meteorológica na OM.” e elemento ou variável meteorológica se refere a “Grandeza que é medida, calculada, estimada ou descrita na OM” sendo, OM observação Meteorológica, e chuva é: “Uma pluviosidade de 1 mm (altura de água) que equivale ao volume de 1 litro (L) de água de chuva que se acumulou sobre uma superfície de 1 m²” (VAREJÃO-SILVA 2006) .

Sendo assim quando chove 1,5 mm não é perceptível a chuva devendo apenas o ar ficar húmido e a calçada estar ligeiramente molhada, é necessário então definirmos qual a quantidade será considerada realmente chuva.

Os meteorologistas e instituições de meteorologia geralmente classificam a quantidade de chuva de acordo com diferentes categorias, como:

* Chuva fraca: até 2,5 mm por hora
* Chuva moderada: de 2,5 mm a 7,6 mm por hora
* Chuva forte: de 7,6 mm a 50 mm por hora
* Chuva torrencial: mais de 50 mm por hora

Essas categorias podem variar dependendo das referências ou critérios adotados em cada país ou região não existindo um critério definido.

Para o presente trabalho foi considerado que, caso em um dia a quantidade for maior ou igual a de 2,5 mm sendo pela soma das observações realizadas ou registro único observado será então chuva.

Para saber se houve mesmo chuva foram criadas duas colunas e trabalhado o dado (a coluna Chuva e Houve\_chuva), somando as observações Precipitação Total Horário, que mede a quantidade de mm de chuva, como são 24, foi realizada a soma destas a verificar se em um mesmo dia houve queda maior ou igual a 2,5 mm, também foi criada uma fórmula para retornar de forma automática se em um determinado dia choveu ou não (feito em Microsoft Excel).

Tabela

Descrição gerada automaticamente com confiança média

Figura 7 – manipulação do conjunto de dados original

Uma vez que é conhecido quando houve chuva em um dia será possível aplicar as bibliotecas em Python por cada observação, por sua vez cada dia. As colunas em que a pressão atmosférica é apresentada foi subtraída a mais alta da mais baixa em um dia para se obter uma amplitude (uma vez que não seria possível trabalhar com todas as 24 observações registradas em função de grande esforço computacional), já as temperaturas foram realizadas médias. Todas as datas permaneceram iguais e as horas não são relevantes para o modelo.

A Coluna houve\_chuva será a chamada variável alvo, o que significa que será através dela que será realizado as principais operações de treinamento, caso a chuva seja fraca, a variável é definida como 0 ao contrário maior do que 2,5 mm no dia a variável será 1, ou seja além de variável alvo ela será também uma variável categórica para facilitar as operações além de exigir menos esforço computacional.

## 4.3 Codificação em Python

### Pré-processamento dos dados

Iniciando a Fase de pré-processando dos dados é necessário trabalhar mais como os dados serão tratados, existem observações faltantes (NULL) que devem ser eliminadas, existem também informações com valores negativos que podem ser resultados de erros de medições por parte dos instrumentos, valores erradamente negativos registrados para velocidade do ar e umidade, além da variável radiação global não entrar na implementação por não apresentar uma relevância para o modelo de predição.

Analisando o conjunto de dados é visto que a coluna data se encontra em um formato que causará erros ao executar comandos, isto é a coluna data se encontra no formato data sendo necessário estar em formato número para trabalhar com comandos de machine learning.

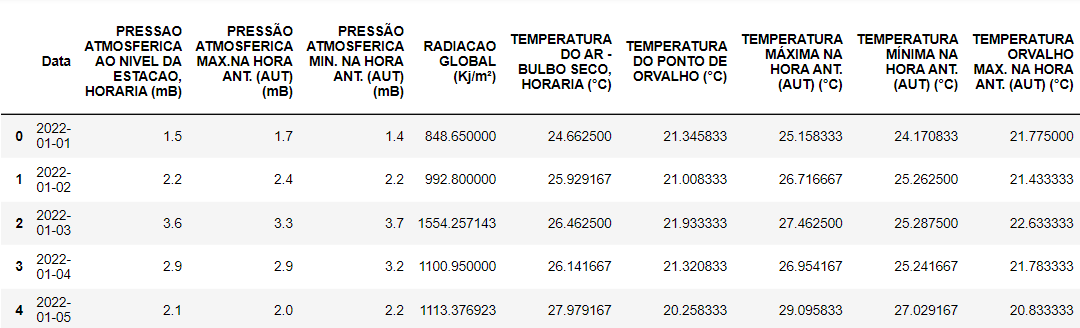


Figura 8 – Coluna Data no formato errado para execução de comandos em Machine Learning

Para resolver esta questão vamos transformar o formato desta variável data em número através do comando:

Texto

Descrição gerada automaticamente com confiança médiaFigura 9 – Comando de conversão de formato data em número

Pela saída do comando Df.info é possível checar se a transformação da coluna data em número resultou no dado esperado checando seu formato:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Figura 10 – Conversão da coluna data resultando em formato Inteiro

Necessário checarmos se existe algum valor faltante no Dataset, pois podem impactar de forma negativa o resultado e até mesmo gerar erros ao trabalhar com as bibliotecas de Machine Learning.

Primeiramente vamos checar quais são as informações faltantes no Dataset realizando uma soma de valores nulos através do comando df.isnull().sum()

Texto

Descrição gerada automaticamente

Figura 11 – Verificado valores faltantes no Dataset

Logo, existem valores faltantes em: Temperatura do Ponto de Orvalho, Temperatura Orvalho MAX, NA Hora ANT. (AUT) (0C), Temperatura Orvalho MIN, NA Hora ANT. (AUT) (0C), UMIDADE REL MAX NA Hora ANT. (AUT) (0C), UMIDADE REL MIN NA Hora ANT. (AUT) (0C) , UMIDADE RELATIVA DO AR, HORARIA (%), VENTO DIREÇÃO HORARIA (gr) (0 (gr)), VENTO RAJADA MAX (m/s)

Como são valores importantes para a análise de dados não poderemos excluí-los ao invés disto iremos usar uma técnica chamada **imputação pela média** que consiste em substituir os valores nulos pelas suas médias.

Texto

Descrição gerada automaticamenteFigura 12 – Utilizando a substituição dos valores faltantes pela imputação pelas médias.

Realizando novamente a checagem da soma, o resultado esperado é 0 em todas as colunas.

Texto

Descrição gerada automaticamente

Figura 13 – Todos os valores nulos eliminados conforme esperado.

Outra forma de checar e ter absoluta certeza de que não existem, mas dados nulos:

Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamente

Figura 14 – Garantindo que não existem dados nulos no Dataset.

Como o resultado foi 0, isto significa que ao vasculhar cada coluna do Dataset não foi encontrado nenhum dado nulo.

Para evitar erros será realizado uma operação de eliminar espaços vazios no começo e no final das palavras com o comando strip.



Figura 14 – Tratamento de dados, valores em String.

Finalizando assim a limpeza e o pré-processamento de dados do dataset.

### Lidando com classes desbalanceadas

Ao analisar o conjunto de dados vemos que a quantidade de dias sem chuva pode estar desbalanceada com relação aos dias com chuva, para verificar esta informação utilizaremos um gráfico do pacote matplotlib que nos mostrará através de gráfico de pizza se esta coluna está realmente desbalanceada.

Gráfico, Gráfico de pizza

Descrição gerada automaticamente

Figura 15 – Confirmação do desbalanceamento na variável alvo

A figura acima nos afirma que a coluna Houve\_chuva se encontra realmente desbalanceada, pois, para o valor 0 temos 77.8% dos valores enquanto para o valor 1, quando houve chuva, se encontra 22,2% dos dados. Certos têm o desbalanceamento característico, principalmente em se tratando de eventos na natureza.

Vamos tentar checar a mesma informação, mas agora com um gráfico que nos dê em número de dias o desbalanceamento:

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura 16 – Utilizando gráfico que demonstrar o desbalanceamento em número de dias

Pela análise do resultado vimos que a classe se encontra desbalanceada pois houve 284 dias em que não chuveu e 81 dias do ano que choveu. Logo uma classe é 3,5 vezes maior do que a outra, podemos considerar que a classe 0 é o triplo da classe 1.

Se desenvolvermos o modelo com os dados como estão, o resultado não fará sentido, conforme foi comprovado pelos próximos comandos:

Instanciando o algoritimo classificador

In [13]:

lr **=** LogisticRegression()

Separando os dados e suas classes do dataframe

In [36]:

x **=** df**.**drop("houve\_chuva", axis**=**1)

y **=** df**.**houve\_chuva

Separando os dados em Treino e Teste:

In [15]:

x\_train, x\_test, y\_train, y\_test **=** train\_test\_split(x, y, random\_state**=**42, stratify**=**y)

Treina o Algoritimo:

In [16]:

lr**.**fit(x\_train,y\_train)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear\_model\\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):

STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:

https://scikit-learn.org/stable/modules/preprocessing.html

Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear\_model.html#logistic-regression

n\_iter\_i = \_check\_optimize\_result(

Out[16]:

LogisticRegression()

Coloca as Classificações na variável y\_pred

In [17]:

y\_pred **=** lr**.**predict(x\_test)

**Checa a Acurácia do Modelo**

Ou seja, checando a porcentagem de acerto do meu modelo (considerando que tenho classes desbalanceadas).

Se eu acertar muito de uma classe e a minha acurácia ficar muito alta, isto ocorre devido ao desbalanceamento (isto é porque realmente as chuvas não ocorrem de maneira simétrica no ano). Logo será necessária uma nova estratégia para conseguir fazer com que o modelo aprenda e tenha um resultado mais crível.

In [18]:

accuracy\_score(y\_test,y\_pred)

Out[18]:

0.9782608695652174

O valor de 97% de acurácia é irreal, agora vamos tentar desenvolver o modelo utilizando outra forma de balanceamento que fará uso de técnicas de undersampling e posteriormente outra técnica de oversampling e comparar qual irá se adequar à realidade.

Aplicando o “Undersampling”:

nr = NearMiss()

nr **=** NearMiss()

In [21]:

print("O valor de X",x**.**shape)

print("O valor de Y",y**.**shape)

*#x = x.transpose()*

*#x = x.values.reshape((365, 162))*

O valor de X (365, 18)

O valor de Y (365,)

In [31]:

**import** numpy **as** np

**from** imblearn.under\_sampling **import** RandomUnderSampler, NearMiss

*# Define o tamanho do vetor y igual ao tamanho do vetor x*

*#y = np.concatenate([np.zeros(181), np.ones(184)], axis=0)*

*# Converte o vetor y em uma matriz bidimensional*

y **=** y**.**reshape(**-**1, 1)

*# Aplica o RandomUnderSampler*

rus **=** RandomUnderSampler(random\_state**=**42)

x\_resampled, y\_resampled **=** rus**.**fit\_resample(x, y)

*#nm = NearMiss(version=1)*

*#x\_nm, y\_nm = nm.fit\_resample(x\_resampled, y\_resampled)*

Agora eu checo visualmente se minha classe esta balanceada:

y\_flat **=** y\_resampled**.**flatten()

ax **=** sns**.**countplot(x**=**y\_flat)

Uma imagem contendo Forma

Descrição gerada automaticamente

Figura 17 – Classe Balanceada ao utilizar Undersamplig

Agora que minha classe "target" se encontra balanceada, isto significa que o algoritimo de Machine Learning tende a aprender e dar um resultado mais realista.

In [33]:

*# Dividir os dados em conjuntos de treinamento e teste*

x\_train, x\_test, y\_train, y\_test **=** train\_test\_split(x\_nm, y\_nm, random\_state**=**42)

In [34]:

lr**=** LogisticRegression()

In [35]:

lr**.**fit(x\_train,y\_train)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear\_model\\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):

STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:

https://scikit-learn.org/stable/modules/preprocessing.html

Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear\_model.html#logistic-regression

n\_iter\_i = \_check\_optimize\_result(

Out[35]:

LogisticRegression()

In [36]:

y\_pred **=** lr**.**predict(x\_test)

**Checando minha Acurácia com o modelo balanceado:**

In [37]:

accuracy\_score(y\_test,y\_pred)

Out[37]:

0.989010989010989

No código acima foi utilizado o "RandomUnderSampler" que é uma técnica de undersampling que seleciona aleatoriamente um subconjunto das instâncias da classe majoritária para tornar o conjunto de dados mais equilibrado. Ele reduz o número de exemplos da classe majoritária para que fiquem em proporção com a classe minoritária.

Logo, fazendo uso de Undersampling vemos que o modelo ainda não está performando corretamente, pois o modelo pode estar sobreajustado ou subajustado, e por isso, resultando altos valores como 98% de acertos, que não retrata a realidade.

Se imprimirmos um relatório de métricas vemos que o modelo está performando pior do que esperávamos:

print(classification\_report(y\_test,y\_pred))

precision recall f1-score support

0.0 0.98 1.00 0.99 48

1.0 1.00 0.98 0.99 43

accuracy 0.99 91

macro avg 0.99 0.99 0.99 91

weighted avg 0.99 0.99 0.99 91

isto significa:

A saída apresentada mostra diversas métricas, sendo elas:

Precision (Precisão): É a proporção de verdadeiros positivos (TP) em relação ao total de previsões positivas do modelo (TP + falsos positivos - FP).

No contexto deste relatório, temos:

* Precisão para a classe 0 (rótulo 0.0): 0.98 ou 98%
* Precisão para a classe 1 (rótulo 1.0): 1.00 ou 100%

Recall (Revocação ou Sensibilidade): É a proporção de verdadeiros positivos (TP) em relação ao total de exemplos verdadeiros da classe no conjunto de teste (TP + falsos negativos - FN).

No contexto deste relatório, temos:

* Revocação para a classe 0 (rótulo 0.0): 1.00 ou 100%
* Revocação para a classe 1 (rótulo 1.0): 0.98 ou 98%

F1-score (Medida F1): É a média harmônica entre a precisão e a revocação. Essa métrica é útil quando você deseja considerar tanto a precisão quanto a revocação em conjunto, especialmente em problemas de desequilíbrio de classe.

No contexto deste relatório, temos:

* Medida F1 para a classe 0 (rótulo 0.0): 0.99
* Medida F1 para a classe 1 (rótulo 1.0): 0.99
* Support (Suporte): É o número de amostras verdadeiras para cada classe presente no conjunto de teste.
* Accuracy (Acurácia): É a proporção de previsões corretas em relação ao total de previsões do modelo. No contexto deste relatório, a acurácia é de 0.99 ou 99%.
* Macro avg (Média Macro): É a média não ponderada das métricas de precisão, revocação e F1-score para todas as classes.
* Weighted avg (Média Ponderada): É a média ponderada das métricas de precisão, revocação e F1-score, levando em consideração o suporte (número de amostras verdadeiras) de cada classe.

Em resumo, o relatório de classificação fornece uma visão detalhada do desempenho do modelo de classificação para cada classe individualmente, bem como uma média geral. É uma ferramenta útil para avaliar o comportamento do modelo em tarefas de classificação com classes desbalanceadas. No exemplo dado, o modelo possui uma alta precisão, revocação e medida F1 para ambas as classes, indicando um bom desempenho na tarefa de classificação.

### Escolha de features e reanalise de dados

Na tentativa de resolver o problema de alta acurácia provocado pelo balanceamento produzido por undersampling, vamos aplicar técnicas de oversampling e checar seu resultado, mas para isto será necessário checar algumas informações antes. Primeiramente, a técnica de oversamplig irá aumentar significamente o número dos dados, pois a técnica irá gerar novos dados chamados sintéticos, ou seja, que originalmente não existe no conjunto de dados original, a técnica utilizada irá gerar esses dados da classe minoritária até alcançar os dados da classe majoritária, conforme já explicado no item 2.3 deste trabalho.

Ao se trata-se de adicionar dados e isto irá causar um maior esforço computacional, será realizado então uma nova análise dos dados para escolher quais features serão selecionadas. A melhor maneira de escolher estes features é checar os dados quanto a sua correlação.

Para realizar isto usamos uma junção do comando de mapa de calor com a função que gera a correção, fazendo:

plt.figure(figsize=(10,10))

sb.heatmap(df.corr() > 0.8,

annot=True,

cbar=False)

plt.show()

Código QR

Descrição gerada automaticamente com confiança baixaFigura 18 – Geração de gráfico onde mostra a correlação/mapa de calor do conjunto de dados

O gráfico acima indica fortes correlações entre os dados, mas como são muitos dados e o indicativo acima é apenas de forma binária, vamos reescrever o comando para apresentar os dados com uma variação numérica de 0 a 1.

corr = df.corr()

mask = np.triu(np.ones\_like(corr, dtype=np.bool))

f, ax = plt.subplots(figsize=(20, 20))

cmap = sns.diverging\_palette(250, 25, as\_cmap=True)

sns.heatmap(corr, mask=mask, cmap=cmap, vmax=None, center=0, square=True, annot=True, linewidths=.5, cbar\_kws={"shrink": .9})

plt.show()

Gráfico

Descrição gerada automaticamente

Figura 19 – Geração de gráfico de mapa de calor com variação numérica de 0 a 1.

Analisando o gráfico vimos que:

* Temperatura de Ponto de orvalho possui correlação acima de 0.60 com os dados de umidade.
* Temperatura de orvalho max também possui alto valor de correlação com os dados de umidade.
* Por sua vez os dados de umidade possuem altos valores com temperaturas de orvalho e com umidade relativa do ar.
* Vento na direção horária possui correlação relevante com vento rajada.

Dados de pressões possuem alta correlação com outros dados de pressão, mas isto é fisicamente explicado, uma vez que, quando a pressão abaixa há maior possibilidade de formação de nuvens de chuva, para que haja esta condição a velocidade do vento diminui, além disto a direção do vento geralmente muda, segundo Mário Adelmo varejão Silva:

Antes de uma chuva ou tempestade, é comum que haja uma diminuição na pressão atmosférica. Uma baixa pressão pode ser um indicativo de que uma área de baixa pressão está se desenvolvendo, o que pode levar à formação de nuvens e precipitação. Além disso, em torno de áreas de baixa pressão, o vento tende a convergir e acelerar, resultando em ventos mais fortes.

Variação na direção do vento: À medida que uma frente ou sistema climático se aproxima, a direção do vento pode mudar. Isso pode ser observado como um vento que vira em direção ao norte, sul, leste ou oeste antes da chegada da chuva. Essas mudanças de direção são comuns quando sistemas frontais estão se aproximando.

Vento quente e úmido: Antes de uma chuva, pode haver uma sensação de vento quente e úmido. Isso ocorre porque a aproximação de uma frente ou sistema de baixa pressão pode transportar ar úmido de regiões próximas, o que pode aumentar a umidade e a sensação de calor antes da precipitação.

Ventos de rajada: Antes de chuvas intensas ou tempestades, é comum ocorrer ventos de rajada, que são ventos rápidos e de curta duração que podem ser causados por células convectivas de tempestades.

É importante notar que essas relações são gerais e podem não se aplicar a todas as situações climáticas. O comportamento do vento e da pressão atmosférica antes da chuva pode variar significativamente em diferentes regiões e em diferentes tipos de sistemas meteorológicos. Para obter informações precisas sobre a relação entre a pressão e o vento antes da chuva em uma localização específica, é necessário consultar dados meteorológicos e previsões locais fornecidas por agências meteorológicas confiáveis (VAREJÃO-SILVA 2006).

Portanto, haja visto que as variáveis pressão e vento são intrínsecas, resultando fisicamente na mesma informação, e alguns dados de temperatura e umidade podem trazer os mesmos dados, além de querer evitar o problema de sobreamostragem no conjunto de dados, minhas novas features serão compostas apenas das colunas:

* temperatura do ponto de orvalho (°c)
* temperatura orvalho max. na hora ant. (aut) (°c)
* umidade rel. max. na hora ant. (aut) (%)
* umidade relativa do ar, horaria (%)
* vento, direção horaria (gr) (° (gr))
* vento, rajada maxima (m/s)

Também foi analisado estes dados quanto a sua distribuição normal em gráfico de histograma, tornando possível a identificação de padrões, possíveis outliers ou informações sobre a dispersão dos dados em cada variável.

Gráfico, Histograma

Descrição gerada automaticamente

Figura 20 – Geração de gráfico para verificação de distribuição normal das features escolhidas.

O gráfico acima nos dá uma excelente visão com relação a distribuição normal mas não uma ideia de dispersão de dados e Outliers, por isso, será realizado um comando para checar estas informações em gráfico de box plot.

Gráfico, Gráfico de caixa estreita

Descrição gerada automaticamente

Figura 21 – Geração de gráfico boxplot para checagem de outliers das features.

O gráfico acima mostra claramente a presença de Outliers, mas como as colunas possuem alto correlação de dados, e são informações coletadas na natureza, foi tomada a decisão de deixar estes dados, como estão. Segundo Mohana Roopa em seu livro intitulado “Rainfall Prediction using Machine Learning”, ao excluir outliers, a distribuição dos dados pode ser alterada, o que pode afetar a análise estatística e a modelagem (Y. Mohana, 2019).

### Dividindo a Base em conjuntos de Treinamento e Teste

Para treinar o modelo efetivamente, temos que dividir a base em conjunto de treinamento e teste é neste momento que vamos aplicar a técnica de oversampling utilizando uma função do Python chamada de RandomOversampling, que irá gerar dados aleatórios na classe minoritária até se igualar a classe majoritária.

import pandas as pd

import numpy as np

In [37]:

# selecione as colunas desejadas

features = df[['TEMPERATURA DO PONTO DE ORVALHO (°C)', 'TEMPERATURA ORVALHO MAX. NA HORA ANT. (AUT) (°C)', 'UMIDADE REL. MAX. NA HORA ANT. (AUT) (%)', 'UMIDADE RELATIVA DO AR, HORARIA (%)', 'VENTO, DIREÇÃO HORARIA (gr) (° (gr))', 'VENTO, RAJADA MAXIMA (m/s)']].values

target = df['houve\_chuva'].values

from sklearn.model\_selection import train\_test\_split

from imblearn.over\_sampling import RandomOverSampler

# Divida os dados em conjuntos de treinamento e validação

X\_train, X\_val, Y\_train, Y\_val = train\_test\_split(features, target, test\_size=0.2, stratify=target, random\_state=2)

# Equilibre os dados de treinamento

ros = RandomOverSampler(sampling\_strategy='minority', random\_state=22)

X\_train\_resampled, Y\_train\_resampled = ros.fit\_resample(X\_train, Y\_train)

# Verifique o tamanho de cada lista

print(len(features), len(target))

print(len(X\_train), len(Y\_train), len(X\_val), len(Y\_val))

print(len(X\_train\_resampled), len(Y\_train\_resampled))

365 365 Out[40]:

292 292 73 73

454 454

O resultado indica que os dados originais possuem 365 observações (dias do ano), com 365 valores para a variável alvo ('houve\_chuva') e 6 variáveis explicativas (features).

Após dividir os dados em conjuntos de treinamento e validação usando a função train\_test\_split, foram obtidas 292 observações para o conjunto de treinamento (X\_train) e 73 observações para o conjunto de validação (X\_val), sendo que ambos os conjuntos possuem correspondentes para a variável alvo (Y\_train e Y\_val, respectivamente).

Em seguida, os dados de treinamento foram equilibrados usando a técnica de oversampling aleatório (RandomOverSampler) para balancear as classes da variável alvo 'houve\_chuva'. Como resultado, foram geradas 454 observações ressampled para o conjunto de treinamento (X\_train\_resampled) e 454 valores correspondentes para a variável alvo (Y\_train\_resampled).

### Normalizando o conjunto de dados

Ao executarmos o comando de treinamento e teste com o atual conjunto de dados vamos ter:

from xgboost import XGBClassifier

from sklearn.svm import SVC

from sklearn import metrics

models = [LogisticRegression(), XGBClassifier(), SVC(kernel='rbf', probability=True)]

for i in range(3):

models[i].fit(X\_train\_resampled, Y\_train\_resampled)

print(f'{models[i]} : ')

train\_preds = models[i].predict\_proba(X\_train\_resampled)

print('Training Accuracy : ', metrics.roc\_auc\_score(Y\_train\_resampled, train\_preds[:,1]))

val\_preds = models[i].predict\_proba(X\_val)

print('Validation Accuracy : ', metrics.roc\_auc\_score(Y\_val, val\_preds[:,1]))

print()

LogisticRegression() :

Training Accuracy : 0.8005589085757534 Out[42]:

Validation Accuracy : 0.8464912280701754

Isto nos mostra que o resultado apresentado melhorou bastante, e está com uma precisão alta, porém para termos resultados mais fidedignos vamos aplicar a normalização e refazer os comandos.

from sklearn.preprocessing import StandardScaler

# Inicialize o scaler

scaler = StandardScaler()

# Normalizar as features de treinamento

X\_train\_resampled = scaler.fit\_transform(X\_train\_resampled)

# Normalizar as features de validação

X\_val = scaler.transform(X\_val)

X\_val\_norm = scaler.transform(X\_val)

from sklearn import metrics

metrics.plot\_confusion\_matrix(models[2], X\_val\_norm, Y\_val)

plt.show()

No último comando acima foi solicitado ao interpretador do python normaolizar e imprimir uma Matriz de confusão para checar os dados, a matriz gerada foi esta:

Gráfico

Descrição gerada automaticamente

Figura 22 – Geração de matriz de confusão dos features selecionados.

A matriz de confusão mostra uma melhor performance graficamente, e para que fique mais claro, iremos solicitar ao sistema o mesmo comando utilizado anteriormente foi checado os dados utilizamos undersampling, temos:

precision recall f1-score support

0 0.91 0.86 0.88 57

1 0.58 0.69 0.63 16

accuracy 0.82 73

macro avg 0.74 0.77 0.76 73

weighted avg 0.84 0.82 0.83 73

Ou seja, temos agora:

Precision: A precisão é a proporção de verdadeiros positivos (TP) em relação a todos os valores previstos como positivos (TP + falsos positivos - FP).

* Precisão para a classe 0 (rótulo 0.91): 0.91 ou 91%
* Precisão para a classe 1(rótulo 0.58): 0.58 ou 58%

Recall: O recall (ou sensitividade) é a proporção de verdadeiros positivos (TP) em relação a todos os valores verdadeiramente positivos (TP + falsos negativos - FN).

Para a classe 0, o recall é de 0.86, o que significa que o modelo identificou corretamente 86% das amostras pertencentes à classe 0. Para a classe 1, o recall é de 0.69, indicando que o modelo conseguiu detectar 69% das amostras verdadeiramente da classe 1.

F1-score: O F1-score é a média harmônica entre precisão e recall e é útil quando há um desequilíbrio entre as classes. Para a classe 0, o F1-score é de 0.88, e para a classe 1, o F1-score é de 0.63.

Support: A coluna "support" mostra o número de amostras verdadeiras em cada classe, ou seja, o número total de amostras para a classe 0 é 57, e para a classe 1 é 16.

Accuracy: A acurácia é a proporção de amostras classificadas corretamente em relação ao total de amostras. Nesse caso, a acurácia é de 0.82, o que significa que o modelo classificou corretamente 82% das amostras.

Macro avg: A média macro é a média das métricas de precisão, recall e F1-score para todas as classes. Neste caso, a média macro das três métricas é de aproximadamente 0.76.

Weighted avg: A média ponderada é a média das métricas de precisão, recall e F1-score, ponderada pelo suporte (número de amostras verdadeiras) de cada classe. Neste caso, a média ponderada das três métricas é de aproximadamente 0.83.

Em resumo, o modelo possui agora bom desempenho na classe 0, com alta precisão e recall, e um desempenho um satisfatório na classe 1. A acurácia geral é de 82%.

5. Resultados Experimentais

Foi verificado que o melhor método que se adequou ao modelo foi o oversampling, produzindo dados sintéticos proporcionais a amostra, realizando uma performance aceitável de aprendizado e acertos, uma vez que o undersampling produziu uma performance irreal de 100% de precisão para amostras da classe 0 e 98% para amostras de classes 1, além de dados de acurácia de 99%. Utilizamos o undersampling para tentar resolver problemas ao tentar executar os comandos de Machine Learning sem utilizar métodos de pré-processamento de dados para lidar com classes desbalanceadas.

Quando não utilizamos nenhum método, obtivemos outro dado irreal de 97% de acurácia, executando um classification report, vemos que as demais métricas se mantenham irreais, como médias macro e média ponderada de 97% e 98% respectivamente, não sendo números em que é possível dizer que o modelo esta funcionando.

Para execução do comando, classification report do conjunto de dados sem um maior tratamento(pré-processamento), temos :

print(classification\_report(y\_test,y\_pred))

precision recall f1-score support

0 0.99 0.99 0.99 72

1 0.95 0.95 0.95 20

accuracy 0.98 92

macro avg 0.97 0.97 0.97 92

weighted avg 0.98 0.98 0.98 92

Com undersampling:

print(classification\_report(y\_test,y\_pred))

precision recall f1-score support

0.0 0.98 1.00 0.99 48

1.0 1.00 0.98 0.99 43

accuracy 0.99 91

macro avg 0.99 0.99 0.99 91

weighted avg 0.99 0.99 0.99 91

com oversampling:

precision recall f1-score support

0 0.91 0.86 0.88 57

1 0.58 0.69 0.63 16

accuracy 0.82 73

macro avg 0.74 0.77 0.76 73

weighted avg 0.84 0.82 0.83 73

6. Conclusões e Trabalhos Futuros

## Conclusões

Os dados foram tratados de 3 formas:

* Apenas com tratamento básico de dados (evitando que a amostra contenha erros de dados faltantes, erros de tipo de dados, dados nulos e etc...)
* Tratando dados com métodos de pré-processamento utilizando Undersamplig
* Tratando dados com métodos de pré-processamento utilizando Oversampling

Temos então em um primeiro momento, resultados não factíveis com o uso de todos os dados do conjunto disponíveis, e isto resultou em um problema de sobreamostragem, fazendo com que a máquina aprenda muito mais com a classe 0 da nossa variável alvo do que com a classe 1, portanto a máquina acabou aprendendo quando não ouve chuva do que quando houve resultando em um “motor” nada útil e sem responder a questão problema, é possível prever chuvas utilizando aprendizado de máquina?

Em um segundo momento do desenvolvimento do modelo, utilizamos o método Undersamplig em que obtivemos resultados piores do que com todo o conjunto de dados, novamente um modelo sem utilidade e sem responder à questão problema.

Ao utilizarmos o método oversamplig obtivemos ótimos resultados, criando um “motor” que pode ser implementado para prever chuvas utilizando dados históricos na região de Copacabana, em se tratando de uma região que atrai muitos turistas seriam de grande utilidade comercial para: hotéis, pontos turísticos, todo comercial local em função do turismo e até mesmo para evitar catástrofes. Respondendo assim a questão problema, é possível prevermos chuvas utilizando Machine Learning? ... Sim.

## Trabalhos Futuros

Vemos que a aprendizagem de máquina vem sendo utilizada em larga escala em diferentes áreas de conhecimento, e é esperado que continue a auxiliar a humanidade em obter uma melhor qualidade de vida, muito ainda é discutido quanto à seu real benefício, várias personalidades importantes do meio científico são contra o seu desenvolvimento, mas é importante notarmos que a sociedade ao longo de sua história sempre esteve em constante mudança e as mudanças muitas vezes não são fácies de se implementar ou ter aceitação em larga escala.

“Hoje já são utilizados modelos de aprendizagem de máquina para execução de trabalhos de espaços confinados na Petrobrás”, conforme engenheiro entrevistado para este trabalho Jose Carlos Caldas, engenheiro de manutenção da Transpetro, são utilizados modelos de aprendizagem de máquina para atendimento ao cliente, poderão existir modelos que auxiliem médicos em decisões com diagnósticos, sistemas de orientação mais precisos para aviões, barcos e até mesmo em rodovias evitando maiores trânsitos em grandes cidades.

A aplicação da tecnologia servirá de ponto de virada na utilização diária, (como aconteceu com celular a anos atrás) e será utilizada em ampla escala, é necessária ainda uma regulamentação de seu uso, além de medidas de proteção para o que os cientistas chamam de “singularidade”, que consiste na tomada de consciência de uma inteligência autômata.

Em termos de previsão na utilização de dados históricos, ainda será possível melhorar muito o sistema, como sua utilização em tempo real aliada á tomada de decisão e isto irá requerer mais processamento por parte de hardwares utilizados. Fazendas de processamentos poderão ser utilizadas para processar uma grande massa em tempo real.

Revisão bibliográfica

[1] Y. Mohana Roopa. Rainfall Prediction using Machine Learning Algorithm. LAMBERT Academic Publishing July 12, 2019

[2] Grus Joel. Data Science do Zero. Alta Books Janeiro 2020

[3] Moulana Mohammed. Prediction Of Rainfall Using Machine

Learning Techniques. Internacional Journal of Scientifc & Technology Research volume 9, issue 01, january 2020

[4] CNA PIB do Agronegócio alcança participação de 26,6% no PIB brasileiro em 2020. 2020. Disponível em [PIB do Agronegócio alcança participação de 26,6% no PIB brasileiro em 2020 | Confederação da Agricultura e Pecuária do Brasil (CNA) (cnabrasil.org.br)](https://cnabrasil.org.br/publicacoes/pib-do-agronegocio-alcanca-participacao-de-26-6-no-pib-brasileiro-em-2020)

[5] Vieira, W. G., & Silva, N. (2005). PRENÚNCIO DE CHUVAS PELAS AVES NA PERCEPÇÃO DE MORADORES DE COMUNIDADES RURAIS NO MUNICÍPIO DE SOLEDADE-PB, BRASIL. Revista de Saúde Pública, 39(12), 1844-1844. Disponível em <http://ve.scielo.org/scielo.php?script=sci_arttext&pid=S0378-18442005001200008>

[6] Z. M. Yaseen, A. Hussain, A. El-Shafie. A review of artificial intelligence techniques in predicting rainfall and drought" Journal of Hydrology (2019) p. 200-214

[7] Christopher M. Pattern Recognition and Machine Learning. Information Science and Statistics. Springer, New York, NY, 1 edition, August 2006

[8] Alice Zeng & Amanda Casari. Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists, 14 abril 2018

[9] Michael W Berry, Azlinah Mohamed, and Bee Wah Yap. Supervised and unsupervised learning for data science. Springer, 2019.

[10] Sonia Sehra, David Flores, and George D Montanez. Undecidability of underfitting in

learning algorithms. In 2021 2nd International Conference on Computing and Data

Science (CDS), pages 591–594. IEEE, 2021.

[11] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: data mining, inference, and prediction. Springer Science & Business Media.

[12] Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. Géron, A. (2019).

[13] Chawla, N.V.; Japkowicz, N.; Kotcz, Editorial: Special Issue on Learning from Imbalanced Data Sets. ACM SIGKDD Explorations. v.6. p.1-6. A. (2004)

[14] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

[15] Instituto Nacional de Metereologia INMET disponível em: [Instituto Nacional de Meteorologia - INMET](https://portal.inmet.gov.br/)

[16] Em meio à onda de calor no verão, medição de temperatura no Rio encontra percalços; entenda disponível em: [Em meio à onda de calor no verão, medição de temperatura no Rio encontra percalços; entenda - Jornal O Globo](https://oglobo.globo.com/rio/em-meio-onda-de-calor-no-verao-medicao-de-temperatura-no-rio-encontra-percalcos-entenda-24866630)

[17] VAREJÃO-SILVA, Mário. Meteorologia e Climatologia. Brasília: Pax, 2001.

[18] Simon da Fonseca, Jairo. Curso de Estatística. Atlas 2006.