Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Data: 14 de dezembro de 2018
Disciplina: Tópicos Especiais *Machine Learning* - 2018.1
Professor(a): Elloá B. Guedes
Alunos: Iêsa B. Lobato, Lucas P. Reis, Victor S. Lopes, Vitor M. de S. Carvalho
Curso: Engenharia da Computação

Mini Teste 2

O objetivo deste projeto é comparar vários modelos de aprendizagem de máquina por meio dos resultados obtidos através da métrica de desempenho F-Score aplicada a um problema de classificação multi-classe. As classes do *dataset* utilizado, *Wine Quality Dataset*, correspondem ao atributo-alvo do problema: a qualidade de vinhos verdes produzidos no norte de Portugal.

O *dataset* é dividido em dois arquivos: um deles contém informações a respeito dos vinhos tintos, e o outro, dos vinhos brancos. Foi necessário concatenar as duas partes para trabalhar com todas as informações disponíveis.

Os atributos preditores são: *fixed acidity* (acidez fixa), *volatile acidity*(acidez volátil), *citric acid*(ácido cítrico), *residual sugar* (açúcar residual), *chlorides* (cloretos), *free sulfur dioxide* (dióxido de enxofre livre), *total sulfur dioxide* (dióxido de enxofre total), *density* (densidade), pH, *sulphates* (sulfatos), e *alcohol* (álcool). O atributo-alvo é chamado de *quality* (qualidade) e medido em um intervalo de zero a dez.

Existem 6.497 exemplos no *dataset*, sendo quase 3.000 deles de qualidade entre seis e sete. É possível perceber que o *dataset* está desbalanceado pois apenas 1.600 destes são exemplos de vinhos tintos, enquanto os restantes são exemplos de vinhos brancos, ou seja, os dados estão mal distribuídos.

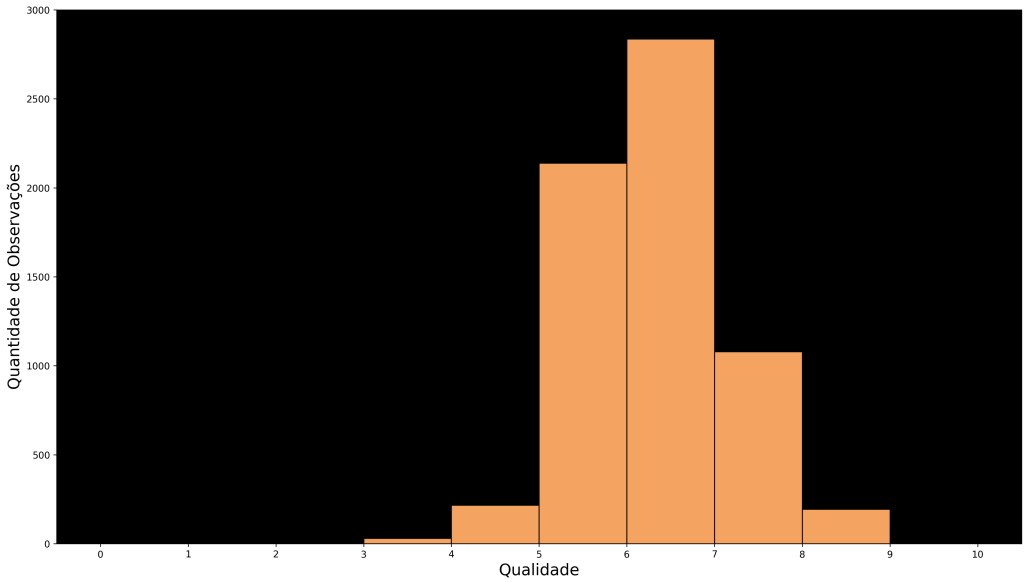

Figura 1: Histograma do atributo-alvo.

Para realizar a implementação de todos os modelos, utilizou-se a função *GridSearchCV* obtida da biblioteca *sklearn.model_selection*, que consiste em uma busca exaustiva através da combinação de parâmetros especificados por um estimador. A busca consiste em:

- Um estimador: um modelo de classificação ou regressão;
- Um dicionário de parâmetros dos modelos em questão;
- Uma função de *score*: utilizou-se F1-Score, em específico o *micro-averaging* pois o atributo-alvo não é balanceado;
- Um método de validação cruzada: utilizou-se k = 3 *folds* como método.

A respeito dos parâmetros, foram utilizados os seguintes valores para cada método:

Parâmetro	Valores
Critério	*Gini* ou Ganho de Informação
Profundidade máxima	Valores no intervalo de [1, 11]
Número máximo de atributos	Valores no intervalo de [1, 11]

Tabela 1: Parâmetros para a Árvore de Decisão e Floresta Aleatória.

Parâmetro	Valores
Algoritmo	*Gini* ou Ganho de Informação
Taxa de aprendizado	Valores no intervalo de [0.1, 1.0] com incremento de 0.1

Tabela 2: Parâmetros para o *Adaboosting*.

Parâmetro	Valores
Reinserção de amostras	Sim ou Não
Reinserção de atributos	Sim ou Não
Número máximo de atributos	Valores no intervalo de [1, 11]

Tabela 3: Parâmetros para o *Bagging*.

Após a implementação da função *GridSearchCV*, chama-se o método *fit* para treinar o modelo. Em seguida, temos o atributo *cv_results_*, um dicionário onde cada linha é um modelo diferente, a quantidade total depende da quantidade de combinação de parâmetros. Por fim, transformou-se este dicionário em um *DataFrame*. Tomando-se o melhor modelo para cada um dos métodos, gerou-se a seguinte tabela ordenada de acordo com o maior F-Score médio.

Modelos	F-Score
Bagging	0.635832
Florestas Aleatórias	0.567954
Árvore de Decisão	0.566107
Adaboosting	0.514083

Tabela 4: Comparativo entre os melhores modelos para cada método.

Como pode ser visto, o modelo baseado em *Bagging* obteve o melhor F-Score, tal modelo considerou sete como quantidade máxima de atributos, com reinserção de amostras e atributos . A seguir o seu tempo necessário para treinamento e teste, junto com sua matriz de confusão para um dos seus *folds*.

Modelo	Tempo médio de treinamento(s)	Tempo médio de teste(s)
Bagging	0.184687	0.005209

Tabela 5: Dados do melhor modelo.

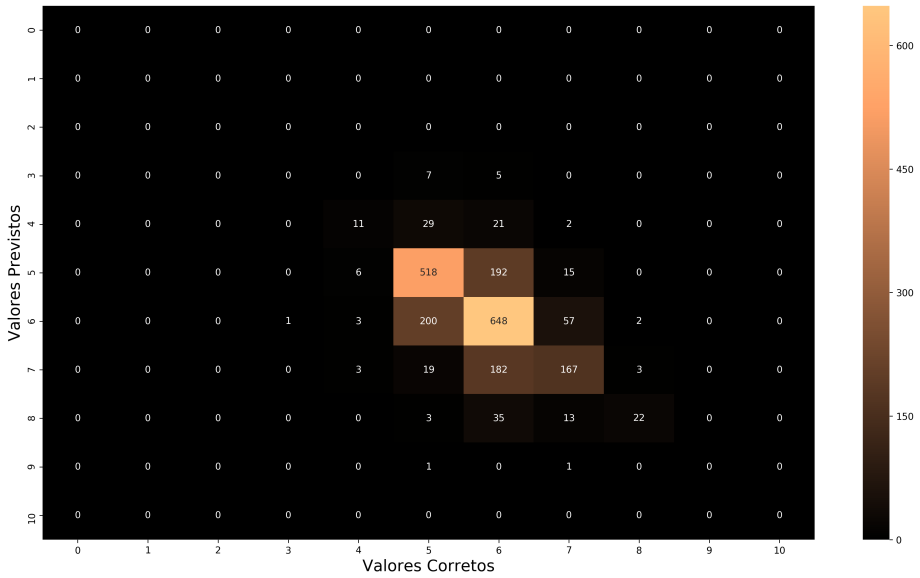

Figura 2: Matriz de Confusão para o melhor modelo.

O modelo Bagging é recomendado para problemas em que a variância dos resultados não é alta. Por uma análise dos seus procedimentos, percebe-se a ocorrência deste fato, pois o modelo melhora sua acurácia por utilizar multiplas versões de um conjunto de treinamento, cada um criado aleatoriamente, com reposições, que são utilizadas para treinar cada um de seus classificadores. Tendo, portanto, uma resposta final mais precisa, baseada no resultado de todos os seus classificadores.