Avaliação de dois Algoritmos de Machine Learning na Previsão do Câncer de Mama

JOÃO AUGUSTO DRUZIAN MULHO
MAURICIO DUARTE

Resumo

A humanidade busca constantemente por respostas imediatas e com alto nível de acurácia. Por isso, a Aprendizagem de Máquina (Machine Learning), considerada como uma subárea da inteligência artificial, é uma das opções viáveis, pois tem potencial para auxiliar as mais diversas áreas e possui uma gama de aplicabilidade, desde questões mais simples até aquelas de maior complexidade, como a da saúde com previsões de diagnósticos de doenças. Este trabalho visa apresentar os resultados de um estudo probabilístico para a previsão do câncer de mama com Machine Learning, no qual foram utilizados dois algoritmos e seus resultados foram comparados. Utilizou-se o SVM (Máquina de Vetores de Suporte) e o Naive Bayes. Para a execução do trabalho foram utilizadas ferramentas como a linguagem de programação Python em conjunto com algumas bibliotecas, e como plataforma para execução e desenvolvimento dos algoritmos o Google Colaboratory. Portanto, ao fim deste estudo foi possível verificar que, a técnica Machine Learning quando executado de maneira correta, com uma base de dados estruturada e tratada, pode gerar bons resultados com uma alta taxa de eficiência.

Palavras-chave*:* Máquina de Vetores de Suporte; Naive Bayes; Python; estudo probabilístico; câncer de mama.

Evaluation of two Machine Learning Algorithms in Breast Cancer Prediction

Abstract

In view of the constant search for immediate responses with a high level of assertiveness by the humanity, there is the Machine Learning, a sub-area of artificial intelligence as one of the viable options, which demonstrates the potential to help in the most diverse areas, and that has a wide range of applicability, from the simplest issues, to the health area with disease prediction. The work in question aims to present the results by executing a probabilistic study in relation to the prediction of breast cancer with Machine Learning, in which two resources were used and compared with the results of both, namely the SVM (Support Vectors Machine) and the Naive Bayes. To execute the work, tools such as the Python programming language were used together with some libraries, and Google Collaboratory as a platform for the execution and development of the algorithms. Therefore, at the end of this study, it was possible to verify that, Machine Learning, when performed correctly, with a studied and treated base, is able to obtain good results with a high rate of assertiveness.

Keywords: *Support Vector Machines;* Naive Bayes*;* Python*; probabilistic study; breast cancer.*

---

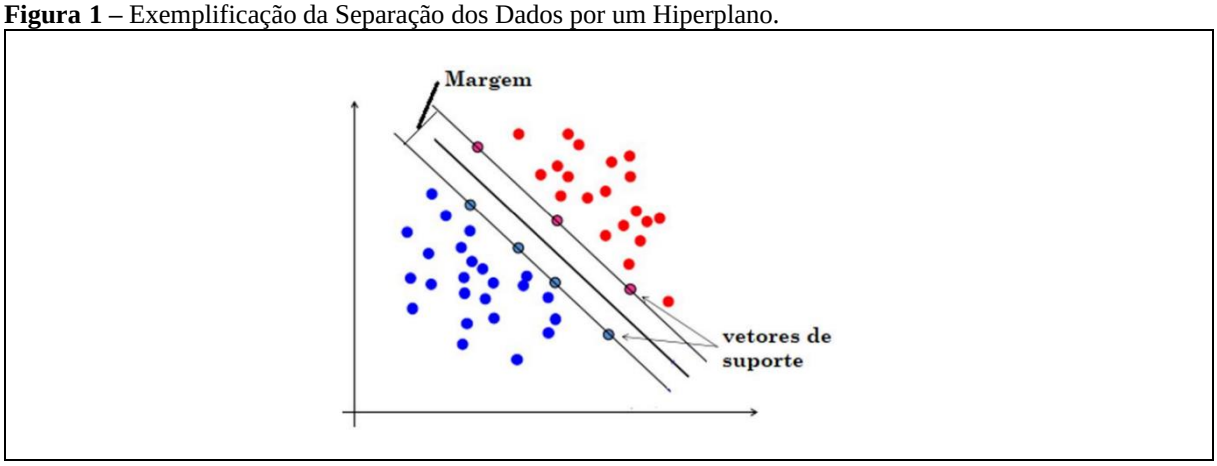
1 INTRODUÇÃO

A área de inteligência artificial, mais precisamente a subárea de Machine Learning, tem como objetivo, de forma resumida, aprender com um conjunto de dados e a partir deles gerar resultados mais condizentes possíveis com a realidade, desde que esses dados sejam devidamente tratados e os mais corretos possíveis, para potencializar, desse modo, a taxa de eficiência do algoritmo desenvolvido. Sendo assim, o aprendizado de máquina tem alta taxa de aplicabilidade nas mais diferentes áreas, com uma gama de algoritmos e formas de aprendizagem para as mais diferentes necessidades.

Para o presente trabalho foi utilizada a aprendizagem supervisionada para a execução do algoritmo, essa que é uma forma de aprendizagem a qual são tidos tanto os dados de entrada (atributos previsores), quanto os dados de saída (atributos alvo) a disposição. O principal ponto é que o algoritmo aprenda com uma base de dados histórica, identificando padrões para gerar resultados mais precisos possíveis. No caso do estudo em questão foi utilizada uma base de dados com registros de pacientes que desenvolveram ou não câncer de mama, com dados em relação às dimensionalidades e características do tumor. Além disso, têm-se duas áreas de estudo específicas dentro da aprendizagem supervisionada: a classificação (o algoritmo gera dados categóricos como resposta, por exemplo, "sim" ou "não") e a regressão (o algoritmo gera resultados numéricos como resposta). Nesse caso foi utilizada a classificação.

O algoritmo SVM (Máquina de Vetores de Suporte) é diretamente aplicado para problemas de aprendizagem supervisionada, tanto em classificação quanto em regressão, podendo ser aplicados em problemas linearmente separáveis. Os algoritmos SVM criam hiperplanos de separação dos dados que estão sendo trabalhados, para assim ser feita a classificação ou regressão deles. Na figura 1 é possível verificar a separação desses dados (linearmente separáveis), os azuis e vermelhos, por uma linha/plano, o qual é visto lateralmente. Os últimos dados que separam os dois tipos são chamados de vetores de suporte, já a distância entre eles é chamada de margem. Portanto, a ideia geral é esta: criar hiperplanos de separação de acordo com as características dos dados que estão sendo trabalhados.

Figura 1 – Exemplificação da Separação dos Dados por um Hiperplano.



Fonte: Material de Apoio (Udemy, Machine Learning com Python) – Máquina de Vetores de Suporte (SVM); Prof. Luciano Galdino.

O algoritmo Naive Bayes é um dos primeiros e mais famosos algoritmos para o estudo do aprendizado de máquina, sendo esse baseado no teorema do matemático Thomas Bayes (1701-1761), chamado de "Teorema de Bayes". Esse algoritmo é um classificador probabilístico, visto que é usado para estudos de probabilidades. Além disso, é exclusivo para problemas de classificação na aprendizagem supervisionada, e é extremamente funcional para trabalhos que envolvem variáveis categóricas.

Esse algoritmo trabalha com uma premissa: que as variáveis que estão sendo trabalhadas sejam independentes entre si, ou seja, ele desconsidera completamente a correlação entre elas. Um exemplo disso seria examinar uma roupa. A mesma roupa possui diversas características (variáveis), como tamanho, cor, sexo para qual é destinada, porém o algoritmo Naive Bayes vai analisar cada uma delas de forma individual, sem fazer ligação alguma. Por esse motivo, recebe a palavra *naive* em seu nome, pois ele é considerado "ingênuo" por tratar as variáveis de forma isolada.

Portanto, na elaboração do algoritmo para a previsão da probabilidade do desenvolvimento do câncer de mama foram aplicados algoritmos referentes à aprendizagem supervisionada em conjunto com a classificação, sendo eles o SVM (Máquina de Vetores de Suporte) e o Naive Bayes, os quais obtiveram resultados distintos em suas aplicações, sendo o objetivo do projeto em questão analisar o desempenho de cada um deles e a forma como a base de dados utilizada foi tratada para potencializar a taxa de acurácia.

2 DESENVOLVIMENTO

2.1 Resumo da Etapa de Tratamentos dos Dados

A etapa em questão visa apresentar de forma geral o tratamento da base de dados utilizada para a elaboração dos algoritmos de Machine Learning.

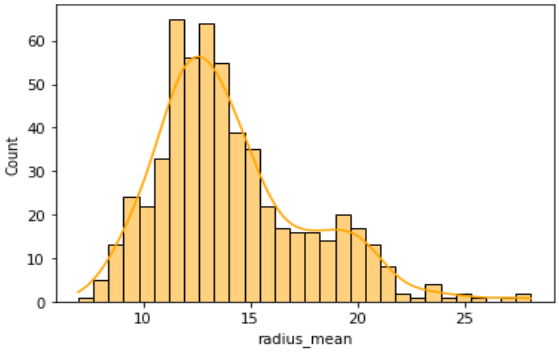
2.1.1 Base de Dados

Para a elaboração dos algoritmos de Machine Learning foi utilizada uma base de dados da Kaggle, que possui inúmeros dados das mais diversas áreas para utilização em trabalhos relacionados à ciência de dados. Nesse caso foram utilizados dados de pacientes que tiveram ou não câncer de mama. A base de dados em questão dispõe de centenas de registros com inúmeros atributos referentes às características do possível tumor, sendo trinta e três colunas de atributos e quinhentos e sessenta e nove linhas de registros disponíveis, no entanto, algumas das colunas e registros foram removidos, pois continham dados incorretos ou muito discrepantes dos outros. Para o tratamento da base foram utilizadas bibliotecas gráficas da linguagem de programação Python, sendo elas o Seaborn e Potly, essas que foram utilizadas como auxílio para a verificação da consistência dos dados de forma gráfica para garantir que não houvesse uma discrepância muito grande entre eles. Além disso, também foram verificados e tratados os valores faltantes (valores que estão em branco na tabela).

2.1.2 Análise das Variáveis (Atributos)

Na análise das variáveis foram utilizadas as bibliotecas gráficas mencionadas para a verificação da consistência dos dados, como mostrado na Figura 2, na qual foram analisados a distribuição e valores dos dados da coluna *radius mean*, que possui dados consistentes e seguem uma boa tendência, ou seja, caracterizam-se com a Distribuição de Probabilidade Normal, confirmada por meio do formato de “boca de sino” presente no gráfico.

Figura 2 – Gráfico para análise do atributo *radius_mean*.

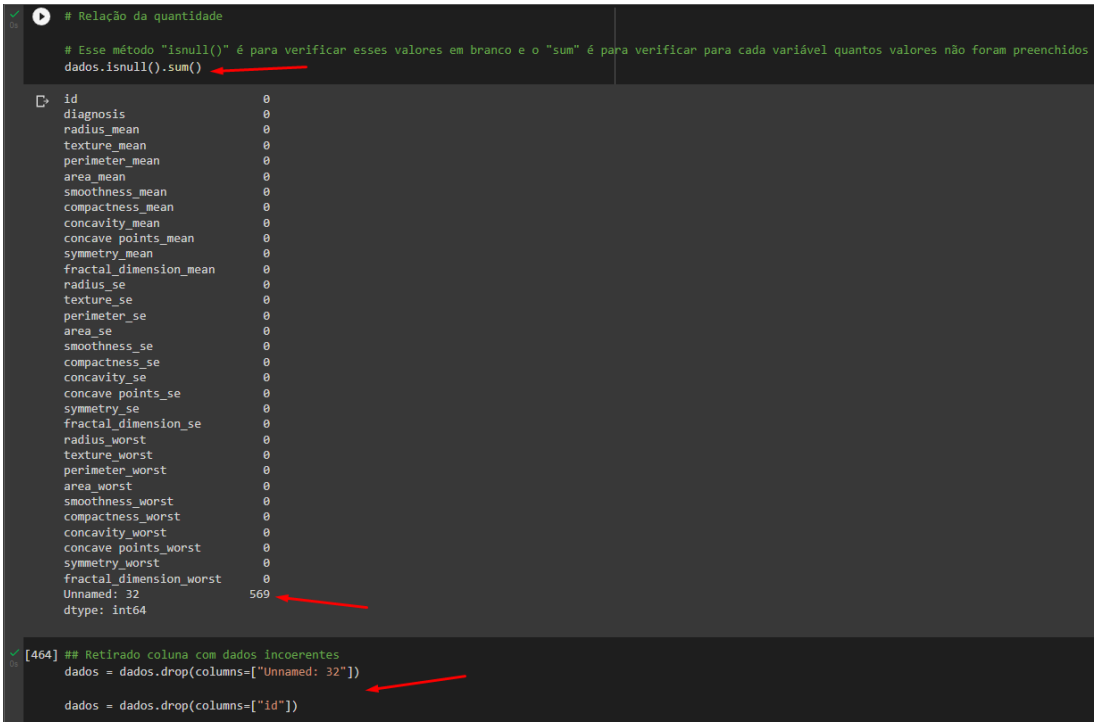


Fonte: Elaborado pelo autor.

2.1.3 Tratamento dos Valores Faltantes e de Dados Inconsistentes

Para o tratamento das variáveis faltantes foram utilizados dois métodos, o primeiro para a verificação dos valores faltantes e o segundo para somar a quantidade desses valores para cada variável da tabela. A figura 3 mostra um pequeno trecho de código e o resultado gerado por ele, no qual é possível observar uma coluna inteira que não tem relação com as outras, e possui dados faltantes, portanto, ela foi removida. Além disso, foi retirada a coluna de *id*, visto que ela não faria sentido para a previsão nos algoritmos.

Figura 3 – Trecho de Código para a Verificação e Remoção de Dados Faltantes.



```
# Relação da quantidade

# Esse método "isnull()" é para verificar esses valores em branco e o "sum" é para verificar para cada variável quantos valores não foram preenchidos
dados.isnull().sum()
```

```
id                        0
diagnosis                 0
radius_mean               0
texture_mean              0
perimeter_mean            0
area_mean                 0
smoothness_mean           0
compactness_mean          0
concavity_mean            0
concave points_mean       0
symmetry_mean             0
fractal_dimension_mean    0
radius_se                 0
texture_se                0
perimeter_se              0
area_se                   0
smoothness_se             0
compactness_se            0
concavity_se              0
concave points_se         0
symmetry_se               0
fractal_dimension_se      0
radius_worst              0
texture_worst             0
perimeter_worst           0
area_worst                0
smoothness_worst          0
compactness_worst         0
concavity_worst           0
concave points_worst      0
symmetry_worst            0
fractal_dimension_worst   0
Unnamed: 32             569
dtype: int64
```

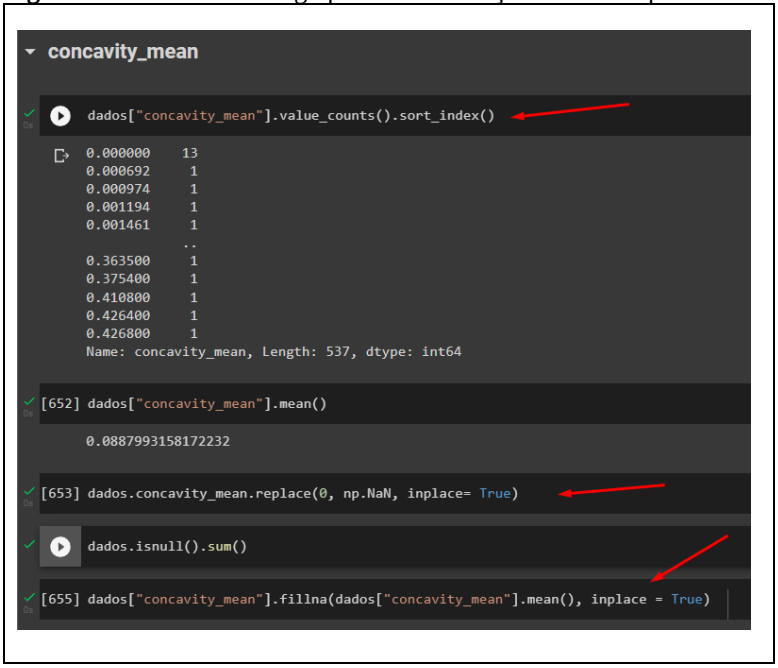
```
[464] ## Retirado coluna com dados incoerentes
dados = dados.drop(columns=["Unnamed: 32"])

dados = dados.drop(columns=["id"])
```

Fonte: Elaborado pelo autor.

2.1.4 Substituindo Valores Zero pela Média

Foi possível verificar alguns registros com valor zero em determinados atributos, os quais foram considerados valores discrepantes se comparado aos outros registros presentes na base de dados, portanto, esses valores foram substituídos nas variáveis pela média sem zeros para que não houvesse perda de registros, como demonstrado na figura 4, com o trecho de código para o atributo *concavity mean*, o qual foi substituído os valores zero por valores faltantes para ser possível substituir pela média.

Figura 4 – Trecho de Código para a Substituição dos Zeros pela Média.



Fonte: Elaborado pelo autor.

2.2 Resumo da Etapa de Classificação

Nesse item serão descritas, de forma geral, as etapas para o desenvolvimento e execução dos algoritmos de Machine Learning, utilizando o Naive Bayes e o SVM (Máquina de Vetores de Suporte), sendo essas: a etapa de pré-processamento dados, a execução dos algoritmos e a comparação dos resultados obtidos com os dados de treino e teste, tendo a acurácia e a matriz de confusão como métricas para analisar o desempenho dos algoritmos e também a validação cruzada para gerar resultados com a mistura desses dados.

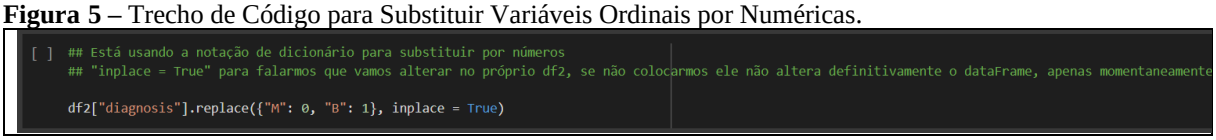
2.2.1 Pré-processamento dos Dados

Essa etapa é responsável pela organização dos dados tratados anteriormente para a execução dos algoritmos de Machine Learning, na qual serão feitas as transformações necessárias e a separação dos atributos previsores e alvo, além da designação do percentual dos dados que serão destinados para treino e teste.

2.2.1.1 Transformando Variáveis Categóricas Nominais em Variáveis Categóricas Ordinais

Nessa etapa foram feitas as transformações das variáveis ordinais em numéricas, uma vez que os algoritmos de classificação trabalham somente com números. O trecho de código da figura 5 demonstra esse processo de transformação para a variável *diagnosis*, em que a letra *"*M*"* representa o diagnóstico do câncer como maligno e a letra *"*B*"* como benigno, portanto, essas classificações foram substituídas pelo valor 0 e 1, respectivamente.

Figura 5 – Trecho de Código para Substituir Variáveis Ordinais por Numéricas.

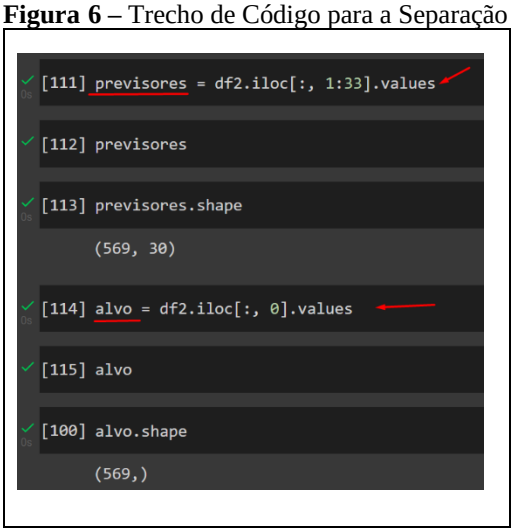


Fonte: Elaborado pelo autor

2.2.1.2 Separação dos Atributos Previsores e Alvo

O trecho de código da figura 5 demonstra a separação dos atributos, no qual o atributo alvo seria somente o *diagnosis* e o restante deles, os previsores.

Figura 6 – Trecho de Código para a Separação dos Atributos da Base de Dados



Fonte: Elaborado pelo autor

2.2.1.3 Análise das Escalas dos Atributos (Escalonamento)

Etapa que visa atribuir proporções de forma semelhante entre os atributos da base trabalhada. O trecho de código da figura 6 demonstra, a princípio, a análise das escalas dos atributos trabalhados, e, percebe-se que há certa diferença entre os valores mínimos e máximos de cada um deles, o que pode gerar confusão na execução dos algoritmos de Machine Learning. Nesse caso, tem-se duas soluções: a padronização, que utiliza a média e o desvio padrão como referência, ou a normalização, que utiliza os valores mínimo e máximo como referência. Nessa situação foi utilizada a padronização, e, para isso foi importado da biblioteca de pré-

processamento de dados *sklearn* o *StandardScaler* que será a escala de padronização para realizar as transformações necessárias na base de dados.

Figura 7 – Trecho de Código com o Escalonamento dos Atributos Previsores

```
[88] ## É uma biblioteca de pré processamento
     ## StandardScaler é uma escala de padronização
     ## Biblioteca sklearn

     from sklearn.preprocessing import StandardScaler

[89] previsores_esc = StandardScaler().fit_transform(previsores)

     previsores_esc

     array([[ 1.09706398e+00, -2.07333501e+00,  1.26993369e+00, ...,
              2.34012296e+00,  2.75062224e+00,  1.93701461e+00],
            [ 1.82982061e+00, -3.53632408e-01,  1.68595471e+00, ...,
              1.08564604e+00, -2.43889668e-01,  2.81189987e-01],
            [ 1.57988811e+00,  4.56186952e-01,  1.56650313e+00, ...,
              1.98621511e+00,  1.15225500e+00,  2.01391209e-01],
            ...,
            [ 7.02284249e-01,  2.04557380e+00,  6.72675785e-01, ...,
              3.87310029e-01, -1.10454895e+00, -3.18409158e-01],
            [ 1.83834103e+00,  2.33645719e+00,  1.98252415e+00, ...,
              2.33380318e+00,  1.91908301e+00,  2.21963528e+00],
            [-1.80840125e+00,  1.22179204e+00, -1.81438851e+00, ...,
             -1.09630758e-15, -4.81382136e-02, -7.51206693e-01]])
```

Fonte: Elaborado pelo autor.

2.2.1.4 Separação dos Dados de Treino e Teste

Nessa etapa foi feita a separação dos dados de treino e teste, tanto para os atributos alvo, quanto para os atributos previsores, na qual 30% desses dados foram reservados para teste e 70% para treino, como demonstrado no trecho de código da figura 7. Vale ressaltar que, a princípio, está sendo trabalhado com os previsores escalonados.

Figura 8 – Trecho de Código com a Separação dos Dados de Treino e Teste

```
[104] ## Os "x_treino" e "x_teste" seriam os dados de treinamento e teste dos atributos PREVISORES
      ## Os "y_treino" e "y_teste" seriam os dados de treinamento e teste dos atributos ALVO


      ## Então na verdade ele vai fazer a combinação de "x_treino" e "y_treino", ele vai estar treinando utilizando estes valores
      ## Depois vamos testar a qualidade do algoritmo com os de teste, quanto acerto e etc, "x_teste" e "y_teste"
      ## "alvo" são nossos atributos alvo para a aprendizagem supervisionada que vamos utilizar!

      ## test_size, aqui no caso vai ser 0.3, pois 30% dos dados serão separados para teste! E 70% automaticamente vai ficar para treino!
      x_treino, x_teste, y_treino, y_teste = train_test_split(previsores_esc, alvo, test_size = 0.3, random_state = 0)
```

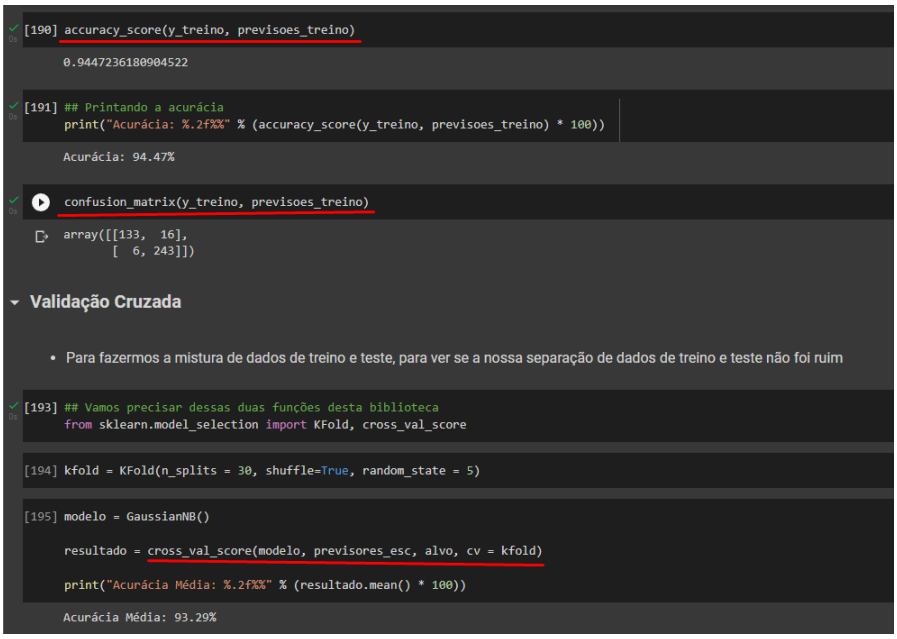
Fonte: Elaborado pelo autor.

2.2.2 Aplicação dos Algoritmos de Machine Learning

Etapa responsável pela aplicação dos algoritmos de Machine Learning SVM (Máquina de Vetores de Suporte) e Naive Bayes para previsão do desenvolvimento de câncer de mama em pacientes, na qual será feita a aplicação destes algoritmos, tal como a análise dos resultados obtidos em ambos, tanto com os previsores padrões, quanto com os previsores escalonados.

Para realizar a verificação da taxa de assertividade foram utilizadas como métricas a matriz de confusão, na qual a diagonal principal contabiliza os acertos e os demais valores contabilizam os erros, e a acurácia, a qual é a taxa de assertividade do algoritmo. Além disso, foi utilizada a validação cruzada para o embaralhamento dos dados de treino e teste a fim de verificar se a separação dos dados foi satisfatória. O trecho de código da figura 8 demonstra a utilização dessas métricas.

Figura 9 – Trecho de Código com a utilização de Métricas para Verificação da Assertividade

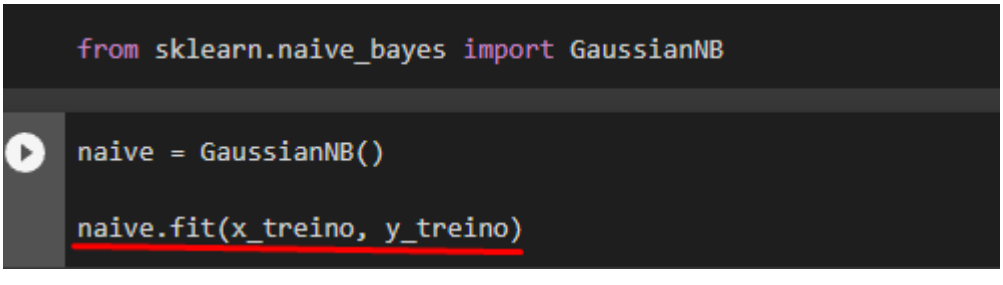


Fonte: Elaborado pelo autor.

2.2.2.1 Aplicação do Algoritmo Naive Bayes

A princípio, para o uso desse algoritmo, foi utilizada a biblioteca sklearn, da qual foi importado o algoritmo de classificação e feito o treinamento dos dados, para que o algoritmo aprendesse com eles e gerasse um resultado mais preciso, como demonstrado na figura 9.

Figura 10 – Trecho de Código com o Treinamento do Algoritmo Naive Bayes



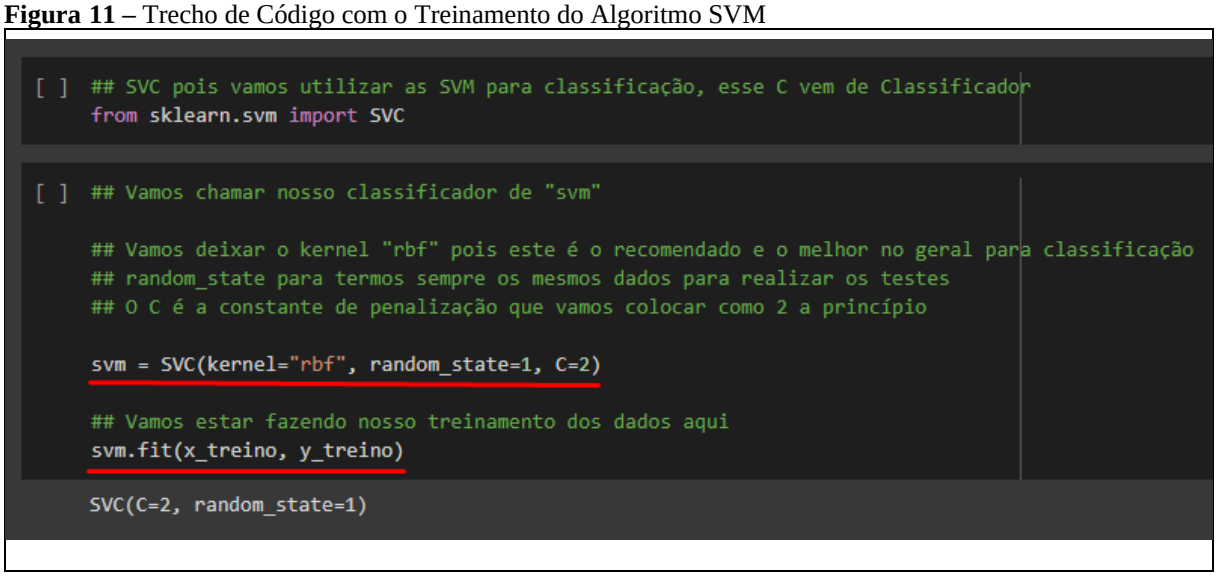
Fonte: Elaborado pelo autor.

Como resultado da aplicação do algoritmo Naive Bayes foi obtido uma acurácia média, com a validação cruzadas, de 93.29% com os previsores escalonados e 93.82% para os comuns, sendo que a análise dos dados de teste obteve resultado inferior aos dados de treino com os previsores comuns, de 92.40% nos de testes, em comparação aos 94.47% dos de treino. A mesma situação ocorreu com os previsores escalonados, cujos dados de teste obtiveram uma acurácia de 91.23% e os de treino 94.22%. Portanto, tais resultados não são discrepantes o suficiente para caracterizar algum problema de *underfitting* ou *overfitting* no tratamento dos dados utilizados. Sendo assim, ele pode ser caracterizado como um algoritmo balanceado, uma vez que se adaptou bem e obteve um bom desempenho em todas as situações testadas.

2.2.2.2 Aplicação do Algoritmo SVM (Máquina de Vetores de Suporte)

A princípio, para o uso deste algoritmo foi utilizada também a biblioteca sklearn, da qual foi importado o algoritmo de classificação em questão, no caso o SVM, esse que foi utilizado com o kernel *"rbf"* que é mais recomendado para a grande parte dos problemas relacionados à classificação, e a constante de penalização igual a dois, a qual obteve melhores resultados nos testes realizados. A figura 10 demonstra a instanciação do algoritmo e o treinamento dele.

Figura 11 – Trecho de Código com o Treinamento do Algoritmo SVM



```
[ ]  ## SVC pois vamos utilizar as SVM para classificação, esse C vem de Classificador
     from sklearn.svm import SVC

[ ]  ## Vamos chamar nosso classificador de "svm"

     ## Vamos deixar o kernel "rbf" pois este é o recomendado e o melhor no geral para classificação
     ## random_state para termos sempre os mesmos dados para realizar os testes
     ## O C é a constante de penalização que vamos colocar como 2 a princípio

     svm = SVC(kernel="rbf", random_state=1, C=2)

     ## Vamos estar fazendo nosso treinamento dos dados aqui
     svm.fit(x_treino, y_treino)

     SVC(C=2, random_state=1)
```

Fonte: Elaborado pelo autor.

Como resultado da aplicação do algoritmo SVM foi obtida uma acurácia média, com a validação cruzada, de 97.88% com os previsores escalonados e de 91.72% para os previsores normais, sendo que, para os previsores escalonados nos dados de treino tivemos uma taxa de assertividade de 97.66% com os dados de teste, inferior a assertividade de 98.49% dos dados de treino. Nos previsores normais foi obtido uma assertividade de 94.74% com os dados de teste, superior a assertividade de 90.95% dos dados de treino. Portanto, tais resultados também não são discrepantes o suficiente para caracterizar algum problema de *underfitting* ou *overfitting* no tratamento dos dados utilizados, sendo assim, ele pode ser caracterizado como

um algoritmo balanceado, tendo em vista que se adaptou bem e obteve um bom desempenho em todas as situações testadas.

2.3 Materiais e Métodos

Nesse capítulo serão demonstrados todas as tecnologias e os materiais que foram utilizados para a elaboração de ambos os algoritmos de Machine Learning, tal como as bibliotecas, base de dados, ambiente de programação, linguagem e outros.

- **Python:** Linguagem de programação utilizada para a elaboração dos algoritmos por apresentar inúmeros recursos relacionados à aprendizagem de máquina, sendo de fácil acesso e utilização.

- **Biblioteca Pandas:** Biblioteca de código aberto disponível para a linguagem Python, a qual é extremamente eficaz para a manipulação e tratamento de dados, e que foi utilizada constantemente ao decorrer do desenvolvimento dos algoritmos.

- **Biblioteca NumPy:** Biblioteca matemática da linguagem Python, a qual fornece um vasto número de funções e operações para facilitar a execução de cálculos numéricos.

- **Biblioteca Plotly e Seaborn:** Bibliotecas gráficas eficientes utilizadas para a elaboração de gráficos com os registros disponíveis na base de dados usada.

- **Biblioteca Scikit-Learn:** Uma das bibliotecas mais conhecidas para a elaboração de algoritmos de Machine Learning, de código aberto e apresenta vários recursos, como a disponibilização de algoritmos, funções métricas para as análises dos resultados e entre outros, sendo utilizada para problemas de aprendizagem supervisionada e não supervisionada.

- **Biblioteca Pickle:** Utilizada nos algoritmos para salvar os atributos previsores e alvo em arquivos separados e para serem lidos posteriormente. Essa biblioteca fica responsável por traduzir o objeto numa *string* para armazenamento e também o inverso para utilização no código elaborado.

- **Google Colaboratory:** Ambiente de programação acessado pelo navegador utilizado para a elaboração de algoritmos em Pyhton, principalmente para a área de ciência de dados e elaboração de algoritmos de Machine Learning, visto que já apresenta todos os recursos necessários para a elaboração deles, como as bibliotecas anteriormente citadas. Além disso, essa ferramenta não utiliza o processamento local da máquina do usuário, mas sim de servidores da Google, o qual disponibiliza recursos de hardware de forma gratuita para a elaboração dos códigos.

- **Base de Dados:** A base utilizada contém registros de pacientes que tiveram diagnóstico positivo ou negativo para o câncer de mama, apresentando assim

diversos atributos relacionados às características do possível tumor. A base em questão foi extraída do site Kaggle, que possui inúmeros dados das mais diversas áreas para utilização em trabalhos relacionados à ciência de dados.

2.4 Resultados e discussões

Ao analisar os resultados obtidos pelos algoritmos de Machine Learning aplicados, foi possível concluir que o SVM (Máquina de Vetores de Suporte) obteve um resultado superior quando trabalhado com previsores escalonados, tendo a maior taxa de assertividade de todos os testes realizados, de 97,88%. Já com os previsores comuns, o melhor resultado é do apresentado pelo algoritmo Naive Bayes, que obteve um resultado superior, de 93,82% de assertividade. Portanto, ao analisar os resultados obtidos, o mais recomendado para se utilizar seria o algoritmo SVM juntamente com os previsores escalonados. Todavia isso não descarta que todos os outros testes tiveram resultados igualmente bons, pois nenhum deles ficou abaixo dos noventa por cento de taxa de assertividade, o que caracteriza que a base utilizada foi bem tratada e que os algoritmos utilizados se adaptaram bem aos dados presentes nela.

3 CONSIDERAÇÕES FINAIS/CONCLUSÃO

O presente trabalho teve como objetivo demonstrar alguns dos conceitos do Machine Learning, juntamente com a aplicação dos conceitos estudados no tratamento e previsão de uma base de dados, nesse caso, uma base histórica com dados de pacientes que desenvolveram ou não câncer de mama. Para isso, foram utilizadas técnicas para o tratamento mais adequado possível da base estudada, retirando colunas incoerentes, tratando valores discrepantes e aplicando dois algoritmos para a previsão, sendo eles o Naive Bayes e o SVM (Máquina de Vetor de Suporte), esse último que obteve a melhor taxa de assertividade com os previsores escalonados. Portanto, é possível verificar que os algoritmos geram resultados muito satisfatórios se bem aplicados em uma base de dados devidamente tratada, sendo necessário um estudo anterior dela para a análise individual dos atributos em busca de dados incoerentes, sendo essa a parte mais complexa da elaboração de algoritmos de Machine Learning. Assim, os bons resultados são proporcionais à qualidade dos dados utilizados e como eles foram tratados.

REFERÊNCIAS

GALDINO, Luciano. **Machine Learning com Python**. Udemy, 2022. Disponível em: https://www.udemy.com/course/machine-learning-com-python. Acesso em: 13 de setembro de 2022.

CASALI, Rudiney. **Máquina de vetores de suporte: o que é?** Digital House, 2021. Disponível em: https://www.digitalhouse.com/br/blog/maquina-de-vetores-de-suporte. Acesso em: 13 de setembro de 2022.

BECKER, Lauro. **Algoritmo de Classificação Naive Bayes**. Orgânica Digital, 2019. Disponível em: https://www.organicadigital.com/blog/algoritmo-de-classificacao-naive-bayes/. Acesso em: 13 de setembro de 2022.

ALBON, Chris. **Support Vector Machines**. Chris Albon, 2017. Disponível em: https://chrisalbon.com/. Acesso em: 13 de setembro de 2022.

REMIGIO, Matheus. **Árvores de Decisão – Decision Trees**. Medium, 2020. Disponível em: https://medium.com/@msremigio/%C3%A1rvores-de-decis%C3%A3o-decision-trees-4cb6857671b3. Acesso em: 13 de setembro de 2022.

TIBCO SOFTWARE. **O que é regressão logística?** Tibco. Disponível em: https://www.tibco.com/pt-br/reference-center/what-is-logistic-regression. Acesso em: 13 de setembro de 2022.

TIBCO SOFTWARE. **O que é aprendizagem supervisionada?** Tibco. Disponível em: https://www.tibco.com/pt-br/reference-center/what-is-supervised-learning. Acesso em: 13 de setembro de 2022.

TIBCO SOFTWARE. **O que é aprendizagem regressão?** Tibco. Disponível em: https://www.tibco.com/pt-br/reference-center/what-is-regression-analysis. Acesso em: 13 de setembro de 2022.

SCIKIT-LEARN. **User Guide**. Scikit-Learn. Disponível em: https://scikit-learn.org/stable/user_guide.html. Acesso em: 13 de setembro de 2022.

AGRAWAL, Ashutosh. **Understanding Pyhton pickling and how to use securely**. Synopsys, 2014. Disponível em: https://www.synopsys.com/blogs/software-security/python-pickling/#:~:text=Pickle%20in%20Python%20is%20primarily,transport%20data%20over%20the%20network. Acesso em: 13 de setembro de 2022.

PANDAS. **Pandas documentation**. Pandas, 2022. Disponível em: https://pandas.pydata.org/docs/. Acesso em: 13 de setembro de 2022.

MULINARI, Bruna. **O que é o NumPy?** Harve. Disponível em: https://harve.com.br/blog/programacao-python-blog/numpy-python-o-que-e-vantagens-e-tutorial-inicial/. Acesso em: 13 de setembro de 2022.

SEABORN. **Seaborn: statistical data visualization**. Seaborn. Disponível em: https://seaborn.pydata.org/. Acesso em: 13 de setembro de 2022.

PLOTLY. **Plotly Open Source Graphing Library for Python**. Plotly. Disponível em: https://plotly.com/python/. Acesso em: 13 de setembro de 2022.

UCI MACHINE LEARNING. **Breast Cancer Wisconsin (Diagnostic) Data Set**. Kaggle, 2016. Disponível em: https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data. Acesso em: 13 de setembro de 2022.