Predição de AVCs Através do Machine Learning

Andrei Gonçalves
andrei.massaini@hotmail.com
ICEI - PUC-MG
Belo Horizonte, Minas Gerais, Brasil

Artur Kazuo
afuzikawa@sga.pucminas.br
ICEI - PUC-MG
Belo Horizonte, Minas Gerais, Brasil

João Vítor Belchior
jbelchior@sga.pucminas.br
ICEI - PUC-MG
Belo Horizonte, Minas Gerais, Brasil

Matheus Marcolino
matheus.marcolino.1320642@sga.pucminas.br
ICEI - PUC-MG
Belo Horizonte, Minas Gerais, Brasil

Vinicius Souza
vfsouza@sga.pucminas.br
ICEI - PUC-MG
Belo Horizonte, Minas Gerais, Brasil

Abstract

Este artigo tem como objetivo analisar o dataset "Stroke Prediction Dataset" [1], com o objetivo de identificar características relevantes para a predição de AVCs. Para isso, utilizaremos uma metodologia baseada em Árvore de Decisão e Random Forest. Todos algoritmos foram implementados via Python, utilizando principalmente as bibliotecas Pandas, SKLearn e MatPlotLib.

Keywords: Datasets, AVC, Machine Learning, Decision Tree, Random forest

ACM Reference Format:
Andrei Gonçalves, Artur Kazuo, João Vítor Belchior, Matheus Marcolino, and Vinicius Souza. 2023. Predição de AVCs Através do Machine Learning. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/none

1 Introdução

O AVC (Acidente Vascular Cerebral), acontece quando um vaso sanguíneo se rompe, acarretando em uma hemorragia cerebral, ou quando o fluxo de sangue para a região é interrompido. Ele é responsável pela maior taxa de mortes entre as doenças no Brasil, com cerca de 307 vítimas fatais por dia, e o número de vítimas aumenta ainda mais quando consideramos as pessoas que sobreviveram com sequelas[2]. Em relação às sequelas, cerca de 70% sofrem um AVC não voltam a trabalhar e 50% delas precisam de auxílio de outras pessoas pra realizar atividades cotidianas[3]. Para mitigar esses possíveis danos, é essencial que o diagnóstico seja feito mais rápido possível.

Colocada a gravidade da doença, é importante conseguir prever quais perfis e características que mais se relacionam com as vítimas, para que assim seja possível prevenir e tratar possíveis comorbidades antes que ocorram. Além disso, mesmo que não seja possível prever, essas análises também podem facilitar um diagnóstico durante o socorro, dessa forma diminuindo as chances de óbitos.

Conference'17, July 2017, Washington, DC, USA
2023. ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/none

2 Descrição da Base de Dados

A base de dados utilizada foi a Stroke Prediction Dataset [1], que contém informações sobre fatores demográficos, de saúde e estilo de vida de pacientes que passaram por uma triagem médica. O dataset é composto por 5111 instâncias, em que cada uma representa um paciente.

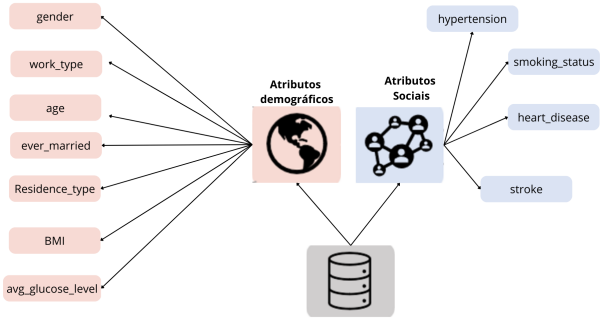

Figure 1. Atributos contidos no dataset

1. **Gender**: gênero do paciente (masculino ou feminino), foi codificada através do label encoder.
2. **Age**: idade do paciente em anos
3. **Hypertension**: Indica se o paciente tem hipertensão (1 para sim, 0 para não)
4. **Heart disease**: Indica se o paciente tem doença cardíaca (1 para sim, 0 para não).
5. **Ever married**: Indica se o paciente já foi casado ou não (sim ou não), foi codificada através do label encoder.
6. **Work type**: Tipo de trabalho do paciente (criança, Privado, Autônomo, Sem trabalho), foi dummificada através do OneHotEncoder.
7. **Residence type**: Tipo de residência do paciente (urbana ou rural), foi codificada através do label encoder.
8. **Avg glucose level**: Nível médio de glicose no sangue em mg/dL
9. **BMI**: Índice de massa corporal do paciente
10. **Smoking status**: Status de tabagismo do paciente (nunca fumou, ex-fumante, fumante ou desconhecido), foi dummificada através do OneHotEncoder.

11. **Stroke**: Indica se o paciente sofreu um derrame (1 para sim, 0 para não)

Os atributos do dataset original estão divididos em três tipos de variáveis:

1. **Variáveis categóricas:** gender, ever married, work type, Residence type e smoking status. Essas variáveis descrevem uma característica do paciente que não pode ser quantificada em termos de quantidade, mas sim em termos de categorias.
2. **Variáveis numéricas contínuas:** age, avg glucose level e bmi. Essas variáveis são quantitativas e representam medidas numéricas que variam continuamente.
3. **Variáveis binárias:** hypertension, heart disease e stroke. Essas variáveis são binárias, ou seja, assumem apenas dois valores possíveis: 0 ou 1, indicando a presença ou ausência de uma determinada condição médica.

3 Pré Processamento

3.1 Valores Ausentes

A primeira etada do pré processamento consistiu em identificar e tratar os valores ausêntes do dataset. No total, encontramos 201 instâncias que possuiam valores ausentes na coluna 'bmi', como observado na imagem abaixo:

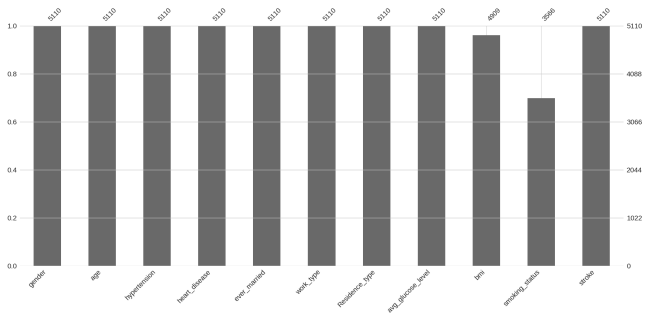

Figure 2. Proporção de dados ausentes por coluna:

Para lidar com os valores ausentes, utilizamos o algoritmo Simple Imputer da biblioteca scikit-learn (sklearn). Optamos pela estratégia de imputação que consiste em calcular a média dos valores não ausentes da coluna 'bmi' em relação às demais instâncias do conjunto de dados.

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='mean')
imputer. fit_transform
df[[ 'bmi']] = imputer. fit_transform (df[[ 'bmi' ]])
```

3.2 Codificação de variáveis:

Após essa etapa, procedemos à codificação das variáveis. Como não havia nenhum atributo que expressava uma ordem, propriamente dita,aplicamos o Label Encoder em todo o conjunto de dados:

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
df = df.apply(label_encoder. fit_transform )
```

3.3 Analisando Correlações:

Após a codificação das variáveis, tornou-se viável analisar as potenciais correlações entre os atributos de entrada e o atributo de classe. Para realizar essa análise, geramos uma matriz de correlação e estabelecemos um limite ou *threshold* de 0.6 em relação ao atributo de classe.

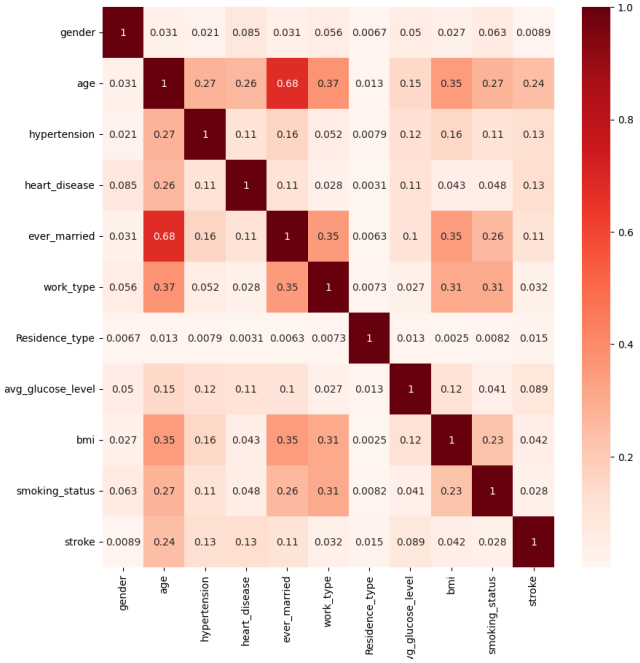

Figure 3. Matriz de Correlação

Ao analisar a matriz de correlação, chegamos à conclusão de que não há nenhuma correlação significativa entre os atributos de entrada e o atributo de classe. Portanto, não foi necessário a remoção de nenhuma coluna do dataset.

3.4 Balanceamento

A última etapa do pré processamento, consistiu em tratar o desbalanceamento evidente da base de dados, como pode ser observado na Na Figura 4 abaixo:

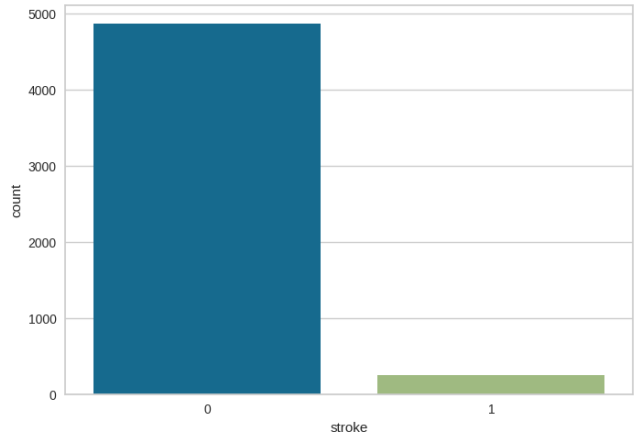

Figure 4. Distribuição original

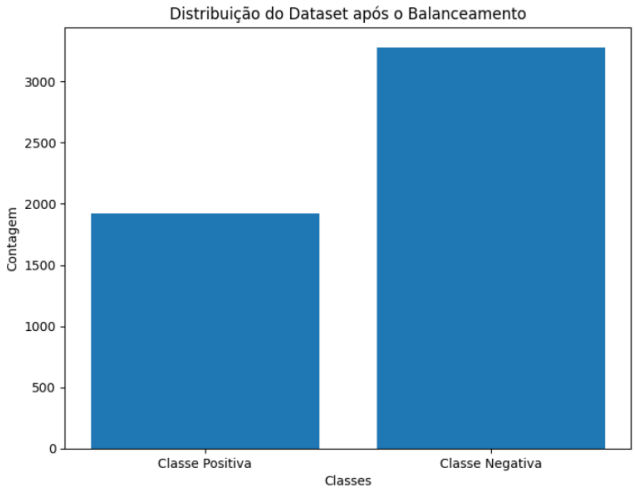

Figure 5. Distribuição após o balanceamento.

A técnica utilizada para tratar o desbalanceamento vigente, foi o *SMOTE Tomek*. Essa técnica combina as abordagens de *SMOTE (Synthetic Minority Over-sampling Technique)* e *Tomek Links* para tratar o desbalanceamento de classes em conjuntos de dados.

O *Smote* é utilizado para gerar instâncias sintéticas da classe minoritária, enquanto o Tomek Links identifica e remove as instâncias de ambas as classes que estão próximas e são consideradas ambíguas.

A combinação dessas técnicas busca criar um conjunto de treinamento balanceado, ao mesmo tempo em que melhora a separabilidade entre as classes. Essa abordagem pode ser eficaz para melhorar o desempenho de modelos de aprendizado de máquina em problemas de desbalanceamento de classes.

O hiperparâmetro utilizado no balanceamento foi o "sampling strategy", que é usado para definir a proporção desejada entre as classes minoritária e majoritária após a aplicação do *Smote Tomek*. Ele indica qual a relação de balanceamento entre as classes que você deseja alcançar. No nosso caso, estipulamos o valor como sendo 0.6, ou seja desejamos aumentar a classe minóritaria para que ela represente 60 porcento da classe majoritária.

Balanceamento:

```
from imblearn.combine import SMOTETomek
os = SMOTETomek(sampling_strategy=0.6)
```

Após o balanceamento, a proporção de classe positiva e negativa ficou da seguinte maneira:

3.5 Divisão do conjunto de teste e treino:

As tabelas a seguir demonstram como se deu a divisão do conjunto de dados entre teste e treino:

Dataset	Classes Positivas	Classes Negativas
Treinamento	174	3403
Teste	75	1458

Table 1. Divisão do conjunto de dados antes do balanceamento.

Conjunto de Treinamento	Quantidade de Instâncias
Positivas	1914
Negativas	3276

Table 2. Divisão do conjunto de treinamento após o balanceamento.

Vale ressaltar que o balanceamento foi aplicado apenas ao conjunto de treinamento, tanto durante a avaliação cruzada quanto na avaliação individual dos modelos. Dessa forma, o conjunto de teste permaneceu "intacto", afim de previnir o vazamento de dado Garantindo que o modelo seja testado e validado apenas com instâncias desconhecidas, por assim dizer.

4 Algoritmos de Aprendizagem utilizados:

4.1 Decision Tree (Árvore de Decisão):

A Árvore de Decisão é um algoritmo de aprendizado supervisionado que constrói um modelo de previsão em forma de estrutura de árvore. A árvore é composta por nós internos que representam testes em atributos e nós folha que representam as classes ou valores de saída. A ideia central é dividir

o conjunto de dados com base em atributos selecionados, de forma que cada divisão maximize a homogeneidade dos dados resultantes.

O processo de construção da árvore envolve a seleção do melhor atributo para dividir o conjunto de dados em cada nó, utilizando critérios como ganho de informação ou índice de Gini. Essa divisão é repetida recursivamente em cada subconjunto até que sejam alcançadas condições de parada, como atingir um número mínimo de amostras em um nó ou quando não há mais atributos disponíveis para dividir.

As árvores de decisão são atraentes por sua interpretabilidade e facilidade de explicar as decisões tomadas. No entanto, elas podem ser propensas a overfitting se não forem adequadamente controladas.

4.2 Random Forest (Floresta Aleatória):

A Floresta Aleatória é uma técnica de aprendizado de conjunto (ensemble) baseada em Árvores de Decisão. Ela cria uma coleção de árvores de decisão individuais, onde cada árvore é construída em um subconjunto aleatório do conjunto de dados de treinamento

A principal ideia por trás da Random Forest é combinar as previsões de várias árvores para obter uma previsão final mais precisa e robusta. Durante o treinamento, para cada árvore, o subconjunto de dados é amostrado aleatoriamente com substituição (conhecido como amostragem bootstrap). Além disso, em cada divisão de nó, apenas um subconjunto aleatório de atributos é considerado para limitar a correlação entre as árvores.

Durante a fase de previsão, cada árvore na floresta produz uma previsão e a classe mais frequente ou a média das previsões é escolhida como a previsão final da floresta.

As Random Forests têm várias vantagens, como serem menos suscetíveis ao overfitting em comparação com árvores individuais, serem capazes de lidar com conjuntos de dados grandes e de alta dimensionalidade e fornecerem uma estimativa da importância dos atributos para o problema.

5 Metodologia de testes

Para a avaliação de desempenho de ambos os modelos mencionados anteriormente, estipulamos dois tipos de testes:

5.1 Tipos de avaliação:

Avaliação individual: Realizamos o treinamento do modelo utilizando os dados de treinamento do dataset original de forma balanceada. Em seguida, utilizamos o conjunto de validação para fazer a predição e avaliar o desempenho do mesmo.

Validação cruzada: Realizamos a validação cruzada do modelo usando o método K-fold, onde o conjunto de dados é dividido em K dobras. Cada dobra passa por um processo de balanceamento, treinamento, validação individual assim como o *Grid Search*, que busca os melhores parâmetros em todos os *folds*. Calculamos as métricas de desempenho para cada dobra separadamente e, além disso, calculamos a média das métricas entre todos os folds. Ao final da validação cruzada, selecionamos o melhor modelo treinado e o utilizamos para fazer a predição das instâncias.

6 Avaliação individual:

Para a avaliação individual,foram considerados os seguintes hiperparâmetros na pesquisa:

- *criterion: gini, Entropy*
- *max depth: None, 2, 4, 6, 8, 10*
- *max features: None, sqrt, log2, 0.2, 0.4, 0.6, 0.8*

Os respectivos hiperparametros e métricas obtidas foram:

Table 3. Melhores Parâmetros na avaliação individual

Modelo	Parâmetro	Valor
Random Forest	*criterion*	*gini*
Random Forest	*max_depth*	*None*
Random Forest	*max_features*	*0.6*
Decision Tree	*criterion*	*entropy*
Decision Tree	*max_depth*	*None*
Decision Tree	*max_features*	*None*

Table 4. Métricas obtidas avaliação individual

Algoritmo	Classe	Precisão	Recall	F-score	Área ROC
Random Forest	0	0.96	0.94	0.95	0.20
Random Forest	1	0.17	0.24	0.20	0.79
Decision Tree	0	0.97	0.91	0.93	0.40
Decision Tree	1	0.17	0.37	0.23	0.59

7 Validação Cruzada

A seguir, veremos as métricas obtidas ao realizar a validação cruzada de ambos os modelos:

7.1 K-Folds

7.1.1 Random forest. Na tabela abaixo, estão disponíveis os hiperparâmetros utilizados em cada um dos folds da Random forest.

Fold	Criterion	Max_Depth	Max_Features
1	*Entropy*	None	0.4
2	*Entropy*	None	0.2
3	*Entropy*	None	0.4
4	Gini	None	Sqrt
5	Gini	None	0.6

Além disso, coletamos a média do F1 Score, Precisão e Recall para as classes superiores e inferiores, conforme apresentado na tabela a seguir:

Métrica	Classe Superior	Classe Inferior
F1 Score Médio	0.95	0.18
Recall Médio	0.93	0.24
Precisão Média	0.96	0.15

7.1.2 Decision Tree. A tabela abaixo mostra os hiperparâmetros utilizados em cada fold da Decision Tree:

Fold	Criterion	Max_Depth	Max_Features
1	Gini	None	log2
2	*Entropy*	None	sqrt
3	*Entropy*	None	0.8
4	Gini	None	log2
5	Gini	None	log2

Também coletamos a média do F1 Score, Precisão e Recall para as classes superiores e inferiores, conforme apresentado na tabela a seguir:

Métrica	Classe Superior	Classe Inferior
F1 Score Médio	0.94	0.16
Recall Médio	0.93	0.2
Precisão Média	0.96	0.13

7.2 Comparação

Após obtermos os resultados dos modelos Random Forest e Decision Tree por meio do processo de validação cruzada com k-folds, procedemos à comparação das métricas de desempenho e da curva AUC (Area Under the Curve).

Para uma análise visual da performance dos modelos, plotamos as curvas AUC. O gráfico AUC permite avaliar a capacidade de classificação dos modelos em diferentes pontos de corte, considerando as taxas de verdadeiros positivos e falsos positivos. Com base nesse gráfico, podemos comparar o desempenho relativo dos modelos em termos de sua capacidade discriminativa.

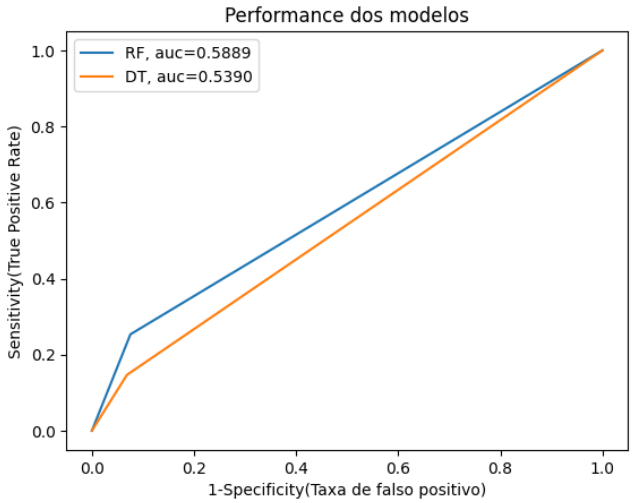

Figure 6. Gráfico AUC

Ao comparar os resultados do Random Forest e da Decision Tree no gráfico AUC, observamos que o Random Forest apresentou uma curva AUC mais próxima de 1 durante a maior parte do teste, mas se assemelhando mais com a Decision Tree no final. Dessa forma, concluímos que não houve diferença tão significativa entre ambos.

Também fizemos comparações baseadas nas métricas de Recall, Precisão e F1-Score entre os modelos, como pode ser visto abaixo:

Métrica	Decision Tree	Random forest
F1 Score Médio	0.175	0.125
Recall Médio	0.15	0.25
Precisão Média	0.11	0.15

7.2.1 Teste T. O teste T foi realizado para comparar os resultados dos modelos Random Forest e Decision Tree. O valor obtido para o T-value foi -1.25, e o valor crítico fornecido foi 1.96. Ao comparar esses valores, concluímos que não há uma diferença estatisticamente significativa entre as médias dos grupos comparados. Isso sugere que os modelos têm desempenhos semelhantes em relação às métricas avaliadas.

8 Conclusões

Com base nos dados anteriores, concluímos que a base de dados utilizada não apresentava qualidade suficiente para obter resultados satisfatórios na classificação e previsão de casos de AVC. Essa conclusão foi alcançada por meio da análise do desempenho dos modelos, que não apresentaram melhorias significativas mesmo após a aplicação de diferentes tratamentos. Além disso, atribuímos esses problemas à falta de um número expressivo de amostras, dado o desbalanceamento extremo da base de dados. Esses fatores combinados

indicam que os resultados obtidos não são confiáveis e limitam a eficácia dos modelos na tarefa de classificação de casos de AVC.

Apesar da média dos resultados e o Teste T não apontarem diferenças tão grandes entre os modelos, ao analisar individualmente cada execução, o Random forest apresentou resultados melhores.

9 Código e Database

Para mais detalhes, o código fonte está disponível em https://colab.research.google.com/drive/1psTB1H1Y7YLGq4B_HB8n_mZkQOjzsBKA?usp=sharing. Databese disponível em https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

References

[1] F. Soriano, "Stroke prediction dataset," Jan 2021. [Online]. Available: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

[2] M. da Saúde, "Acidente vascular cerebral," 2023. [Online]. Available: https://www.gov.br/saude/pt-br/assuntos/saude-de-a-a-z/a/avc

[3] Pfizer, "O que é o acidente vascular cerebral, quais os tipos, como prevenir e tratar," 2019. [Online]. Available: https://www.pfizer.com.br/noticias/ultimas-noticias/o-que-e-acidente-vascular-cerebral-AVC-tipos-prevencao-tratamento