UNIVERSIDADE FEDERAL DE ALAGOAS
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

CARLOS ALBERTO FEITOSA DA SILVA

Aprendizado de máquina aplicado à predição de desempenho de alunos com base em dados socioeconômicos e acadêmicos

Maceió
2019

CARLOS ALBERTO FEITOSA DA SILVA

APRENDIZADO DE MÁQUINA APLICADO À PREDIÇÃO DE DESEMPENHO DE ALUNOS COM BASE EM DADOS SOCIOECONÔMICOS E ACADÊMICOS

Trabalho de Conclusão de Curso submetido ao Curso de Sistemas de Informação do Instituto de Computação da Universidade Federal de Alagoas como requisito parcial para a obtenção do Grau de Bacharel em Sistemas de Informação.

Orientador: Prof. LUCAS BENEVIDES VIANA DE AMORIM

Maceió

2019

CARLOS ALBERTO FEITOSA DA SILVA

APRENDIZADO DE MÁQUINA APLICADO À PREDIÇÃO DE DESEMPENHO DE ALUNOS COM BASE EM DADOS SOCIOECONÔMICOS E ACADÊMICOS

Este Trabalho de Conclusão de Curso (TCC) foi julgado adequado para obtenção do Título de Bacharel em Sistemas de Informação e aprovado em sua forma final pelo Instituto de Computação da Universidade Federal de Alagoas.
Maceió, 25 de setembro de 2019.

Banca Examinadora:

_____

Prof. PETRÚCIO ANTÔNIO MEDEIROS BARROS, Ms.
Coordenador do Curso de Sistemas de Informação

_____

Prof. LUCAS BENEVIDES VIANA DE AMORIM, Ms.
Orientador

_____

Prof. EVANDRO DE BARROS COSTA, Dr.
Universidade Federal de Alagoas

DEDICATÓRIA

Dedico este trabalho a minha esposa Ana Paula Souza Santos que foi a principal incentivadora dessa jornada, com seu amor, inteligência e paciência pudemos passar juntos por todas as dificuldades e vitórias dos últimos 12 anos de nossas vidas. Especialmente os últimos anos em que por inúmeras vezes a minha ausência sempre foi compreendida possibilitando a realização desse sonho.

Aos meus pais, José Martins da Silva Irmão e Antônia Honorato Feitosa da Silva e irmão Luiz Felipe Feitosa da Silva que doaram seu amor incondicional desde o momento que saí do segurança do lar familiar para ir em busca de melhores oportunidades. Todas as ferramentas, incentivos e suporte necessário me foram dados possibilitando o meu desenvolvimento e crescimento pessoal. Com isso em mãos pude acreditar que era capaz de realizar tudo que desejasse desde que tivesse honra, honestidade e dedicação.

AGRADECIMENTOS

Agradeço a Universidade Federal de Alagoas, seu corpo docente, técnicos, tutores e direção por possibilitar a realização do curso e promovendo educação de qualidade e gratuita. Em especial ao professor Ms. Lucas Benevides Viana que me orientou na elaboração e execução deste trabalho, com dedicação, cordialidade e confiança até o último momento.

Aos amigos mais próximos que fiz no período como bolsista do Núcleo de Tecnologia da Informação, onde aprendi muito. Todos acompanharam a minha jornada na universidade e sempre proferiram palavras de incentivo e acolhimento em especial Jonathas Aberto (um irmão que escolhi ter), Ially Cristina, Jonatas Gonzaga, Kleymerson Pereira, Marília Inês e Clayton Nilo.

A pessoas especiais do polo Maceió que sempre estavam dispostas a ajudar com seu carinho e afeto. Em especial a Ana Medeiros e Elielba Mendes

Aos amigos e companheiros do Instituto Federal de Alagoas que sempre incentivaram e compreenderam as necessidades e ausências necessárias para o desenvolvimento deste trabalho em especial Tâmara Bastos, Lidianne Lira, Jakelline Raposo e Esther Freitas.

Aos companheiros de turma por termos passados juntos todos os momentos de alegria e dificuldades desta jornada inesquecível.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A identificação precoce de alunos que podem ter um desempenho abaixo do esperado propicia a equipe de ensino de uma universidade pensar em estratégias para melhorar o desempenho dos discentes. Por isso o trabalho tem como objetivo prever o desempenho final dos alunos, distinguindo-os entre os que provavelmente alcançarão bom desempenho e os que terão desempenho regular, com base em seus dados socioeconômicos e registros escolares. O aprendizado de máquina dispõe de várias técnicas de classificação, neste trabalho foram selecionadas dez, que após a preparação da base, foram aplicadas e tiveram sua métricas apresentadas através de gráficos. Foi visto que as técnicas *Gaussian Classifier* e *Random Forest* tiveram os melhores resultados predizendo corretamente em cerca de 86% dos casos e assim mostrando-se mais adequadas para cumprir o objetivo. Foi verificado também que os atributos sociais e econômicos são extremamente relevantes para a predição de desempenho dos alunos. Por fim foi verificado que o desempenho acadêmico anterior dos alunos é irrelevante para a mesma tarefa.

Palavras-chaves: Aprendizado de Máquina; Classificação; Dados Educacionais.

ABSTRACT

Early identification of students underperform and enables a university's teaching staff to think of strategies for improving student performance. Therefore, the objective of the work is to predict students' final performance, distinguishing between those who are likely to achieve good performance and those who will perform regularly, based on their socioeconomic data and school records. Machine learning has several classification techniques, in this work ten were selected, which after the preparation of the base, were applied and had their metrics presented through graphics. It was seen that the Gaussian Classifier and Random Forest techniques had the best results, correctly predicting in about 86% of the cases and thus proving to be more adequate to meet the objective. It was also found that social and economic attributes are extremely relevant to predicting student performance. Finally, it was found that students' previous academic performance is irrelevant to the same task.
Keywords: Machine learning; Classification; Educational Data

CAPÍTULO 1 - INTRODUÇÃO

A área educacional é uma área estrategicamente importante para o desenvolvimento humano, econômico e social da população de um país. Um dos maiores desafios do sistema educacional atual é a identificação precoce de estudantes que, por motivos variados, necessitam de uma abordagem diferenciada ou a utilização de técnicas e conceitos específicos dos demais para que possam alcançar a êxito acadêmico em sua jornada.

O aprendizado de máquina é uma área da computação em franco desenvolvimento. Nela podemos utilizar computadores para reconhecer padrões, aprender e prever resultados através da análise dos dados.

O presente trabalho visa propor a união dessas áreas de maneira que o aprendizado de máquina possa prever o desempenho dos alunos e contribuir na identificação precoce de estudantes que possam ter um desempenho não satisfatório no decorrer do ensino superior.

1.1 MOTIVAÇÃO

A identificação precoce de alunos que podem ter um desempenho abaixo do esperado propicia que docentes, tutores, pedagogos e toda equipe envolvida com o ensino possa pensar em estratégias para melhorar o desempenho dos discentes. Com isso podemos inferir que alguns indicadores, tais como: taxa de evasão, taxa de conclusão do curso, taxa de retenção escolar, etc. apresentarão melhores resultados.

Além da identificação de um baixo desempenho pode-se encontrar a causa do resultado negativo e traçar estratégias que visam resolver ou pelo menos amenizar o problema. Por exemplo caso seja identificado que um grupo de alunos terá problemas de desempenho porque reside muito distante da universidade pode-se adotar uma abordagem diferentes, trabalhando alguns conteúdos à distância ou em caso mais severos trazer o aluno para a residência universitária.

1.2 TRABALHOS RELACIONADOS

Hussain et. al. (2018), coletaram dados de três faculdades de Assam, na Índia Tais dados consistem em informações socioeconômicas, demográficas e acadêmicas de trezentos estudantes com vinte e quatro atributos. Foram utilizados quatro algoritmos de classificação: J48, PART, *Random Forest* e *Naive Bayes*. A ferramenta de mineração de dados usada foi o WEKA. Os resultados mostraram que a técnica *Random Forest* supera os outros classificadores com base em erros de precisão de classificação.

Alloghani et al. (2018), usaram a técnica de *Random Forest* e mais 7 outros algoritmos para estabelecer os mais eficientes. O estudo utilizou dois conjuntos de dados diferentes. O primeiro banco consiste em mais de 100 medidas de atividade da equipe, além de resultados para 74 equipes de estudantes. As informações foram coletadas para mais de 383 alunos correspondentes a 18 sessões da turma. Os dados agrupados do *Team Activity Measure* (TAM) possuem 115 atributos e 2 rótulos de classe. O TAM é composto por 59 equipes locais e mais de 15 equipes globais. Rede Neural mostrou-se o modelo mais eficiente em relação ao primeiro conjunto de dados e a *Random Forest* mais eficiente em relação ao segundo conjunto de dados.

Trakunphutthirak et al. (2019), usaram uma fonte de dados de um arquivo de log da universidade para prever o desempenho acadêmico. Ele investiga as categorias de navegação e as atividades de acesso à Internet dos alunos com relação ao gerenciamento do tempo durante os estudos. Foram empregamos dois conjuntos de dados, como categorias de navegação na Web e tipos de atividades de acesso à Internet. Cada entrada contém 54 atributos como identificação, data e hora do acesso, endereços IP, nome do aplicativo, categoria e um período de tempo decorrido. Os autores usaram a validação cruzada de 10 *folds* para reduzir o viés de um conjunto de dados de teste. Eles compararam a precisão dos resultados entre *Decision Trees, Naïve Bayes, Logistic Regression* e *Neural Network* e *Random* Forest Verificaram que a última é mais adequada para esse tipo de conjunto de dados. Além disso, descobriram que os dados de suas atividades de acesso à Internet revelam resultados mais precisos do que os dados das categorias de navegação. Por fim, verificaram que a combinação de dois conjuntos de dados revela uma melhor imagem do uso da Internet pelos alunos e, assim, identifica os estudantes que estão academicamente sob risco de falha.

Fernandes et al. (2018) apresentam uma análise preditiva do desempenho acadêmico de alunos de escolas públicas do Distrito Federal do Brasil durante os períodos escolares de 2015 e 2016. Dois conjuntos de dados foram obtidos, o primeiro conjunto de dados contém variáveis obtidas antes do início do ano letivo e o segundo incluiu variáveis acadêmicas coletadas dois meses após o início do semestre. Modelos de classificação baseados no *Gradient Boosting Machine* (GBM) foram criados para prever os resultados acadêmicos do desempenho dos alunos no final do ano letivo para cada conjunto de dados. Os resultados indicaram que os atributos identificados antes do início do ano letivo (primeiro conjunto de dados) foram contribuintes relevantes para as taxas de reprovação. Particularmente, as variáveis 'vizinhança' (residência do aluno) e 'escola' foram os principais fatores que afetam a taxa de reprovação do aluno.

1.3 OBJETIVOS

O presente trabalho tem como objetivo prever o desempenho final dos alunos, distinguindo-os entre os que provavelmente alcançarão bom desempenho e os que terão desempenho regular, com base em seus dados socioeconômicos e registros escolares. Na base de dados o desempenho dos estudantes foi classificado como "*Best*", "*Very Good*", "*Good*", "*Pass*" e "*Fail*", porém não foi encontrada nenhuma ocorrência da situação "Fail". Então para esse trabalho o resultado foi dividido em duas categorias, são elas: "Bom Desempenho" que agrupa ocorrências de "*Best*" e "*Very Good*"; e "Desempenho Regular" que agrupa as ocorrências "*Good*" e "*Pass*".

O objetivo específico é identificar técnicas de aprendizado de máquina que podem ser utilizadas apresentando resultados consistentes e confiáveis na previsão do desempenho de estudantes no ensino superior.

1.4 METODOLOGIA

A técnica de aprendizado de máquina disponível para realizar a tarefa de prever o desempenho acadêmico dos alunos é a classificação, já que em última análise trata-se de uma tarefa de classificar os alunos entre as duas classes de desempenho supracitadas (bom desempenho ou desempenho regular).

O conjunto de dados utilizado neste trabalho está disponível no repositório UCI (UCI, 2019) onde está descrito como "*Student Academics Performance Data Set*" e é composto de 300 instâncias, que representam os estudantes e 22 atributos contendo informações como desempenho acadêmico anterior, localização onde mora, ocupação dos pais, renda familiar, frequência nas aulas, horas de estudos, etc. são dados são classificados como multivariados. Mais especificamente, os 22 atributos são:

- GE: armazena o gênero do estudante
- CST: armazena a casta do estudante
- ARR: verifica se aluno teve algum trabalho com falha em qualquer um dos semestres anteriores
- MS: estado civil
- LS: armazena se o aluno morar na cidade ou vila
- AS: categoria de admissão
- FO: ocupação do pai
- MO: ocupação da mãe
- SS: tipo de escola que estudante frequentou

- ME: idioma falado pelo aluno.
- TNP: classificação de desempenho alcançado pelo aluno em relação a Turma
- FMI: renda mensal familiar
- FS: tamanho da família
- NF: número de amigos
- TT: tempo de viagem da faculdade e sua casa
- FQ: qualificação educacional do pai do estudante.
- MQ: qualificação educacional da mãe do estudante.
- SH: horas destinadas ao estudo
- ATD: percentagem de frequência
- ESP: desempenho no exame final semestral

O conjunto de dados foi submetido a aplicação de 11 (onze) técnicas de aprendizado de máquina de classificação são elas:

- *AdaBoost*
- *Decision Tree Classifier*
- *Gaussian Process Classifier*
- *KNN*
- *Naive Bayes*
- *Neural Net*
- *QDA*
- *Random Forest*
- *SVM com kernel RBF e Linear*
- *XGBoost*

Para a avaliação do desempenho das técnicas foram verificadas com as seguintes métricas:

- *Accuracy*
- F1
- Taxa de Falso positivo
- Taxa de Falso Negativo
- Taxa de Verdadeiro positivo
- Taxa de Verdadeiro Negativo
- *Precision*
- *Recall*
- *Specificity*

- *Área da curva ROC*

Para tanto, foram utilizadas a linguagem de programação Python (versão 3.7.4) as bibliotecas *Sklearn* (versão 0.21.3), Pandas (versão 0.25.1) e *Numpy* (versão 1.17.1), *XGBoost* (versão 0.90). Os modelos foram executados em um computador com processador Intel Core I7 - 5500U 2.4GHz e 8GB de memória RAM, levando aproximadamente entre 0,266386s e 1,918982s segundos para executar.

1.5 ESTRUTURA DO TRABALHO

O trabalho está dividido em 5 (cinco) capítulos. No primeiro é introduzido o assunto contextualizando e apresentando o problema a ser tratado. No subtítulo 1.1 temos a motivação do trabalho, no 1.2 trabalhos relacionados são citados, 1.3 os objetivos são tratados e no 1.4 a proposta metodológica é discutida. No capítulo 2 são apresentados os conceitos de aprendizagem de máquina, preparação dos dados, é dada ênfase nas técnicas e algoritmos utilizados no trabalho e para finalizar o capítulo temos as técnicas de avaliação dos modelos. No capítulo 3 são apresentados os materiais e métodos utilizados. No capítulo 4 temos os resultados e suas discussões. Por fim, no capítulo 5 é apresentada a conclusão.

CAPÍTULO 2 – APRENDIZADO DE MÁQUINA

2.1 INTRODUÇÃO

Aprendizado de máquina ou *Machine Learning* é um campo da inteligência artificial cujo principal objetivo é fazer com que a máquina realize inferência através do aprendizado prévio com o mínimo de intervenção humana. Para isso é fornecido um conjunto de dados que após a preparação de seus atributos, que será discutida no tópico 2.2, é submetido a algoritmos de aprendizado e por fim seus resultados são submetidos a métricas para avaliar se a predição de resultados é ou não confiável.

Witten e Frank (2005) descreveram quatro conceitos caracterizando os vários algoritmos de aprendizado de máquina em diferentes tarefas, são eles: o aprendizado de classificação (*classification learning*), onde o esquema de aprendizagem é apresentado com um conjunto de exemplos classificados, dos quais espera-se que o algoritmo aprenda uma maneira de classificar exemplos não vistos; O aprendizado por associação (*association learning*) , que procura qualquer associação entre recursos, e não apenas aqueles que predizem um determinado valor de classe; o aprendizado por agrupamento (*clustering*), no qual exemplos semelhantes são agrupados de acordo com um critério estabelecido; e previsão numérica (*numeric prediction*), onde o resultado a ser previsto não é uma classe discreta, mas uma quantidade numérica.

Além das tarefas de aprendizado, podemos classificar o aprendizado segundo o seu método. Os métodos mais comuns são os de aprendizado supervisionado e não-supervisionado. Os métodos mais usados são: o aprendizado supervisionado, que envolve treinamento por meio de exemplos rotulados, como uma entrada na qual a saída desejada é conhecida (SAS, 2019). Neste caso, algoritmo de aprendizado recebe um conjunto de entradas junto com as saídas corretas correspondentes, e aprende ao comparar a saída real com as saídas preditas para encontrar erros; o aprendizado não-supervisionado é utilizado com dados que não possuem rótulos históricos. O algoritmo deve descobrir o que está sendo mostrado. O objetivo é explorar os dados e encontrar alguma estrutura dentro deles; aprendizado semi-supervisionado é utilizado para as mesmas aplicações que o aprendizado supervisionado. Mas este aqui manipula tanto dados rotulados quanto não-rotulados para treinamento; e o Aprendizado por reforço, onde o algoritmo descobre, através de testes do tipo 'tentativa e erro', quais ações rendem as maiores recompensas. Este tipo de aprendizado possui três componentes principais: o agente (o aprendiz ou tomador de decisão), o ambiente (tudo com que o agente interage) e ações (o que o agente pode fazer). O objetivo é que o agente escolha ações que maximizem a recompensa esperada em um período de tempo

determinado. O agente atingirá o objetivo muito mais rápido se seguir uma boa política. Então o foco do aprendizado por reforço é descobrir a melhor política.

2.2 PREPARAÇÃO DOS DADOS

A preparação dos dados é uma das etapas mais importantes do processo de aprendizagem de máquina. É nesse momento que muitos problemas futuros podem ser evitados e onde pode-se tornar a predição mais eficiente. Para isso, é necessário realizar algumas etapas como: seleção, limpeza e transformação de dados.

2.2.1 Limpeza de dados

A limpeza de dados é a etapa executada principalmente para remover possíveis inconsistências nos dados, os chamados ruídos e valores ausentes.

É comum em bases de dados alguns dados estarem ausentes, seja por problemas no recolhimento, medição ineficiente, e etc. Estas características devem ser tratadas antes do modelo de aprendizagem de máquina recebê-las, pois a ausência de dados pode afetar a eficiência e precisão do modelo. Caso os valores ausentes sejam relevantes para o aprendizado de máquina deve-se adotar técnicas que minimizem o efeito negativo dessa ocorrência. Não faz parte do escopo deste trabalho discutir tais técnicas, mas pode-se citar a criação de uma nova categoria para tratá-los ou alimentá-lo com um valor fora variação do atributo, por exemplo, caso a variação seja entre 0 e 1 atribui se ao valor ausente -1. Caso os valores ausentes não sejam relevantes para o aprendizado de máquina pode-se apenas retirado do processo de seleção de dados as instâncias com dados faltosos. Porém, não é incomum que tal ação cause problemas principalmente em uma base uniforme onde a retirada pode modificar o padrão das instâncias que pode diminuir a precisão do modelo. Onde uma das possibilidades para resolver esse problema é substituir o valor ausente pela média das características que possuem valor definido.

2.2.2 Seleção de atributos

Na seleção de dados analisa-se quais dados podem ser importantes para o processo de aprendizado de máquina, e remove-se os que não possuem relevância para a atividade. Para tal ação é necessário conhecer bem a base de dados e os algoritmos que serão utilizados no processo.

2.2.3 Transformação de dados

A transformação de dados para o aprendizado de máquina é necessária visto que geralmente os algoritmos utilizados suportam dois tipos de dados: numéricos e categóricos. Os dados numéricos são representados por um valor numérico, já as características categóricas são caracterizadas por dividir seus valores em diferentes categorias, podendo ser ordinal ou não.

Dito isto, é necessário transformar os dados, seja por característica do classificador, seja por característica da base de dados. Frequentemente há ocorrências de dados ordinais na base de dados que devem ser transformados em numéricos ou categóricos.

2.2.4 Validação Cruzada

Dois grandes problemas do aprendizado de máquina são a ocorrência de *overfitting* e *underfitting*.

O *overfitting* ocorre quando o modelo aprende sempre com o mesmo conjunto de treinamento e atinge precisão de 100% ou muito próximo desse valor. Porém quando é submetido a predição no conjunto de teste essa taxa cai vertiginosamente. O que nos leva a concluir que o modelo está superespecializado no conjunto de treinamento e pouco generalista para ser útil em outros conjuntos. Tal ocorrência pode ser causada por ruídos no conjunto de dados de treinamento ou o conjunto de dados de treinamento não representa de forma adequada a base de dados completa ou ainda um conjunto de dados de treinamento está pequeno demais.

O *underfitting* ocorre quando o modelo apresenta problemas de predição no próprio conjunto de dados de treinamento, porém pode se sair bem com novas instâncias. Isso ocorre quando o modelo não consegue identificar padrões no conjunto de treinamento. Tal ocorrência é vista quando o algoritmo utilizado não é adequado para a base de dados ou quando o algoritmo não consegue trabalhar bem com o conjunto de treinamento fornecido.

Devido a esses dois fenômenos é necessário que o conjunto de dados de treinamento contenha instâncias que representem de forma satisfatória a base de dados inteira, ou seja, que o conjunto de testes mantenha a representatividade de cada classe.

A validação cruzada é amplamente utilizada para evitar problemas de *overfitting* e *underfitting.* Pode-se aplicar tal técnica de várias formas: método de re-substituição, método *holdout*, método *k-folds* e *Leave-One-out.*

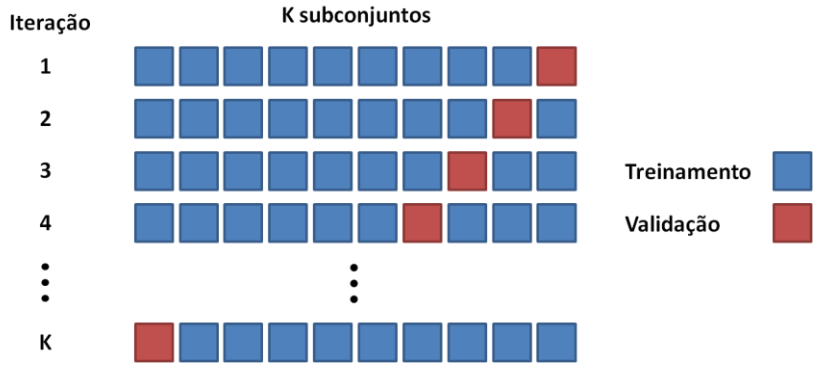
Na validação de re-substituição, o modelo aprende com todos os dados disponíveis e, em seguida, é testado no mesmo conjunto de dados. Esse processo de validação usa todos os

dados disponíveis, mas sofre seriamente de excesso de ajuste. Ou seja, o algoritmo pode ter um bom desempenho nos dados disponíveis, mas ter um desempenho ruim nos dados de teste futuros não vistos.

O método *holdout* é aplicado dividindo o conjunto de dados em dois: o conjunto de dados de treinamento e o conjunto de testes. Esta divisão é realizada de forma aleatória e repetida diversas vezes. A cada iteração se obtém a margem de erro, e a média entre as margens de erros obtidas no final é a margem de erro do modelo.

O método *k-fold* divide o conjunto de dados em *k* dobras ou partições (*folds*) de forma aleatória. Segundo Witten e Frank (2005), testes extensivos em diversas bases de dados, utilizando diversos algoritmos, identificaram o valor de *k* para identificar a melhor margem de erro como sendo 10. O conjunto de dados de treinamento é criado com *k-1* partição e 1 partição é usada para teste. São realizadas *k* iterações, onde cada partição é utilizada uma vez para testes enquanto as outras são utilizadas para treinamento.

Figura 2.1 - Validação Cruzada



Fonte: Couto (2013)

A validação cruzada *Leave-One-Out* (exclusão única) é um caso especial de validação cruzada de *k-folds* em que *k* é igual ao número de instâncias do conjunto de dados. Em outras palavras, em cada iteração, quase todos as instâncias, com exceção de uma única, são usadas para treinamento e o modelo é testado nessa única amostra.

2.3 ALGORITMOS DE CLASSIFICAÇÃO

Algoritmos de classificação foram criados para executar a tarefa de classificação supervisionada que é usada para predizer a qual categoria uma instância dos dados pertence. Para isso é utilizado um conjunto de dados rotulados que será dividido em treino e teste. Inicialmente o algoritmo aprenderá e criará padrões utilizando o conjunto de treino e por fim utilizará o conjunto de teste para avaliar se a predição foi satisfatória ou não.

2.3.1 AdaBoost

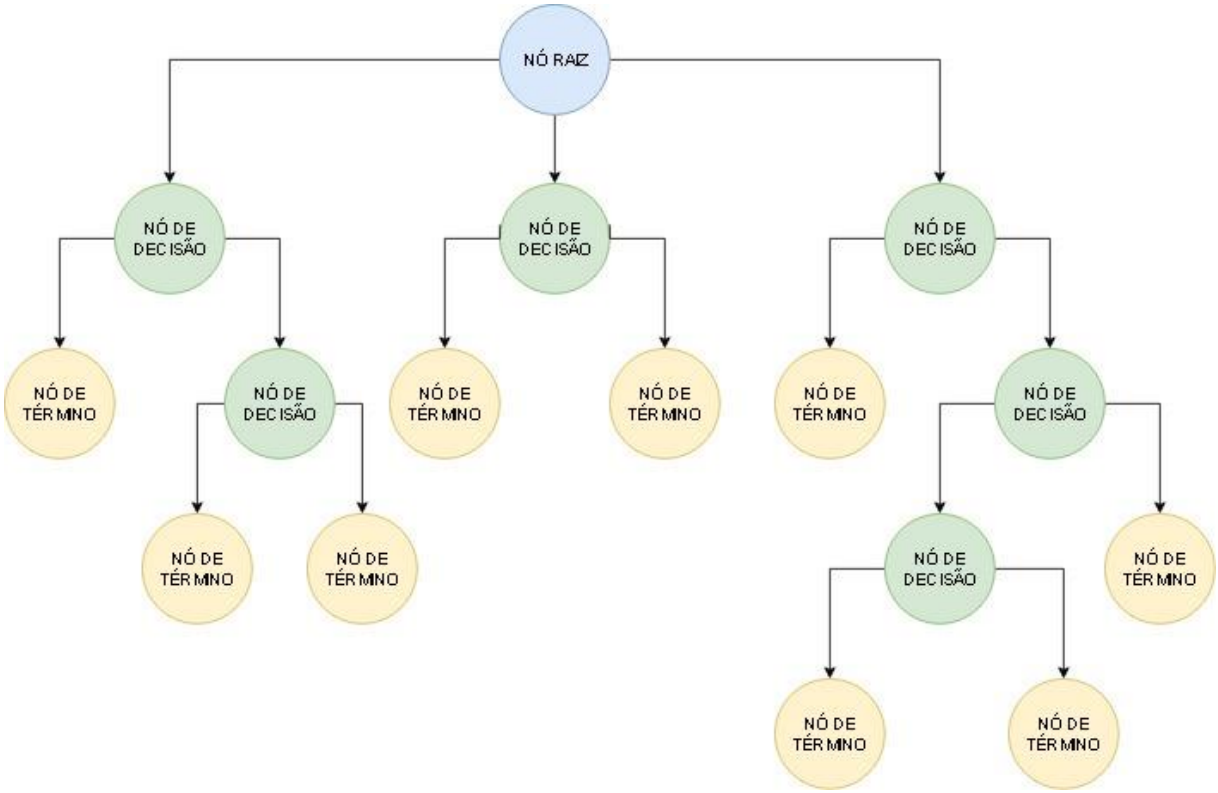
O AdaBoost ou *Adaptive Boosting* (reforço adaptativo) é um classificador que é usado em conjunto com um outro classificador que se procura reforçar. Inicialmente, usando uma árvore de decisão, por exemplo, o *AdaBoost* separa o conjunto de dados original e os classifica, em seguida, atribui pesos para as classes que foram classificadas. Aumentando o peso das erroneamente classificadas e diminuindo o peso das que foram classificadas de maneira correta. Na próxima iteração o classificador deve concentrar-se nas classes mais difíceis de prever, ou seja, as que têm maior peso. O processo se repete até que o número de iterações chegue a um limite previamente definido. Por fim a previsão final do modelo será a soma ponderada das previsões feitas pelos modelos de árvores anteriores.

2.3.2 Árvore de Decisão

Árvore de decisão (*Decision Tree*) são modelos supervisionados que podem ser utilizados tanto para classificação como para regressão. Witten e Frank (2005) descrevem o funcionamento do algoritmo da seguinte forma: “Ele (o algoritmo *Decision Tree*) utiliza o modelo “se então” que é mutuamente exclusivo. As regras são aprendidas sequencialmente usando os dados de treinamento, um de cada vez”. Já Han and Kamber (2006), descreve o funcionamento da seguinte forma: “Uma árvore de decisão possui uma estrutura de árvore, onde cada nó interno (não-folha), pode ser entendido como um atributo de teste, e cada nó-folha (nó-terminal) possui um rótulo de classe”. Com as descrições acima podemos citar um exemplo do funcionamento do algoritmo na Figura 2.2.

No modelo de classificação de teste as variáveis preditas são conhecidas e com isso a árvore realiza seu treinamento e identifica os atributos mais importantes tornando-os nós de mais alto nível, pois a mudança nestes gera maior impacto no resultado final.

Figura 2.2 - Árvore de Decisão



Fonte: o autor.

2.3.3 Gaussian Process Classifier

O *Gaussian Process Classifier* implementa processos Gaussianos para fins de classificação, mais especificamente para classificação probabilística.

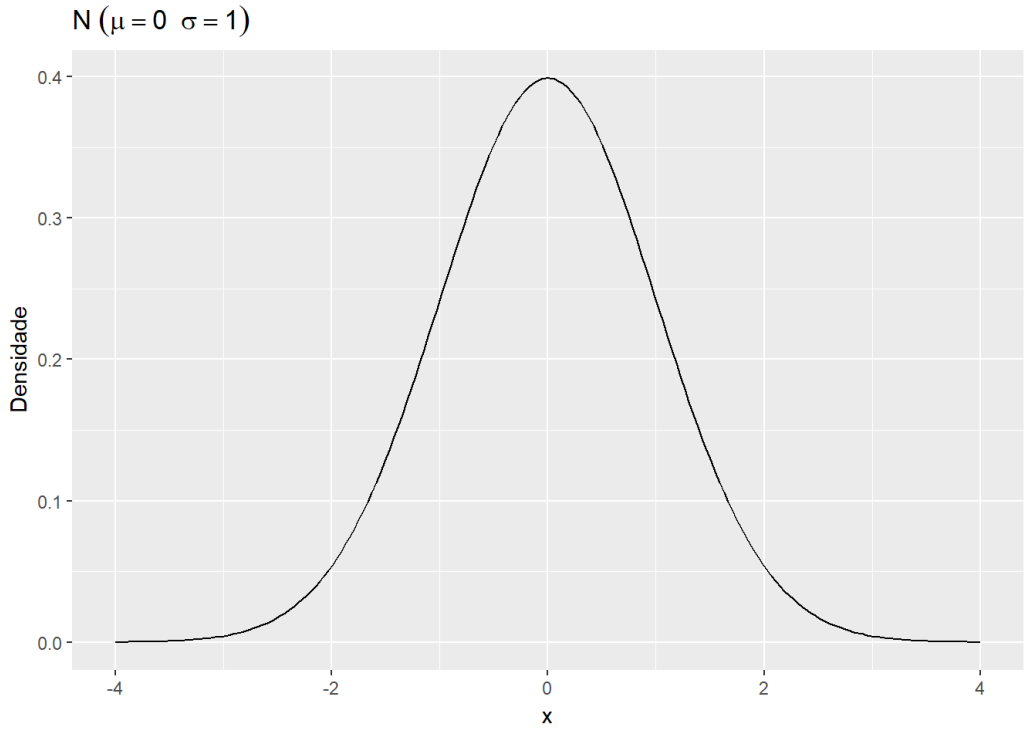
Rasmussen (2004) define o processo Gaussiano como um conjunto de variáveis de números finitos dos quais possuam distribuição normal (distribuição Gaussiana) e é obtida a função média e a função de covariância. De maneira geral podemos definir da seguinte forma:

$$f \sim GP(\mu, k)$$

Onde a função f é distribuída normalmente por um Processo Gaussiano (GP) com a função média μ e a função de covariância *k*.

A distribuição normal ou gaussiana é uma curva simétrica em torno do seu ponto médio. A figura abaixo ilustra a distribuição normal com média 0 e desvio padrão igual 1.

Figura 2.3 - Distribuição Gaussiana



Fonte: Zibett (2019)

Considerando a probabilidade de ocorrência de um fenômeno estar na área sob a curva como 100%, isso quer dizer que a probabilidade de uma observação assumir um valor entre dois pontos quaisquer é igual à área compreendida entre esses dois pontos.

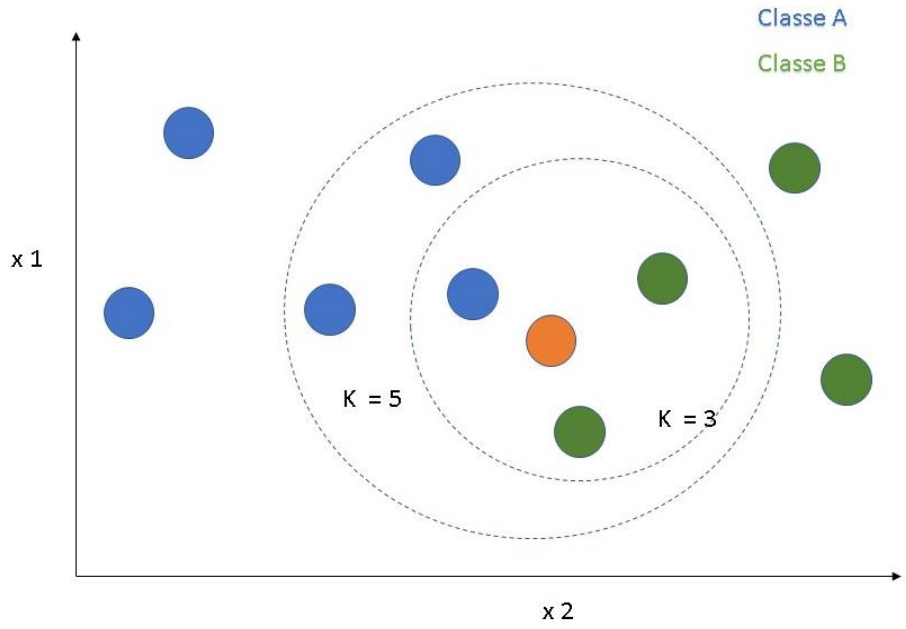
Os processos gaussianos (GP) são baseados no pressuposto de que os dados subjacentes são distribuídos normalmente em conjunto. Quanto mais próximo dessas suposições um conjunto de dados estiver, melhor será o desempenho deste classificador.

2.3.4 KNN

O KNN ou k-vizinhos mais próximos (*K-Nearest Neighbors*) é um algoritmo de aprendizagem supervisionada que consiste em encontrar, segundo alguma medida de similaridade, os k exemplos mais próximos de um exemplo ainda não-rotulado e, baseado nos rótulos desses k exemplos próximos rotular o novo exemplo (Han and Kamber, 2006). Apesar de termos várias abordagens possíveis usualmente o algoritmo atribui pesos para as amostras vizinhas. Quanto mais próximas a amostra está maior a influência dela na decisão do rótulo a ser definido para o exemplo em questão.

Na figura 2.4, com $k = 3$, temos que provavelmente a amostra ainda não rotulada (em laranja) poderá ser rotulada como classe B pela quantidade de vizinhos e relativa proximidade, porém se o valor de k for 5 o resultado pode mudar a depender do cálculo da distância levando em consideração o peso dos vizinhos mais próximos.

Figura 2.4 - *k-Nearest Neighbors*



Fonte: o autor.

2.3.5 Naive Bayes

Naive Bayes é um classificador que utiliza o teorema de *Bayes*. O teorema de *Bayes* relaciona probabilidades prévias, com a probabilidade de a ocorrência após um fato novo ter ocorrido, então é gerada uma nova probabilidade quando os fatos acontecem de maneira relacionada ou são dependentes. Também é importante ressaltar que essas probabilidades podem e devem ser revistas à medida que são observados novos fatos que podem alterar a probabilidade de cada um dos acontecimentos envolvidos (SILVER, 2013). A fórmula geral do teorema é:

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

Onde P(A|B) é a probabilidade do evento A ocorrer dado que o evento B ocorreu, P(B|A) é a probabilidade do evento B ocorrer dado que o evento A ocorreu, $P(A)$é a probabilidade do evento A ocorrer e $P(B)$é a probabilidade do evento B ocorrer,

Usando o teorema de *Bayes*, podemos encontrar a probabilidade de A acontecer, dado que B ocorreu. Aqui, B é a evidência e A é a hipótese. A suposição feita é que os preditores são independentes e quando um tiver alteração não deve ter consequências para o outro preditor.

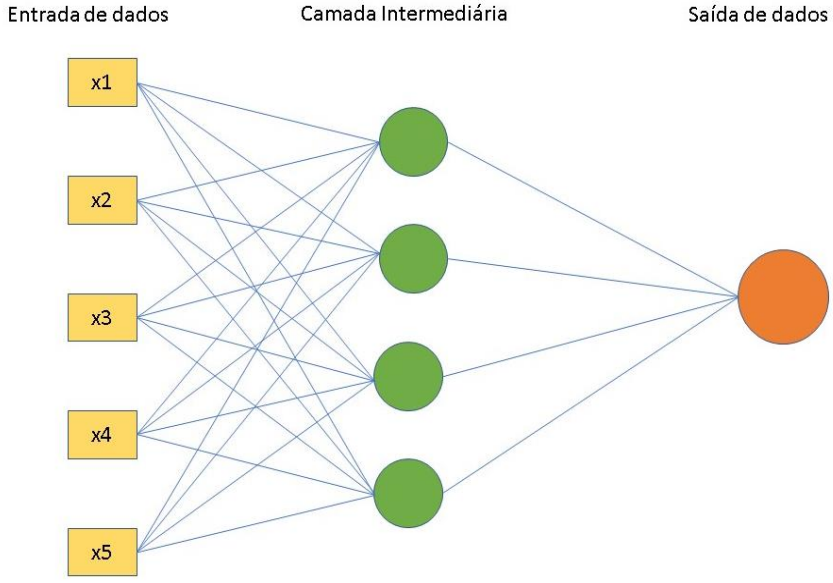
O algoritmo *Naive Bayes* é rápido e fácil de implementar, porém, quando os preditores não são independentes entre si, a rigor, o teorema de *Bayes* não se aplica e opor isso o desempenho do algoritmo pode não ser satisfatório.

2.3.6 Rede Neural

A Rede Neural (*Neural Net*), também chamada de Rede Neural Artificial (RNA) tem esse nome devido ao seu funcionamento que se assemelha ao sistema nervoso biológico humano. Haykin diz que: “As RNAs são processadoras massivamente paralelos e distribuídos que têm uma propensão natural para armazenar o conhecimento proveniente da experiência e torná-lo útil. Desta forma, assemelhando-se ao cérebro humano em dois aspectos: 1. o de que o conhecimento é adquirido pela rede através de um processo de aprendizado e 2. o de que as intensidades das conexões entre neurônios, conhecidas como pesos sinápticos, são usadas para armazenar o conhecimento." (HAYKIN, 1994). De um modo mais prático também temos que: “A Rede Neural é um conjunto de unidades de entrada / saída conectadas onde cada conexão tem um peso associado a ela”. (Witten e Frank, 2005).

Para a atribuição de pesos e o treinamento do modelo podem ser criadas camadas intermediárias que ajudarão a predição da variável alvo. Um esquema com uma camada intermediária é apresentado na figura 2.5.

Figura 2.5 - Rede Neural



Fonte: o autor.

2.3.7 QDA

Para a definição do algoritmo de classificação QDA ou *Quadratic discriminant analysis* (Análise discriminante quadrática) é necessária uma rápida definição e alguns conceitos estatísticos.

A análise discriminante é uma técnica da estatística multivariada que estuda a separação de objetos de uma população em duas ou mais classes. A discriminação ou

separação é a primeira etapa, sendo a parte exploratória da análise e consiste em procurar características capazes de serem utilizadas para alocar objetos em diferentes grupos previamente definidos. (KHATTREE & NAIK, 2000)

A matriz de covariância é uma matriz quadrada que contém as variâncias (medida de dispersão que mostra o quão distante cada valor está do valor médio) e covariâncias (taxa de dependência entre variáveis) associadas a diversas variáveis. Os elementos da diagonal da matriz contêm as variâncias e os elementos fora da diagonal contém as covariâncias entre todos os possíveis pares de variáveis. (MINITAB, 2019)

Tabela 1 - Matriz discriminante linear

	X	Y	Z
X	**2,0**	-0,86	-0,15
Y	-0,86	**3,4**	0,48
Z	-0,15	0,48	**0,82**

Fonte: MINITAB, 2019

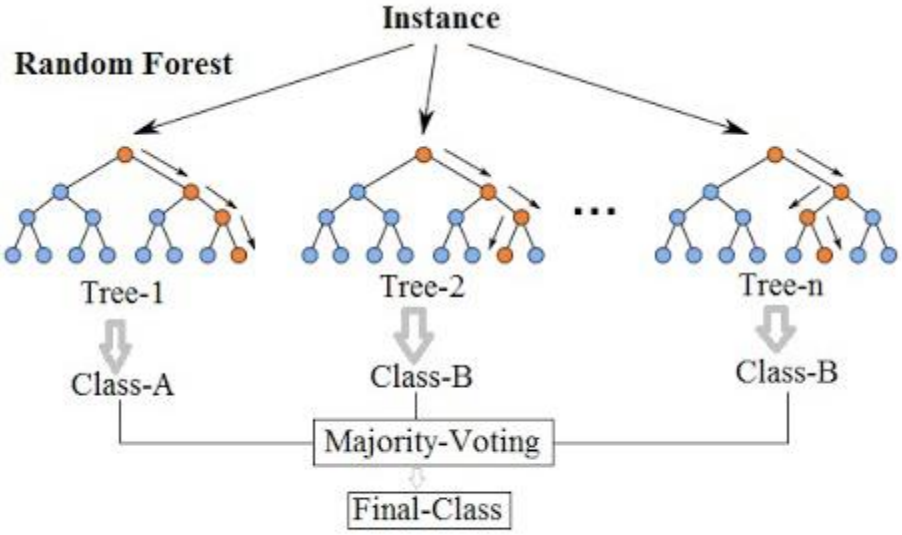
Por fim, a análise discriminante quadrática, que não prevê que os grupos possuam matrizes de covariância iguais, é utilizada para casos mais gerais que não se limitam aos casos do modelo da matriz discriminante linear, assim como é explicado em MINITAB (2019) Ao contrário da distância linear, a distância quadrática não é simétrica. Em outras palavras, a função discriminante quadrática do grupo *i* avaliado com a média do grupo *j* não é igual à função discriminante quadrática de grupo *j* avaliado com a média do grupo *i*. Nos resultados, a distância quadrática é chamada distância quadrada generalizada. Se o determinante da matriz de covariância de grupo amostral for menor que um, a distância quadrada generalizada pode ser negativa. (MINITAB, 2019).

Dessa forma o algoritmo utilizará a análise discriminante quadrática para dividir os conjuntos de treino e teste em grupos, mediante os critérios avaliados, e faz a predição do resultado.

2.3.8 Random Forest

O classificador *Random Forest* (Floresta Aleatória) gera um conjunto de árvores de decisão dentro do mesmo objeto. Cada conjunto de classificação é analisada e com os atributos mais referenciados pelas árvores é construída a árvore mais adequada que realiza a classificação

Figura 2.6 – *Random Forest*



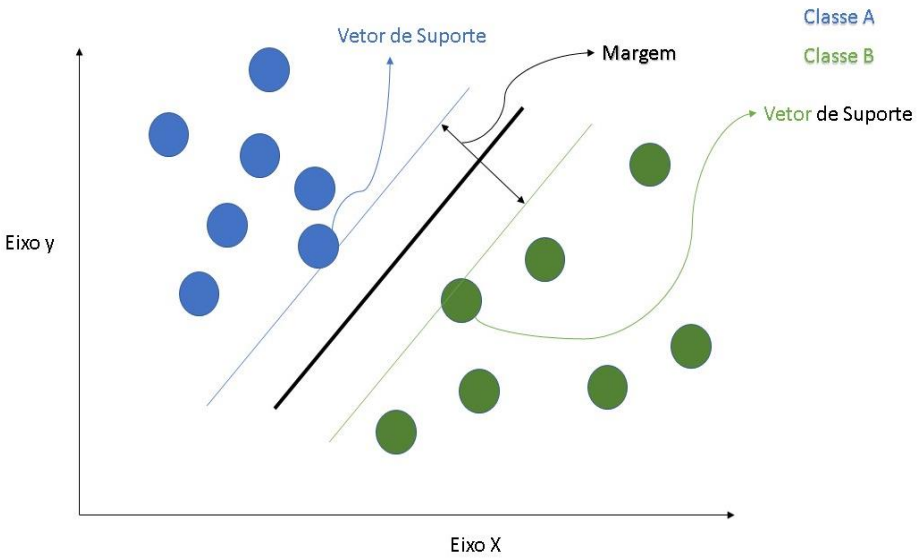
Fonte: *kdnuggets* (2019)

Na figura 2.6 observamos que o *Random Forest* utilizou várias árvores de decisão e verificou quais foram os atributos mais votados como importantes e por fim do mais votado construiu a árvore final que fez a predição.

2.3.9 SVM

O SVM, Máquina de Vetores de Suporte (*Support Vector Machine*) cria um hiperplano que divide os exemplos dados em dois rótulos. A margem é obtida pela distância entre o hiperplano e os vetores mais próximos a ele, denominados vetores de suporte.

Figura 2.7 - SVM



Fonte: o autor.

Quando não é possível ter os dados de treinamento separáveis de forma satisfatória por meio de uma função linear, é necessária a utilização da função *kernel*. Em Bonesso (2013), define-se que o truque do *kernel* possibilita que o espaço original seja mapeado em um espaço de produto escalar de alta dimensão chamado espaço de características, onde os dados podem ser linearmente separáveis. Os *kernels* mais comumente usados na literatura são *kernel* Linear, Gaussiano e RBF (*Radial basis function* ou Função de Base Radial).

2.3.10 XGBoost

O *XGBoost* ou *Extreme Gradient Boosting* (Reforço extremo do gradiente) é uma implementação do algoritmo de árvore de decisão baseado em uma estrutura de *Gradient boosting.*

O *Gradient Boosting* é muito parecido com *AdaBoost*, porém há uma diferença fundamental entre eles: a maneira como cada um identifica as deficiências de classes fracas. Enquanto o modelo *AdaBoost* identifica as deficiências aumentando seu peso para a próxima iteração, o *Gradient Boosting* executa o mesmo usando gradientes na função de perda. A função de perda é uma medida que indica quão bons são os coeficientes do modelo no ajuste dos dados subjacentes.

Por fim, o *XGBoost* é uma atualização do *Gradient Boost* utilizando o algoritmo de árvore de decisão. Os itens mais importantes é que o *XGBoost*, de maneira geral, é mais rápido e tem um melhor desempenho que seu antecessor.

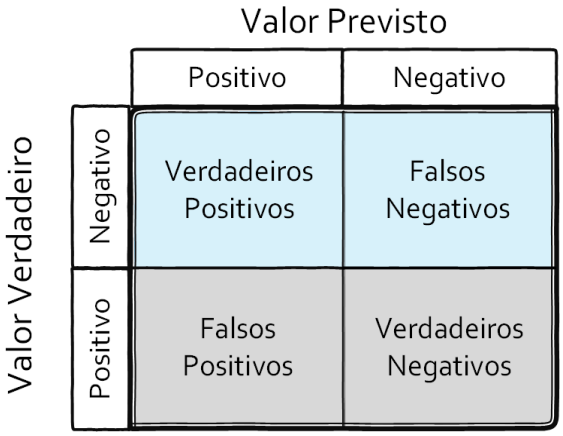
2.4 AVALIAÇÃO DOS MODELOS DE CLASSIFICAÇÃO

Para avaliar o desempenho de cada algoritmo é necessária a utilização de algumas métricas. Essas métricas, de maneira geral, têm como base as quantidades de Verdadeiros Negativos, Falsos Negativos, Falsos Positivos e Verdadeiros Positivos. Cada valor é explicado em seguida:

- Verdadeiro Negativo (TN - *True Negative*) o valor é negativo e o algoritmo classifica corretamente o valor como negativo;
- Falso Negativo (FN - *False Negative*) o valor é positivo e o algoritmo classifica erroneamente o valor como negativo;
- Falso Positivo (FP - *False positive*) o valor é negativo e o algoritmo classifica erroneamente o valor como positivo e;
- Verdadeiro Positivo (TP - *True Positive*) o valor é positivo e o algoritmo classifica corretamente o valor como positivo.

Com os valores apresentados acima é possível construir a matriz de confusão, que é uma tabela que mostra o desempenho da classificação de um modelo de previsão comparando o valor previsto com o valor real.

Figura 2.8 - Matriz de confusão



Fonte: Andreoni (2012)

Na matriz de confusão a diagonal principal representa as predições corretas, ou seja, Verdadeiros Positivos e Verdadeiros Negativos. Já os outros elementos representam as predições erradas, ou seja, Falsos Negativos e Falsos positivos.

Além da representação dada pela matriz de confusão, com esses valores coletados podemos calcular algumas métricas importantes para a avaliação do desempenho do classificador, que detalhamos em seguida

2.4.1. Accuracy

A métrica *accuracy* (acurácia) mede a taxa de acerto, que é a razão entre o somatório das previsões corretas (verdadeiros positivos com verdadeiros negativos) sobre o somatório das previsões, ou seja:

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

Uma pontuação alta pode revelar um bom desempenho do classificador. Porém olhar apenas para essa métrica poderá inferir em erro pois quando a base de dados não é balanceada o classificador tende a uniformizar a predição. Por exemplo, um classificador binário prevê corretamente 950 casos (TP = 100 e TN = 850) e erroneamente 200 casos (FP = 100 e FN = 50) teremos uma *accuracy* de aproximadamente 83%. Mas se um classificador uniformizasse a predição, sem fazer nenhuma análise, prevendo todos como negativos, poderíamos ter 1000 corretamente classificados (TP = 0 e TN = 1000) e 150 casos erroneamente (FP = 0 e FN = 150) e teríamos uma *accuracy* de aproximadamente 87%. Ou seja, apenas analisando a métrica $accuracy$ pode-se deduzir erroneamente que um classificador que não realiza nenhuma análise é melhor que um classificador que realiza análises.

2.4.2 Precision

A *Precision* (precisão) verifica a relação entre as previsões positivas realizadas corretamente e todas as previsões positivas (incluindo as falsas).

$$precision = \frac{TP}{TP + FP}$$

A *Precision* nos dá informação sobre falsos positivos. Quanto maior o valor, melhor, pois indica que os falsos positivos não estão causando muito impacto na precisão do classificador.

2.4.3 Recall

O *Recall* é utilizado para indicar a relação entre as previsões positivas realizadas corretamente e todas as previsões que realmente são positivas

$$recall = \frac{TP}{TP + FN}$$

O *Recall* nos informa também os falsos negativos. Quanto maior o valor, melhor, pois indica que os falsos negativos não estão causando muito impacto na precisão do classificador.

2.4.4 F1

O f1 ou f-score é a métrica onde podemos visualizar o desempenho de *Precision* e *Recall* juntas. Para isso usamos a média harmônica.

A média harmônica está relacionada ao cálculo matemático das situações envolvendo as grandezas inversamente proporcionais. (Ferretto, 2019)

Utilizando a média harmônica temos:

$$f1 = \frac{2 * (Precision * Recall)}{Precision + Recall}$$

Caso o f1 forneça um valor alto podemos confiar que realmente o desempenho do classificador é bom, pois a análise está levando em consideração vários fatores e não apenas a taxa de acerto (*accuracy*). Quando o valor desta métrica for baixo, o classificador não está indo bem em alguma métrica anterior.

2.4.5 Specificity

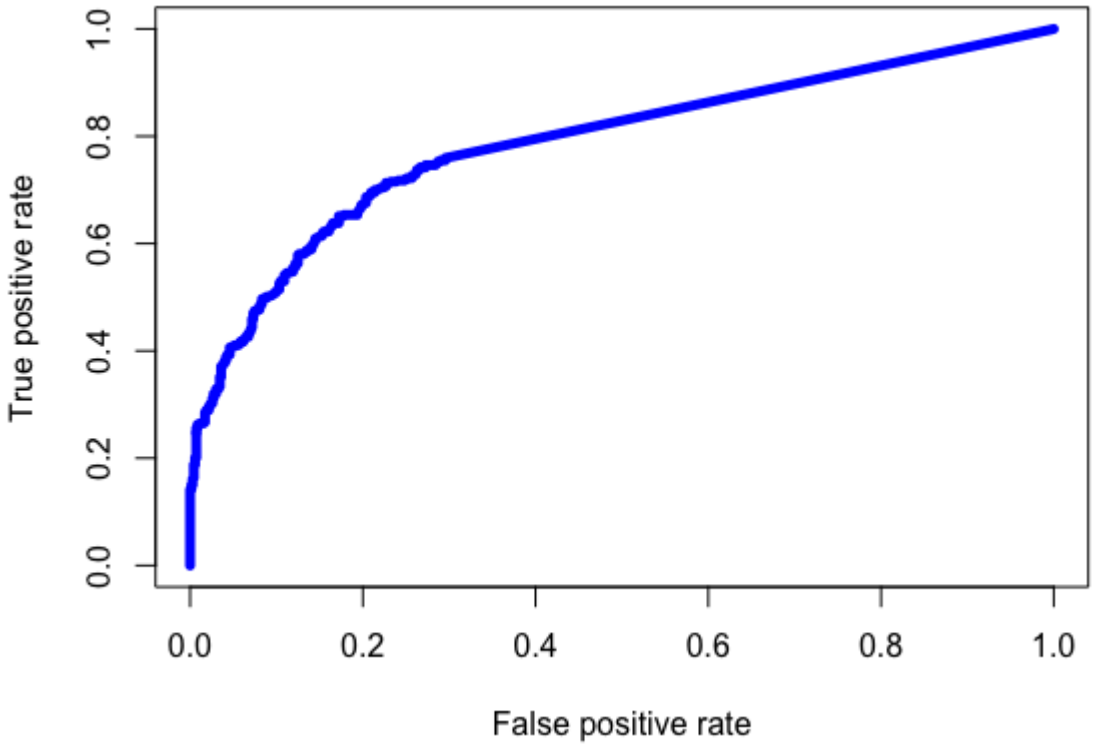
A *Specificity* (especificidade) é a proporção de casos negativos que foram identificados corretamente. Para calculá-la dividimos o número de acertos negativos pelo total de negativos.

$$specificity = \frac{TN}{TN + FP}$$

2.4.6 Área da Curva ROC

A curva ROC ou *Receiver Operating Characteristics* (Características de operação do receptor) é uma métrica para a visualização e a seleção de classificadores baseado no seu desempenho. A figura 2.9 mostra um exemplo de uma curva ROC que é gerada tendo em seu eixo vertical a taxa de Verdadeiros positivos e no eixo horizontal a taxa de Falsos Positivos.

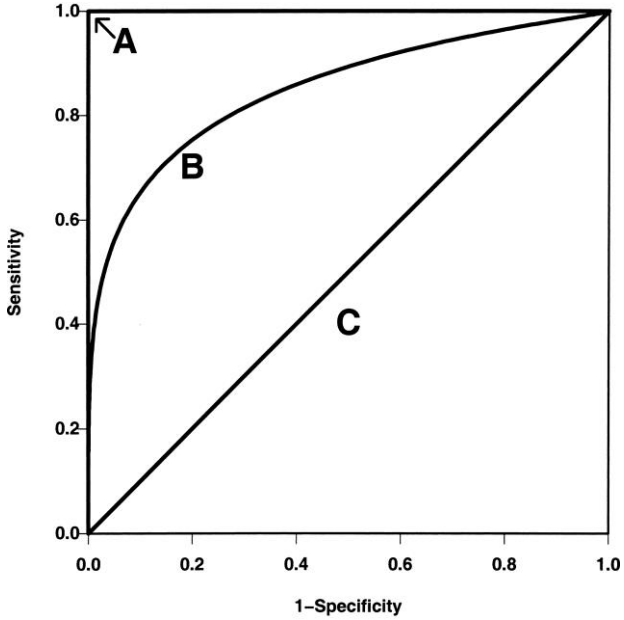
Figura 2.9 - Curva ROC



Fonte: Estatsite (2019)

Muitos pontos são importantes no gráfico ROC. O ponto inferior esquerdo (0, 0) representa um classificador que nunca gera uma classificação positiva, como consequência também não comete erros false positives. Um classificador com estratégia oposta, gera *true positives* e é representada pelo ponto superior direito (1, 1). O ponto (0, 1) representa uma classificação perfeita e o ponto (1,0) representa uma classificação apenas com erros. Classificadores no lado esquerdo do gráfico ROC (perto do eixo Y) são chamados de conservadores, pois fazem classificações positivas somente com uma evidência forte, portanto cometem poucos erros *false positives*. Classificadores no lado direito são ditos liberais, pois fazem classificações positivas com pouca evidência, mas comentem muitos erros *false positives*.

Figura 2.10 - Curva ROC aleatória, comum e perfeita



Fonte: Zou (2007)

Na figura 2.10 temos que a curva C representa um classificador aleatório que tem 50% de acertar e 50% de errar a sua predição. A curva B um classificador melhor que o anterior e o classificador A que representa o classificador perfeito.

Para simplificar a análise da curva ROC calculamos a AUC ou *Area Under the Curve* (Área Sob a Curva) onde teremos agregados todos os limites da curva ROC. O valor da área varia de 0 a 1 e quanto mais próximo do 1 esse valor estiver, melhor pois indica que o classificador está acertando mais que errando e quanto mais próximo de 0 pior pois indica que o classificador está errando mais que acertando.

CAPÍTULO 3 – MATERIAIS E MÉTODOS

Para o processo de aprendizado de máquina proposto neste trabalho foi usado um computador com um processador Intel Core I7 - 5500U 2.4GHz, 8GB de memória RAM, 1TB de HD e sistema operacional Windows 10.

Os programas e bibliotecas utilizados foram a linguagem de programação Python (versão 3.7.4). Foi utilizado também a IDE *Spyder* (versão 3.3.6) que é um console Python interativo com visualizador de documentação, além de gerenciadores de variáveis e de arquivos. Essas ferramentas auxiliaram na rapidez de visualização de dados. Já biblioteca *Scikit-Learn* ou simplesmente *Sklearn* (SCIKIT-LEARN, 2019) na versão 0.21.3 foi escolhida pois é uma das bibliotecas mais famosas, consolidadas e documentadas para criação de modelos supervisionados e não-supervisionados em Python. A biblioteca Pandas (PANDAS, 2019) na versão 0.25.1 foi usada por ser a principal e mais completa ferramenta para análise e estrutura de dados de alta performance, além de ser de fácil uso. manipula dados como a biblioteca *Numpy* versão 1.17.1 (NUMPY, 2019) que é uma das mais completas e fáceis de usar para a computação científica e a biblioteca *XGBoost* versão 0.90 (XGBOOST, 2019) que tem mostrado resultados extremamente animadores no aprendizado de máquina.

O conjunto de dados utilizado também foi descrito no capítulo 1.4 não tendo ocorrências de valores ausentes.

Para a preparação dos dados, as técnicas de limpeza de dados e tratamento de valores ausentes não foram utilizadas pois que a base escolhida não necessitou nenhuma ação dessas técnicas. Foi utilizada a técnica de transformação de tipo de dados em 12 atributos da base de dados que eram do tipo ordinal e em outros 10 que eram do tipo categóricos. A primeira transformação foi realizada através da utilização de funções criadas pelo autor para esse fim, os atributos que eram *strings* ordinais foram convertidos em numerais mantendo a ordem original. Há uma exceção a afirmação anterior que é atributo “esp” que é o atributo a ser predito (atributo alvo da classificação) e que foi transformado em binário. O atributo originalmente contém *strings* como: “*Best*”, “*Very Good*”, “*Good*”, “*Pass*” e “*Fail*”. Foi observado a ausência da ocorrência “Fail” na base de dados então decidiu-se que a transformação mais adequada é 0 para “Desempenho Regular” que representa os alunos com desempenho “*Fail*”, “*Pass*” e “*Good*” e 1 para “Bom Desempenho” que representa os alunos com desempenho “*Very Good*” e “*Best*”. A segunda transformação de dados categóricos com *strings* e transformadas em binários utilizando o módulo “*OneHotEncoder*” da biblioteca “*sklearn.preprocessing*”.

A técnica de validação cruzada escolhida foi a *k-fold.* A base foi dividida em 10 folds de forma aleatória balanceada, garantido que em cada fold contivesse 50% dos dados pertencentes a classe 0 (Desempenho regular) e 50% dos dados pertencentes a classe 1 (Bom desempenho). Para tanto, observamos que, após a transformação de dados supracitada, a base continha 81 instâncias com "esp" = 0 e 50 instâncias com "esp" = 1, portanto, inicialmente foi necessário realizar uma subamostragem aleatória da classe majoritária de forma que ambas as classes ficassem igualmente representadas na validação cruzada.

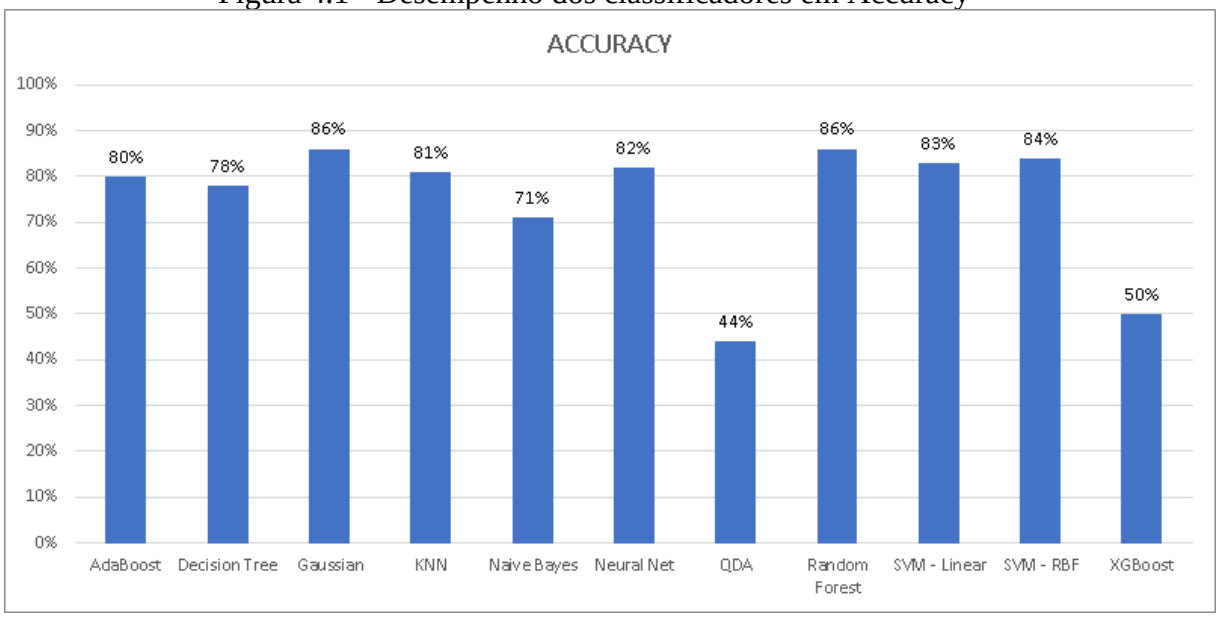
Foram aplicadas as 10 técnicas de classificação de aprendizado de máquina que estão listadas no capítulo 1.4, detalhe para a técnica SVM que foi aplicada duas vezes. Uma com *kernel* linear e outra com o *Kernel* RBF. Todas as técnicas apresentadas no capítulo 1.4 foram calculadas para cada classificador.

CAPÍTULO 4 – RESULTADO E DISCUSSÃO

Neste capítulo, serão apresentados os resultados da análise do desempenho dos modelos de classificação com base nas 8 métricas apresentadas na seção 2.4.1. Em seguida, serão apresentados os classificadores que melhor se adequaram ao conjunto de dados utilizado.

O gráfico de desempenho dos classificadores na primeira métrica apresentada é de *Accuracy*, que mede a taxa de acerto do classificador.

Figura 4.1 - Desempenho dos classificadores em Accuracy

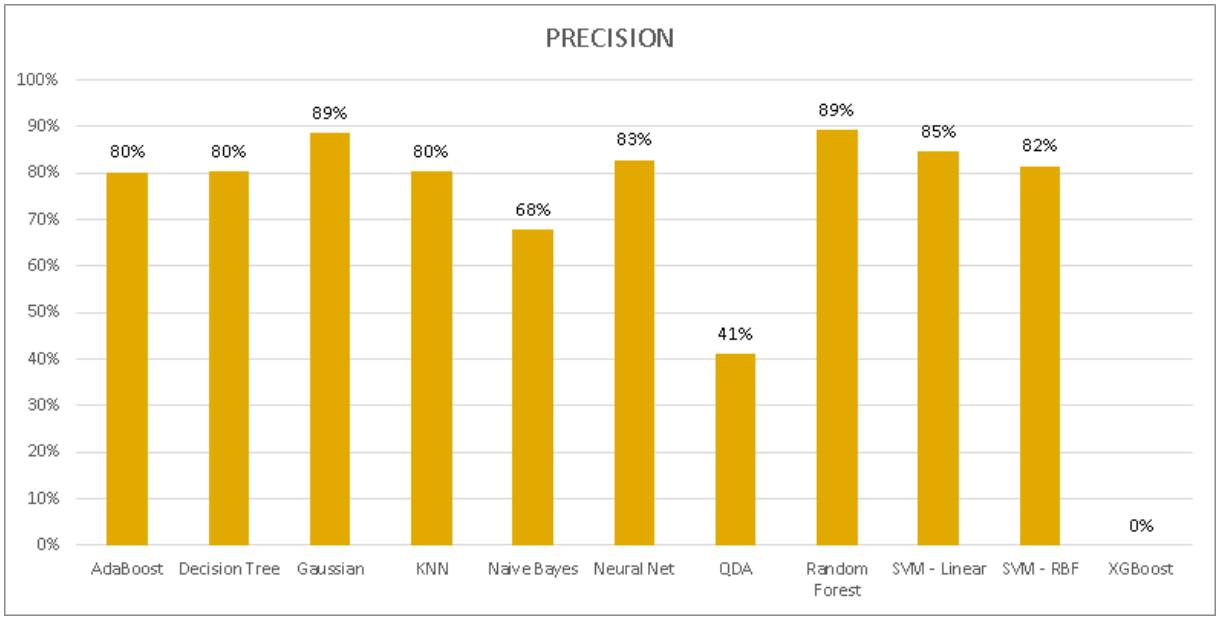


fonte: o Autor

O gráfico apresenta os classificadores *Gaussian Classifier* e *Random Forest* tiveram um bom desempenho ambos com 86% de precisão, já os classificadores QDA e *XGBoost* tiveram desempenho ruim 44% e 50% respectivamente. Como já comentado, a acurácia é uma métrica fácil de entender, mas sua simplicidade pode esconder certas deficiências dos modelos, sendo assim necessário analisar outras métricas para que possamos avaliar melhor o desempenho dos modelos.

O próximo gráfico apresentado é o de *Precision* que nos dá informação sobre o impacto que falsos positivos tiveram na precisão do classificador.

Figura 4.2 - Desempenho dos classificadores em *Precision*

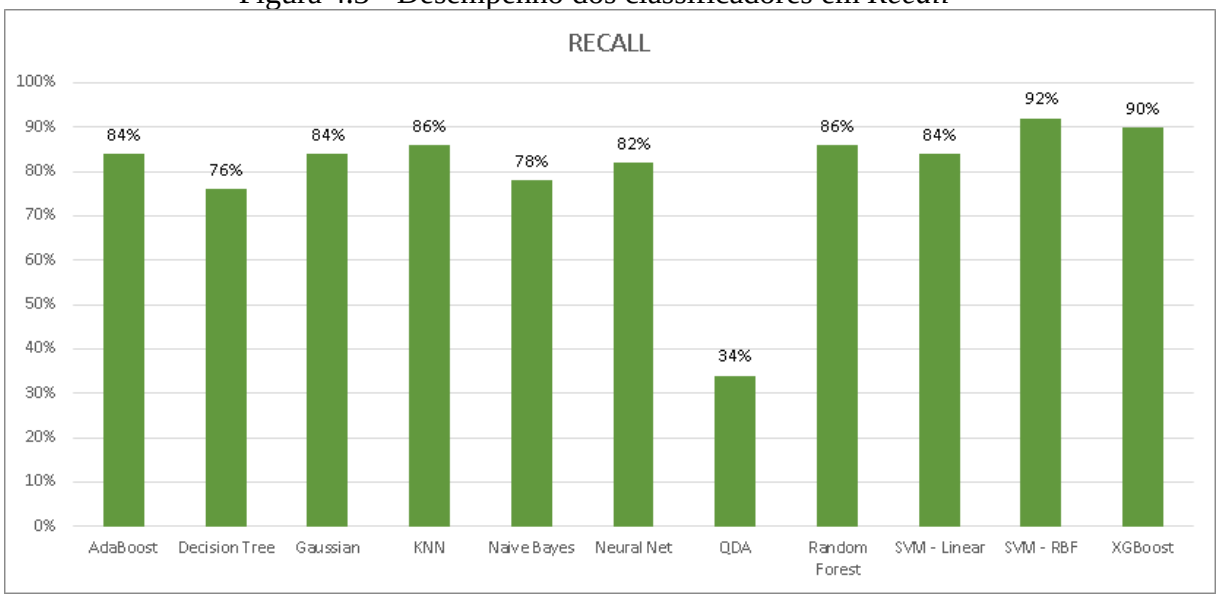


fonte: o Autor

Nesta métrica novamente os classificadores *Gaussian Classifier* e *Random Forest* tiveram o melhor desempenho ambos com 89%. Indicando os falsos positivos fora de apenas 11% das predições. Já os classificadores QDA e *XGBoost* tiveram desempenho ruim com 41% e 0% respectivamente, indicando que mais da metade das predições positivas do QDA não estavam corretas.

Junto com a *precision* é pertinente verificar o desempenho dos classificadores por meio da métrica *Recall* que nos informa os impactos dos os falsos negativos no desempenho do classificador.

Figura 4.3 - Desempenho dos classificadores em *Recall*

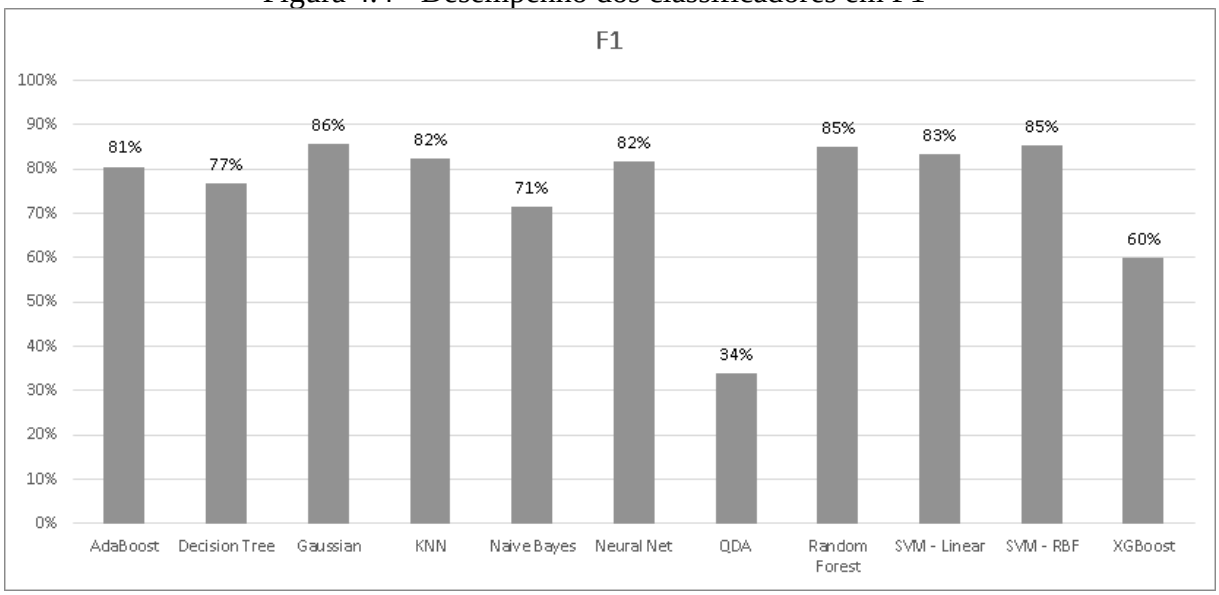


fonte: o Autor

Com o gráfico acima verificamos que os melhores desempenhos foram dos classificadores SVM com *Kernel* RBF e do *XGBoost* com desempenho 92% e 90% respectivamente. Isso indica que os falsos negativos foram de apenas 8% e 10% das predições. Destaque também para o desempenho satisfatório dos classificadores *Gaussian Classifier* e *Random Forest* com 84% e 86% respectivamente. Em contrapartida o classificador QDA teve apenas 34% na métrica indicando que os falsos negativos foram 66% dos dados preditos.

Com os dados acima é pertinente verificar o desempenho dos classificadores na métrica f1, que relaciona *precision* e *recall* e contribui para a análise do desempenho do classificador.

Figura 4.4 - Desempenho dos classificadores em F1

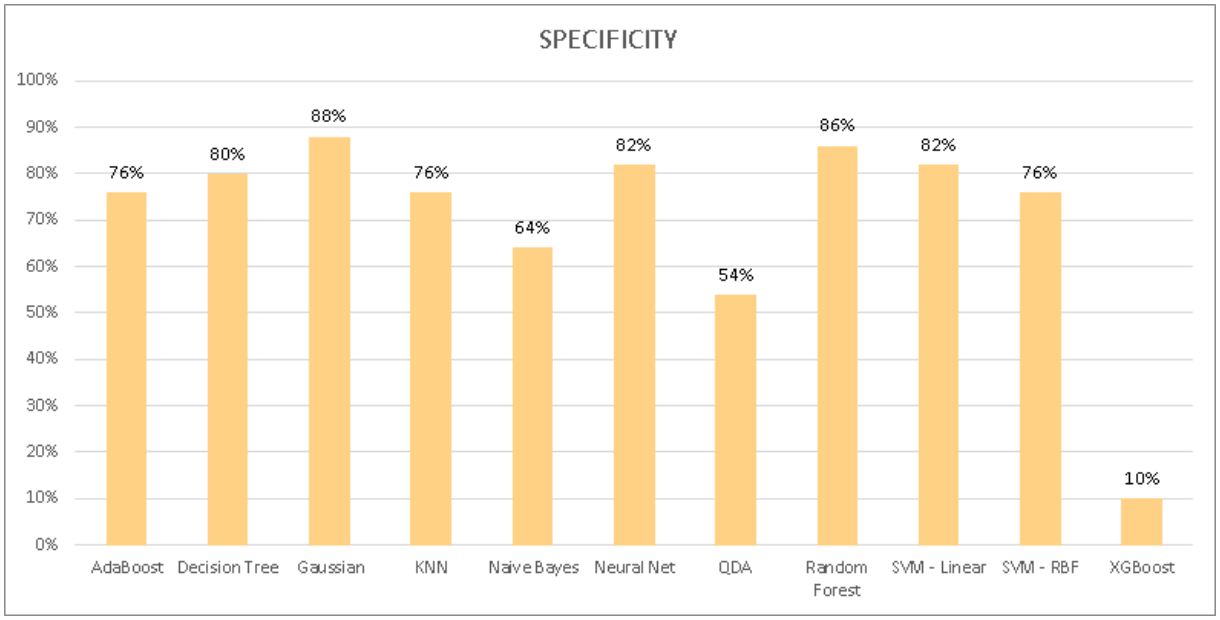


fonte: o Autor

Nesta relevante métrica temos os classificadores *Gaussian Classifier* com 86%, *Random Forest* e SVM com *Kernel* RBF ambos que 85% indicando que as falsas predições não tiveram grande impacto no desempenho geral dos classificadores. Como era previsto pelo desempenho nas métricas anteriores, o classificador QDA teve um desempenho ruim com apenas 34% nesta métrica.

A *specificity*, que mede o impacto no desempenho do classificador de casos negativos que foram identificados corretamente, temos:

Figura 4.5 - Desempenho dos classificadores em *specificity*

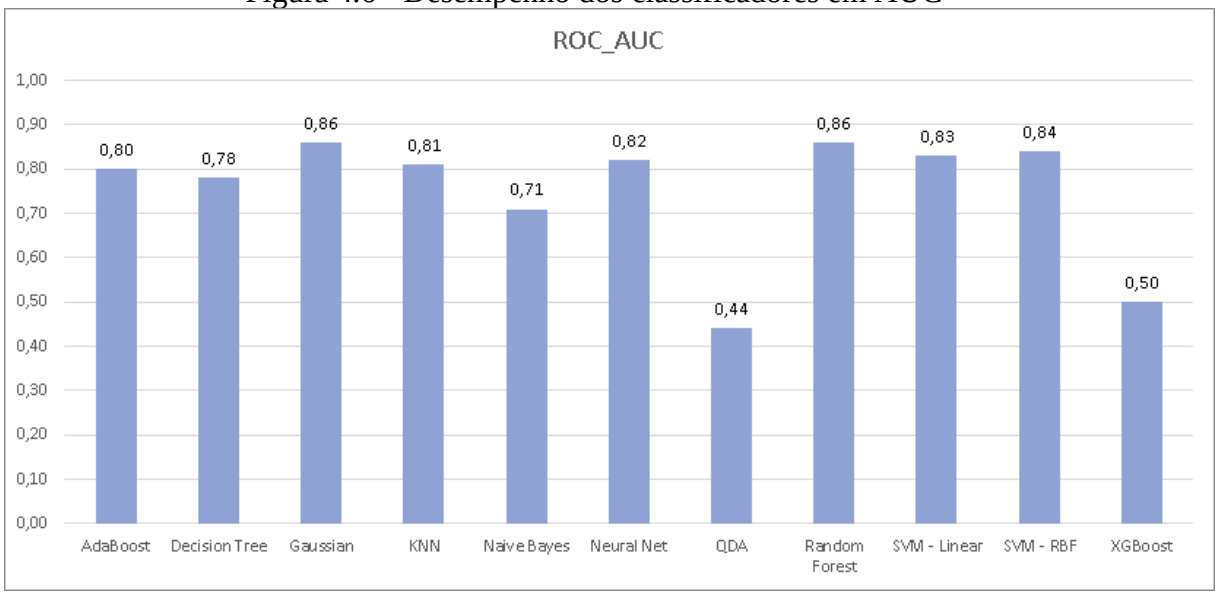


fonte: o Autor

Os classificadores *Gaussian Classifier* e *Random Forest* apresentaram novamente os melhores desempenhos, indicando que os casos negativos foram indicados corretamente em 88% e 86% respectivamente. Já o classificador *XGBoost* teve um desempenho muito ruim indicando corretamente os casos negativos em apenas 10% dos casos.

Com as duas últimas métricas podemos gerar a curva ROC e principalmente calcular a área sob a curva ROC (AUC), que é apresentada no gráfico abaixo:

Figura 4.6 - Desempenho dos classificadores em AUC

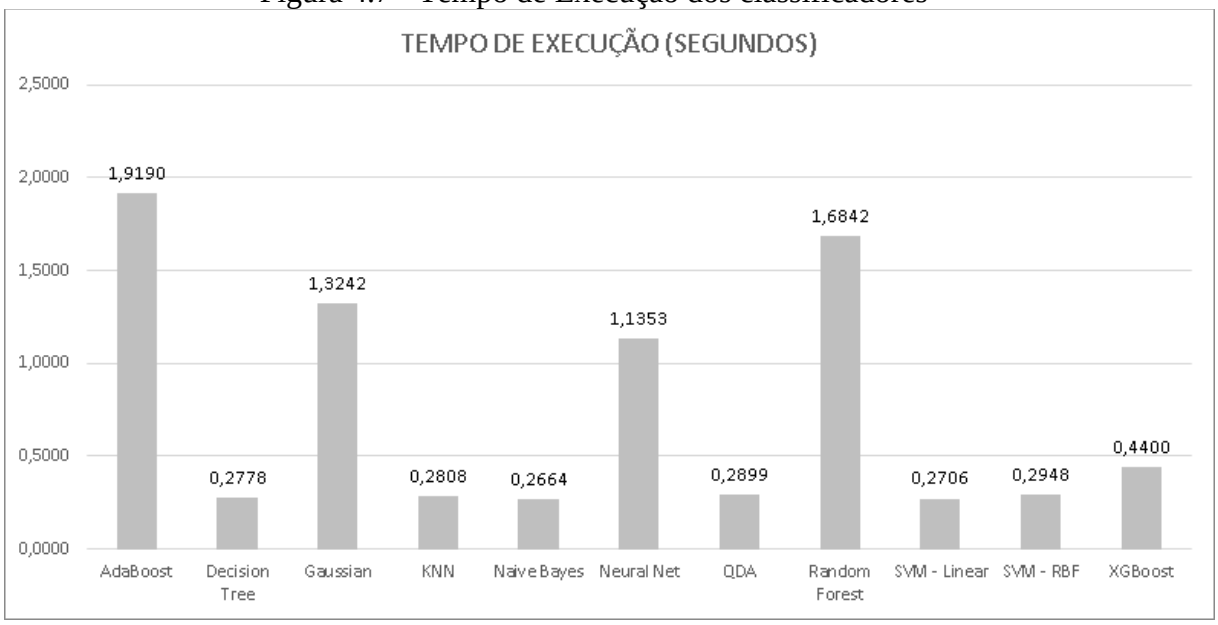


fonte: o Autor

Verificamos que com 0,86 os classificadores *Gaussian Classifier* e *Random Forest* foram novamente os melhores. E com apenas 0,44 e 0,50 os classificadores QDA e *XGBoost* tiveram o pior desempenho respectivamente.

Outra métrica relevante é o tempo de execução que verifica o custo da aplicação de cada técnica de classificação para o usuário.

Figura 4.7 - Tempo de Execução dos classificadores



fonte: o Autor

Podemos notar que o classificador *AdaBoost* teve o maior tempo de execução com aproximadamente 1,91 segundos, é importante observar que os classificadores *Gaussian Classifier* e *Random Forest* que tiveram ótimo desempenho nas métricas anteriores necessitaram de um tempo considerável para finalizar a execução com aproximadamente 1,32s e 1,68s respectivamente. Já o classificador SVM tanto com *kernel Linear* e tanto com *kernel* RBF tiveram um bom desempenho nas métricas anteriores e um pequeno tempo de execução, com aproximadamente 0,27s e 0,29s respectivamente. Nesta métricas o classificador que teve o melhor desempenho foi o *Naive Bayes* consumindo apenas 0,26s do tempo computacional.

Por fim, outro resultado relevante foi obtido pelo classificador *Random Forest* que pontua os atributos mais relevantes para a classificação. Abaixo a tabela 2 com os resultados obtidos.

Tabela 2 - Relevância dos atributos na base de dados

ORDEM	ATRIBUTO	IMPORTÂNCIA	DESCRIÇÃO
1	NF	0,257457	Número de amigos.
2	FO	0,1917699	Ocupação do pai
3	FMI	0,0977992	Renda mensal familiar
4	MO	0,07910527	Ocupação da mãe
5	MQ	0,0439803	Qualificação educacional da mãe
6	ME	0,039986719	Línguas indianas modernas
7	FS	0,0397117	Tamanho da Família
8	SH	0,0361761	Horas de Estudo
9	AS	0,03148573	Forma de admissão
10	MS	0,03148573	Estado Civil
11	TT	0,0236194	Tempo de viagem da faculdade e sua casa
12	GE	0,01802921	GÊNERO
13	LS	0,01802921	Morar na cidade ou vila
14	ARR	0,01781041	Verifica se aluno teve algum trabalho com falha em qualquer um dos semestres anteriores
15	IAP	0,0174095	Porcentagem de Avaliação Interna garantida pelo aluno no nível de Grau
16	FQ	0,0122793	Qualificação educacional do pai
17	ATD	0,00905267	Frequência da turma
18	CST	0,00567101	Casta
19	GS	0,00522127	Escola em que Estudante frequentou em classe X média
20	TNP	0,00144608	Percentual alcançado pelo aluno na Classe X
21	TWP	0,00077432	Porcentagem de Classe XI garantida pelo aluno

Fonte: o autor

Na tabela 2 podemos observar que os quatro atributos mais relevantes são: NF (Número de amigos) com alta pontuação de aproximadamente 0,26 pontos, FO (Ocupação

do pai) com também expressivos 0,19 pontos aproximadamente, FMI (Renda familiar mensal) com 0,10 pontos aproximadamente e MO (Ocupação da mãe) com aproximadamente 0,08 pontos. Já os quatro atributos que aparentemente não são relevante são: CST (Casta) com aproximadamente 0,006 pontos, GS (Escola em que Estudante frequentou em classe X) com aproximadamente 0,005 pontos, TNP (Percentual alcançado pelo aluno na Classe X) com aproximadamente apenas 0,001 pontos e o TWP (Porcentagem de Classe XI garantida pelo aluno) com aproximadamente irrisórios 0,0008 pontos.

CAPÍTULO 5 – CONSIDERAÇÕES FINAIS

Pode-se concluir que apesar de todos os classificadores usados terem conseguido realizar a tarefa proposta, os algoritmos *Gaussian Classifier* e *Random Forest* tiveram os melhores resultados. Ambos predizendo corretamente em cerca de 86% dos casos. Com essa taxa, é possível utilizá-los para a tarefa de predição de quais alunos terão um desempenho regular e quais terão um bom desempenho acadêmico na Universidade. Há de se notar que ambos os classificadores tiveram um tempo de execução considerável que a depender do tamanho da base pode consumir recursos computacionais importantes.

Outro ponto a se destacar é o bom desempenho do algoritmo *Gaussian Classifier* que, como podemos observar, na sessão de trabalhos relacionados, não é um algoritmo frequentemente utilizado para a predição de dados educacionais. Já o algoritmo *Random Forest* aparece corriqueiramente com um bom desempenho para a mesma atividade.

O algoritmo *Random Forest* também fornece a pontuação dos atributos mais relevante para a classificação. Com esses dados podemos concluir que atributos sociais e econômicos são extremamente relevantes para o desempenho dos alunos, pois entre os 5 atributos mais relevantes temos apenas características sociais e econômicas. Um fator acadêmico aparece apenas na 8ª posição que é a quantidade de horas de estudo. Podemos concluir também que o desempenho acadêmico anterior do aluno é praticamente irrelevante pois dos 5 atributos menos relevantes temos 4 fatores acadêmicos e apenas 1 social (casta).

Para trabalhos futuros recomenda-se a utilização de outras bases de dados educacionais que contenham dados socioeconômicos e acadêmicos para confirmar ou não que os algoritmos *Gaussian Classifier* e *Random Forest* apresentam o melhor desempenho.

Outra abordagem relevante é o estudo aprofundado das duas técnicas para que se possa otimizar seu desempenho aumentado a taxa de sucesso e diminuindo o tempo de execução.

Por fim, recomenda-se um o estudo em parceria com a área pedagógica e/ou social para que possa propor ações que visem aumentem o desempenho acadêmico de alunos das Universidades.

REFERÊNCIAS

ALLOGHANI, M.; AL-JUMEILY, D.; BAKER, T.; HUSSAIN, A.; MUSTAFINA, J.; ALJAAF, A.J. ***Applications of Machine Learning Techniques for Software Engineering Learning and Early Prediction of Students' Performance.*** *Soft Computing in Data Science. SCDS 2018. Communications in Computer and Information Science*, vol 937. Springer, Singapore, p. 246-258.

ANDREONI, M. **An Intrusion Detection and Prevention Architecture for Software Defined Networking**. Dissertação (Mestrado em Engenharia Elétrica) - UFRJ, Rio de Janeiro, p.11. 2014.

BONESSO, D. **Estimação dos Parâmetros do Kernel em um classificador SVM na Classificação de Imagens Hiperespectrais em uma abordagem Multiclasse**. Tese (Mestrado em Sensoriamento Remoto) - Centro Estadual de Sensoriamento Remoto e Meteorologia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2013, p. 32-33.

COUTO, E. **Bias vs. Variância**. ERICCOUTO. Disponível em: <https://ericcouto.wordpress.com/2013/07/18/bias-vs-variancia-parte-2/>. Acesso em 14 de setembro de 2019.

FERNANDES, E.; HOLANDA, M.; VICTORINO, M.; BORGES, V.; CARVALHO, R.; VAN ERVEN, G. ***Educational data mining****: Predictive analysis of academic performance of public school students in the capital of Brazil. Journal of Business Research*, v. 94, p. 335-343, 2018.

FERRETTO. **Média Harmônica.** 24 de maio de 2019. Professor Ferretto. Disponível em <https://www.professorferretto.com.br/media-harmonica/>. Acesso em 14 de setembro de 2019.

GOMES, P. **Conheça o algoritmo XGBoost**. Data Geeks. Disponívem em <https://www.datageeks.com.br/xgboost/> acesso em 12 de setembro de 2019.

HAN, J., KAMBER, M. ***Data mining:*** *concepts and techniques. Morgan Kaufmann Publishers* Inc., San Francisco, CA, USA, 2000.
HUSSAIN, S., DAHAN, N.; BA-ALWIB, F.; RIBATA, N. ***Educational Data Mining and Analysis of Students' Academic Performance Using WEKA***. *Indonesian Journal of Electrical Engineering and Computer Science*, v. 9, n. 2, p. 447-459, 2018.

KHATTREE, R; NAIK, D. **Multivariate data reduction and discrimination with SAS software**. Cary, NC, USA: SAS Institute Inc., 2000. 558 p.

KDNUGGESTS. **Random-Forest**. Disponível em: <https://www.kdnuggets.com>. Acesso em 18 de setembro de 2019.

NARKHEDE, S. **Understanding AUC - ROC Curve**. Towards Data Science. Disponível em: <https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5>. Acesso em 14 de setembro de 2019.

NUMPY. Disponível em: https://numpy.org/. Acesso em 16 de setembro de 2019.

O que é uma matriz de covariância. Minitab. Disponível em <https://support.minitab.com/pt-br/minitab/18/help-and-how-to/modeling-statistics/anova/supporting-topics/anova-statistics/what-is-the-variance-covariance-matrix/>. Acesso em 12 de setembro de 2019.

PANDAS. Disponível em: https://pandas.pydata.org/. Acesso em 16 de setembro de 2019.

PEDREGOSA, F. *et al*.. **Scikit-learn**: *Machine Learning in Python. Journal of Machine Learning Research*, v. 12, p. 2825-2830, 2011. Disponível em: <https://scikit-learn.org/stable/supervised_learning.html#supervised-learning> Acesso em 11 de setembro de 2019.

Quais são os métodos mais populares de machine learning? SAS. Disponível em: <https://www.sas.com/pt_br/insights/analytics/machine-learning.html> Acesso em: 09 de set. de 2019

RASMUSSEN, C. **Gaussian Processes in Machine Learning**. Lecture Notes in Computer Science, Springer, Berlin, v 3176, p.64, 2003.

REFAEILZADEH, P., TANG, L., LIU, H. **Cross-Validation.** Encyclopedia of Database Systems. Springer, Boston. p. 24, ed. 2009.

REGRESSÃO LOGÍSTICA NO R. Estatsite. Disponível em : <https://estatsite.com/2018/08/26/regressao-logistica-no-r/>. Acesso em: 14 de setembro de 2019.

SCHADE, G. **Machine Learning:** métricas para Modelos de Classificação. 12 de abril de 2018. Imasters. Disponível em: <https://imasters.com.br/desenvolvimento/machine-learning-metricas-para-modelos-de-classificaco>. Acesso em 14 de setembro de 2019.

SCIKIT-LEARN. Disponível em <https://scikit-learn.org/stable/>. Acesso em 16 de setembro de 2019.

SILVA, J. **Algoritmos de Aprendizagem de Máquina:** qual deles escolher? Machina Sapiens. Disponível em: <https://medium.com/machina-sapiens/algoritmos-de-aprendizagem-de-m%C3%A1quina-qual-deles-escolher-67040ad68737>. Acesso em 13 de setembro de 2019.

SILVER, NATE. **O sinal e o ruído**: porque tantas previsões falham e outras não. Rio de Janeiro: Intrínseca, 2013, p.544.

Student Academics Performance Data Set. UCI, Machine Learning Repository. Disponível em: <https://archive.ics.uci.edu/ml/datasets/Student+Academics+Performance> Acesso em 28 de janeiro de 2019.

TRAKUNPHUTTHIRAK, R; CHEUNG, Y; LEE, V. C. S. ***A Study of Educational Data Mining:*** *Evidence from a Thai University. Proceedings of the Thirty-Third AAAI Conference on* Artificial Intelligence, Palo Alto, California. v. 33, n.1, p. 734-741, 2019.

WITTEN, I. H., FRANK, E. ***Data Mining****: Practical Machine Learning Tools and Techniques*. San Francisco, CA: Morgan Kaufmann, 2005.

XGBOOST. Disponível em: https://xgboost.readthedocs.io/en/latest/build.html. Acesso em 16 de setembro de 2019.

ZIBETTI, A. **Distribuição Normal (Gaussiana) Distribuição De-Moivre-Laplace-Gauss.** Probabilidade para Engenharias utilizando o Rstudio UFSC. Disponível em: <https://www.inf.ufsc.br/~andre.zibetti/probabilidade/normal.html>. Acesso em: 16 de setembro de 2019.

ZOU, K. et. al. ***Receiver-Operating Characteristic Analysis for Evaluating Diagnostic Tests and Predictive Models*.** Ahajournals.Greenville Ave, Dallas, v. 115, n.5, p. 654–657, 2007.