Análise comparativa de técnicas de classificação de dados KNN e SVM para previsão de ataques cardiacos

1st Jackson Willian Silva Agostinho
Instituto Federal do Espírito Santo
Serra-ES, Brasil
jacksonwillianjbv@gmail.com

2nd Jardielma Queiroz De Lima
Instituto Federal do Espírito Santo
Serra-ES, Brasil
jardielmaqueiroz@hotmail.com

***Resumo*—Este trabalho apresenta um estudo comparativo do desempenho de algoritmos de classificação supervisionada o K-Nearest Neighbor (KNN) e o Support Vector Machine (SVM) na previsão de ataques cardíacos. A análise foi feita em uma base de dados com 303 registros de pacientes previamente atendidos na emergência de um hospital. Para análise utilizou-se técnicas de validação cruzada, métricas de acurácia, precisão, revocação, análise de variância (ANOVA) e o teste F. O estudo e a utilização dos métodos contribuiram para a solidificação da ideia de que não existe um método melhor ou pior e sim o método mais adequado para determinado padrão de dados.**

***Palavras-chave*—Algoritmos de Classificação, KNN, SVM, Mineração de Dados, Previsão de Ataque Cardíaco**

I. Introdução

A evolução tecnológica trouxe mudanças na forma de obter informação, uma vez que permite captar, armazenar e analisar grandes volumes de dados para gerar informação capaz de auxiliar na tomada de decisão. Uma forma de obter informações escondidas em grandes volumes de dados é através da Mineração de Dados, ela é formada por ferramentas e técnicas que através do uso de algoritmos são capazes de explorar um conjunto de dados, extraindo ou evidenciando padrões.

Um hospital deseja prever ataques cardíacos nos pacientes que dão entrada no seu setor de emergência, por esse motivo realizou-se o estudo de algoritmos de Mineração de Dados do tipo classificação supervisionada: o K-Nearest Neighbor (KNN) e o Support Vector Machine (SVM). O objetivo deste trabalho é comparar o desempenho de classificação dos algoritmos em uma base de dados com 303 registros de pacientes previamente diagnosticados. A verificação consiste em utilizar técnicas de validação cruzada, métricas de acurácia, precisão, revocação, análise de variância (ANOVA) e o teste F.

II. Referencial teórico

O algoritmo de Aprendizagem de Máquina (*Machine Learning*) do tipo supervisionado utiliza registros históricos rótulados para "aprender" a classificar novos dados desconhecidos. O algoritmo deve encontrar uma função de predição que melhor se adeque à classificação daqueles tipos de registros e isso é verificado por meio de treino/teste com os registros que já possui o rótulo da classe a qual pertence [1].

A. K-Nearest Neighbor (KNN)

KNN é um algoritmo de aprendizagem supervisionada que classifica um novo registro através do cálculo de distância com os K-vizinhos (quantidade de registros) mais próximos que estão armazenados e classificados [2]. A distância entre os pontos de dados de registros podem ser obtidas usando o cáculo de distância Euclidiana, Manhattan ou Markowski, então, o algoritmo é definido para usar um dos cálculos para encontrar as distâncias entre o novo registro e os registros já classificados, em seguida, o algoritmo seleciona os K-vizinho que tem a menor distância e realiza a contagem das classes dessa amostra, a classe que aparecer em maior frequência será considerada para classificar o novo registro [3].

Um exemplo de classificação usando o KNN pode ser visto na Figura 1, o registro representado pela bolinha cinza (não classificado) será classificado como classe Laranja, porque a classe Laranja prevalece em maior quantidade tanto para a amostra de K=3 quanto para a amostra K=5. O valor de K pode ser definido para outros valores, mas percebe-se pelo exemplo visto na imagem que o ideal é definir o valor de K como ímpar para evitar que a contagem das classes resultem em proporções iguais na amostra.

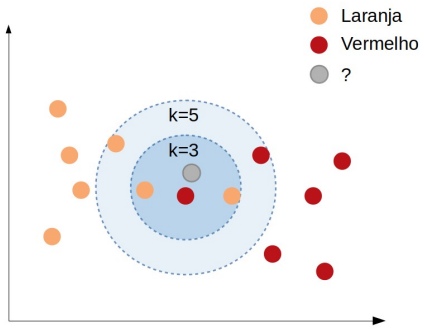

Figura 1. Exemplo de classificação KNN.
Fonte: O autor (2020)

B. *Support Vector Machine (SVM)*

O SVM é um algoritmo de aprendizagem supervisionada que classifica os registros com base na segregação das classes estabelecida pelo hiperplano ("fronteira de decisão"), a forma do hiperplano dependerá da dimensão em que os registros se encontram, deste modo, ele pode ser um ponto, uma reta ou um plano [4]. A Figura 2 apresenta um exemplo de segregação de dados linearmente separáveis, nessa imagem há três retas tracejada e cada uma delas representa três posições possíveis para o hiperplano, contudo apenas um deve ser escolhido, é perceptível que o posicionamento do hiperplano altera a classificação de um novo registro (bolinha cinza), porque o novo registro pertencerá a classe Laranja se estiver do lado esquerdo do hiperplano, mas se o novo registro estiver do lado direito será classificado como classe Vermelho.

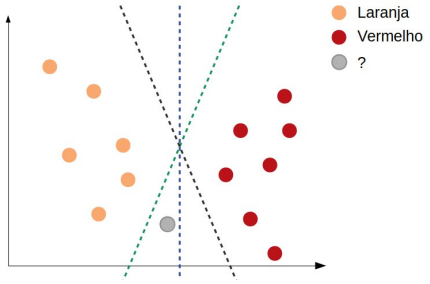

Figura 2. Segregação de dados linearmente separáveis.
Fonte: O autor (2020)

Para encontrar o melhor posicionamento do hiperplano deve-se considerar a margem do mesmo, a margem é a distância da lacuna entre o hiperplano e os vetores de suporte, sendo os vetores de suporte os ponto de dados de cada classe que estão mais perto do hiperplano [5]. Uma boa margem é encontrada quando a distância dos vetores de suporte até a fronteira de decisão tem valores proporcionais para as classes segregadas [6].

Existe um parâmentro denominado gama que define a relevância dos pontos de dados mais distantes no cálculo realizado para encontrar o hiperplano, para gama com valor mais alto somente os pontos de dados próximos são considerados e para valor mais baixo os pontos mais distantes também são incluídos no cálculo [6].

Nas situações em que há muitos pontos de dados próximos ao hiperplano é recomendado ajustar o parâmentro de regularização, tal que, para valores alto a tendência é buscar uma margem menor que proporcione uma classificação mais correta, e para valores muito baixo o algoritmo tende a encontrar uma margem maior mesmo que, consequentemente, proporcione alguns erros de classificação [6].

Quando os pontos de dados estão organizados como não sendo linearmente separáveis, como pode ser visto na Figura 3, item I, não há reta que possibilite separar as duas classes e, então, para resolver isso o algoritmo realiza uma transformação que consiste em mudar a dimensão atual para uma dimensão superior, essa técnina é chamada de Truque do Kernel, porque é a responsabilidade da função de Kernel realizar a transformação, e ela pode ser do tipo linear, polinomial e radial (RBF - Radial Basis Function), sendo polinomial e RBF úteis para hiperplano não linear [5]. A Figura 3, item II, mostra um exemplo de transformação e segregação para resolver o problema de pontos de dados linearmente não separáveis, de modo mais detalhado, a Figura 3 mostra que o algoritmo transformou o plano R^2 (item I) em um espaço R^3 (item II) para que fosse possível criar um plano para separar as classes.

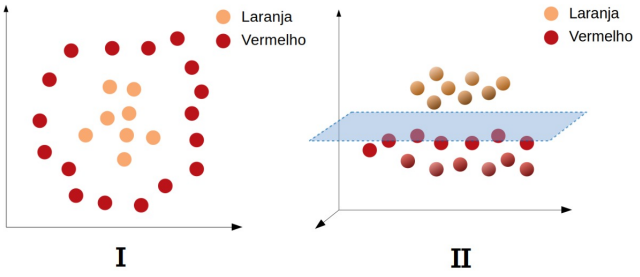

Figura 3. Transformação e segregação.
Fonte: O autor (2020)

III. Metodologia

A base de dados utilizada foi obtida de um hospital que reuniu registros a respeito de 303 pacientes atendidos na emergência e previamente diagnosticados, cada registro possui treze atributos que indicam características do paciente (idade, colesterol, pressão sanguínea, e etc.) e também o resultado do diagnóstico. Nessa base de dados há 165 registros de pacientes que sofreram ataque cardíaco e 138 registros de pacientes que não sofreram ataque cardíaco.

Um resumo estatístico da base de dados pode ser visto na Tabela I, onde as treze primeiras colunas com dados numéricos representam as características e a última o diagnóstico (indicador a ser previsto). Para as linhas da tabela: *count* indica o total de registros, *mean* indica o valor da média do atributo, *std* indica o desvio padrão, *min* indica o valor mínimo e o *max* o valor máximo.

Tabela I
Resumo estatístico da base de dados.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0	303.0
mean	54.37	0.68	0.97	131.62	246.26	0.15	0.53	149.65	0.33	1.04	1.4	0.73	2.31	0.54
std	9.08	0.47	1.03	17.54	51.83	0.36	0.53	22.91	0.47	1.16	0.62	1.02	0.61	0.5
min	29.0	0.0	0.0	94.0	126.0	0.0	0.0	71.0	0.0	0.0	0.0	0.0	0.0	0.0
max	77.0	1.0	3.0	200.0	564.0	1.0	2.0	202.0	1.0	6.2	2.0	4.0	3.0	1.0

Fonte: O autor (2020)

O cenário criado para a realização dos testes e verificação de desempenho dos algoritmos de classificação envolve, nesta ordem, a preparação dos dados de treino e teste, definição das configurações dos algoritmos, a criação das matrizes de confusão e o cálculo das métricas de desempenho.

A. Preparação dos dados de treino e teste

Na base de dados aplicou-se uma transformação sobre os valores de modo a resultar em uma média próxima de 0 e um desvio padrão próximo 1. Essa transformação é recomendada porque os dados podem ter escalas numéricas bastante diferentes e isso pode resultar em um péssimo desempenho dos algoritmos de classificação. Além disso, a base de dados foi divida em amostras de dados de treino/teste obtidas pela técnica de validação cruzada K-Fold, com K=5.

O K-Fold é utilizado para realizar a divisão da base de dados em K partes e executar K iterações, onde para cada iteração uma parte diferente é usada como teste e as outras partes restantes são usadas como treino, deste modo, a validação cruzada K-Fold permite verificar o desempenho dos algoritmos utilizando todos os dados da base [7]. A Figura 4 apresenta um exemplo da divisão do K-Fold.

No teste utilizou-se a implementação do algoritmo K-Fold que fornece opções de embaralhamento dos dados e que gera divisões de um modo que as classes sejam bem distribuídas em cada parte.

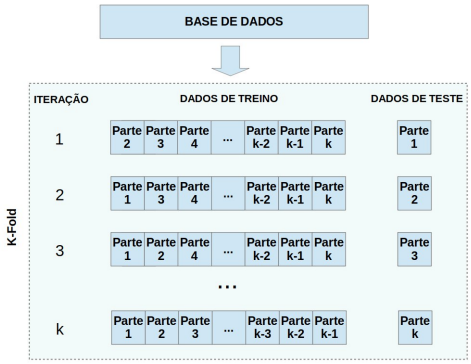

Figura 4. Divisão treino/teste do K-Fold.

Fonte: O autor (2020)

B. Configurações dos algoritmos

Para o algoritmo KNN foi definido a quantidade de vizinhos para k=3, k=5, k=7 e k=9 e para o SVM foi definido o tipo de função de kernel como kernel=Linear e kernel=RBF, ou seja, não considerou os outros parâmetros do SVM como gamma e regularização.

Após definir as configurações dos algoritmos foi realizada a etapa de treino e teste com as amostras do K-Fold (fold=5), assim, criou-se 5 amostras diferentes de treino e teste, e essas mesmas amostras foram utilizadas por todos os algoritmos.

C. Criação das matrizes de confusão

A matriz de confusão é gerada a partir dos resultados das informações de teste e por causa do K-Fold haverá 5 iterações de treino/teste e, consequentemente, haverá também 5 matrizes de confusão parciais, contudo será criado uma matriz de confusão acumulada resultante das somas das 5 matrizes de confusão.

A matriz de confusão permite ter uma visão geral acerca da capacidade preditiva do algoritmo classificador, ela permite identificar as classificações preditas corretas (diagonal principal da matriz) e as classificações preditas incorretas (diagonal secundária da matriz), mais informações acerca da matriz de confusão são apresentadas na Figura 5.

		PREVISTO	
		Negativo	**Positivo**
REAL	**Negativo**	Casos negativos que o sistema previu como negativos (Verdadeiro Negativo - VN)	Casos negativos que o sistema previu como positivo (Falso Positivo - FP)
	Positivo	Casos positivos que o sistema previu como negativo (Falso Negativo - FN)	Casos positivos que o sistema previu como positivo (Verdadeiro Positivo - VP)

Figura 5. Matriz de confusão.

Fonte: O autor (2020)

D. Cálculos das métricas

Através da matriz de confusão é possível encontrar métricas que apresentam o desempenho do algoritmo [8], são elas:

- Acurácia (*Accuracy*): representa o total de previsões corretas (VN + VP) divida pelo total de previsões corretas (VN + VP) e incorretas (FP + FN). A fórmula da acurária é dada por:

$$Acurácia = \frac{VN+VP}{VN+VP+FP+FN}$$

- Precisão (*Precision*): representa o número de predição do tipo verdadeiro positivo (VP) divido pela soma de todos os positivos previstos, ou seja, predições verdadeiro positivo (VP) e falso positivo (FP). A fórmula da precisão é dada por:

$$Precisão = \frac{VP}{VP+FP}$$

- Revocação (*Recall*): representa o número de predição do tipo verdadeiro positivo (VP) divido pela soma de todos os positivos reais, ou seja, o verdadeiro positivo (VP) e o falso negativo (FN). A fórmula da revocação é dada por:

$$Revocação = \frac{VP}{VP+FN}$$

No estudo realizado a métrica de acurácia é obtida pela média das acurácias calculadas a partir de cada matriz parcial, e as métricas de precisão e de revocação são obtidas usando a matriz de confusão acumulada.

IV. Resultados

Os resultados apresentados pelo processo de classificação considerou o modelo básico dos métodos K-Nearest Neighbor (KNN) e o Support Vector Machine (SVM) analisados, buscando suas variações de melhor resultados. As etapas de treino e teste efetuadas nos algoritmos foram realizadas a partir de um conjunto de 303 registros, onde 242 registros foram

utilizados para treino e 61 registros utilizados para testes, fornecidos em diferentes amostras a cada uma das 5 iterações do K-Fold, além disso, as divisões do dados de treino e teste são as mesmas para todos algoritmos a fim de garantir que o cenário seja o mesmo e, deste modo, permitir uma comparação de desempenho de classificação mais verídica. A Figura 6 apresenta exemplos de possíveis entradas para o propósito deste trabalho.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Figura 6. Amostra de dados

Fonte: O autor (2020)

A Figura em questão reforça o fato dos métodos terem sido submetidos aos mais diversos tipos de registros. Garantiu-se também que todos fossem testados com os mesmos conjuntos de dados, permitindo desta forma que as análises fossem realizadas segundo dados pareados. Cada algoritmo depois de treinado foi exposto a uma série de testes, onde foi observado as métricas de acurácia, precisão e revocação, sendo os resultados de cada método apresentados no tópico a seguir.

A classificação feita pelo método (KNN) K-Vizinhos mais próximos, apresenta evolução inversa ao aumento do valor de K, como pode ser visto na tabela II. No entanto, o algoritmo proposto apresenta taxas de acerto de até 80.3%, acurácia de 81.2% e revocação de 86.7%, que comprovam a eficiência do classificador. Após realizados os testes, destacou-se o melhor resultado para o valor de k=3, apresentando valor superior para as três métricas de desempenho.

O classificador SVM foi submetido a simulações, utilizando kernel linear e kernel RBF como configurações básicas. Para a simulação utilizando kernel linear o algorítimo apresenta taxas de acerto de 81.3%, acurácia de 84.2% e revocação de 92.1%, e para simulação utilizando kernel RBF o algoritmo apresenta taxas de acerto de 79.5%, acurácia de 81.5% e revocação de 89.1%. Após realizados os testes, detectou-se o melhor resultado ao se utilizar o kernel linear, sendo superior nas três métricas de desempenho, como pode ser visto na tabela II.

Todos os métodos foram submetidos a análises comparativas simples, como comparação direta entre os resultados, como pode ser observado no gráfico apresentado na Figura 7.

Após uma comparação direta nota-se pouca diferença entre as taxas de acerto dos métodos analisados, sendo os melhores apresentados na seguinte ordem, KNN para k = 3 e SVM utilizando kernel linear. No entanto, para que se possa afirmar que um método apesar de apresentar maior taxa de acerto é ou não mais eficaz do que outro para o teste realizado, é necessário aplicar outros métodos de comparação estatística, sendo neste trabalho aplicado a análise de variância (ANOVA), juntamente com o teste F [10]. Pode-se observar na tabela III o pareamento dos dados obtidos durante os testes realizados em

Tabela II
COMPARAÇÃO DAS MÉTRICAS DE DESEMPENHO.

Algoritmo	Métricas			Matriz de Confusão
	Acurácia	Revocação	Precisão	
KNN(K=3)	0,812	0,867	0,803	0 1 <- classified as 103 35 0: não 22 143 1: sim
KNN(K=5)	0,798	0,873	0,783	0 1 <- classified as 98 40 0: não 21 144 1: sim
KNN(K=7)	0,805	0,891	0,782	0 1 <- classified as 97 41 0: não 18 147 1: sim
KNN(K=9)	0,808	0,891	0,786	0 1 <- classified as 98 40 0: não 18 147 1: sim
SVM (Kernel=Linear)	0,842	0,921	0,813	0 1 <- classified as 103 35 0: não 13 152 1: sim
SVM (Kernel=RBF)	0,815	0,891	0,795	0 1 <- classified as 100 38 0: não 18 147 1: sim

Fonte: O autor (2020)

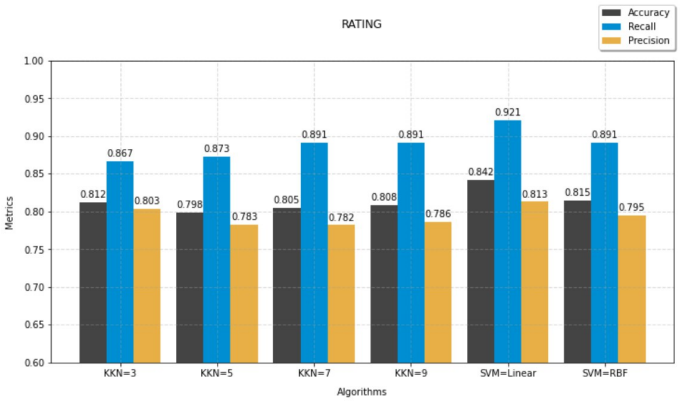

Figura 7. Comparação das métricas de desempenho.

Fonte: O autor (2020)

cada um dos métodos analisados. Observa-se também nesta tabela a variância e o desvio padrão entre os valores.

Através do desvio padrão é possível medir o quão dispersos estão os dados em relação à média. Como se pode observar na tabela II, os dados de precisão tendem a estar próximo da média, o que aumenta a confiabilidade nos resultados. Foi realizada também o cálculo da variância dos dados coletados, sendo os resultados apresentados na tabela III.

Tabela III
CÁLCULO DA VARIÂNCIA E DESVIO PADRÃO

Grupos	Contagem	Média	Variância	Desvio Padrão
KNN	4	0.783	0.000107	0.010344
SVM	2	0.804	0.000162	0.0127279

Fonte: O autor (2020)

O cálculo da variância é imprescindível para realização da análise de variância, por permite definir e quantificar o quanto uma média é estatisticamente diferente entre as médias

comparadas. A Figura 8 apresenta o resultado do teste de análise de variância, juntamente com a análise do teste F.

Fonte	Grau de Liberdade	Soma dos quadrados	Quadro médio	F-estatística	P-valor
Entre os grupos	1	0.0004	0.0004	**3.0278**	**0.1568**
Dentro dos grupos	4	0.0005	0.0001		
Total	5	0.0008			

Figura 8. Análise de Variância

Fonte: O autor (2020)

O resultado da análise de variância, juntamente com o teste F demonstram que não existe diferença estatística entre as médias dos métodos. Dado que o P-valor é maior que o nível de significância (0.05), a hipótese nula é aceita. Logo, as médias dos algoritimos são consideradas iguais. Em outras palavras, a diferença entre as médias não é grande o suficiente para ser estatisticamente significativa. P-valor é igual a 0.1568. Isso significa que se rejeitássemos a hipótese nula, a chance de erro do tipo (rejeitar uma hipóte correta) seria muito alto, sendo de aproximadamente 15.68%. A Figura 9 apresenta os valores análisados e a Figura 10 apresenta diferença entre a média e o desvio padrão dos mesmos.

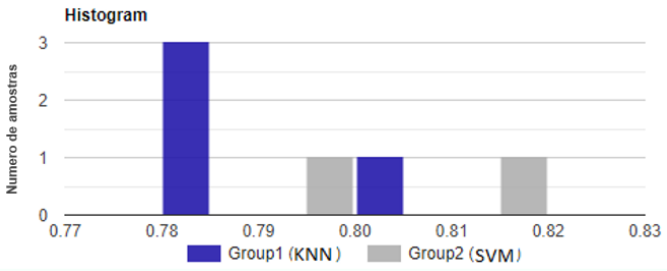

Figura 9. Dados utilizados na ánlise da variância

Fonte: O autor (2020)

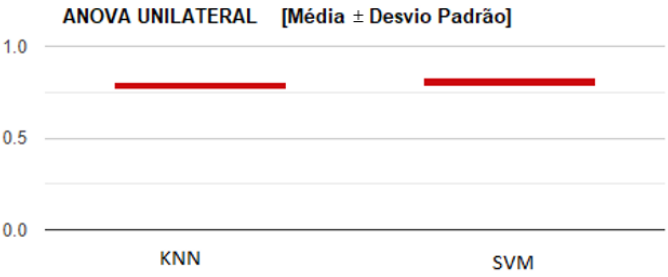

Figura 10. Análise da ANOVA unilateria - média e desvio pradrão

Fonte: O autor (2020)

Conclusão

Ao final dos testes foi possível concluir que, os parâmetros obtidos através das operações estatísticas realizadas (ANOVA e Teste F) indicam não haver diferenças no índice de acertos/aptidão entre os métodos de KNN e SVM, nota-se que existe, de acordo com a análise simples e o teste F, diferenças estatísticas de aproximadamente 3% entre os dois métodos, ainda que, as diferenças nas taxas de acerto sejam minimas. Conclui-se com o estudo que, aprimoramentos nas técnicas podem levar a resultados mais eficientes, possibilitando inclusive alterações no resultado final do comparativo. Contudo, como neste trabalho foram utilizados os algoritmos em suas configurações básicas, entende-se que, para solução do problema em questão, não houve um destaque para um dos métodos, com isso, concluimos que ambos apresentaram dados satisfatórios a ponto de definirmos que ambos possuem a média de precisão igual. Por fim, o estudo e a utilização dos métodos contribuiram para solidificação da ideia de que não existe um método melhor ou pior e sim um método mais adequado para determinado padrão de dados.

Referências

[1] D. Korbut. “Machine Learning Algorithms: Which One to Choose for Your Problem”. Medium. https://blog.statsbot.co/machine-learning-algorithms-183cc73197c (acessado em 05 Set. 2020)

[2] F. Santana. “Machine Learning na prática com o algoritmo KNN em Python”. Minerando Dados. https://minerandodados.com.br/machine-learning-na-pratica-knn-python (acessado em 06 Set. 2020)

[3] I. José. “KNN (K-Nearest Neighbors)”. Medium. https://medium.com/brasil-ai/knn-k-nearest-neighbors-1-e140c82e9c4e (acessado em 06 Set. 2020)

[4] J. Shubham. “Support Vector Machines (SVM)”. Medium. https://medium.com/coinmonks/support-vector-machines-svm-b2b433419d73 (acessado em 06 Set. 2020)

[5] A. Navlani. “Support Vector Machines with Scikit-learn”. DataCamp https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python (acessado em 06 Set. 2020)

[6] S. Patel. “Chapter 2: SVM (Support Vector Machine) - Theory”. Medium. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72 (acessado em 06 Set. 2020)

[7] S. Yıldırım. “How to train_test_split : KFold vs StratifiedKFold”. Medium. https://towardsdatascience.com/how-to-train-test-split-kfold-vs-stratifiedkfold-281767b93869 (acessado em 10 Set. 2020)

[8] Intellipaat. “Introduction to Confusion Matrix in Python Sklearn”. Intellipaat https://intellipaat.com/blog/confusion-matrix-python (acessado em 12 Set. 2020)

[9] M. Filho. “As Métricas Mais Populares para Avaliar Modelos de Machine Learning”. Mariofilho. https://www.mariofilho.com/as-metricas-mais-populares-para-avaliar-modelos-de-machine-learning (acessado em 12 Set. 2020)

[10] A. M. Souza. “Análise de variância anova”. UFSM. http://w3.ufsm.br/adriano/aulas/anova/T[12].anova.pdf (acessado em 12 Set. 2020)

[11] L. Martinez, A. Ferreira. Análise de Dados com SPSS. Escolar editora, 2007.