Classification of Wine Quality

Anonymous Authors

January 10, 2019

1 Introduction

The following report outlines our attempts to use common classification algorithms to predict the wine quality depending on the given features. We use a data set from the UCI Machine Learning Repository [1] [**data**].

1.1 Data Description

The data set contains of 4898 data points, each of them described by 11 features and an integer quality label ranging from 3 to 9. Note that the distribution of the quality is unbalanced: More than 90% of the data points are assigned to a quality between 5 and 7 (Figure 1).

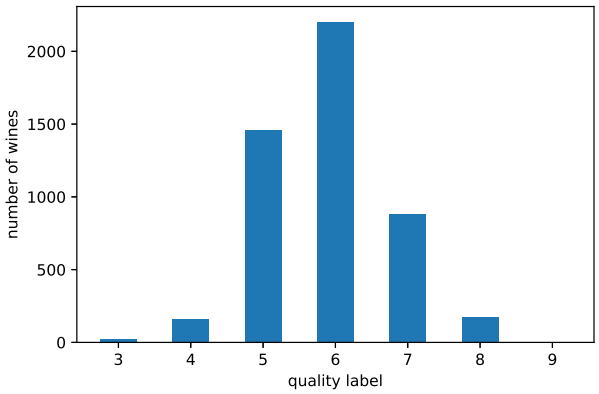

Figure 1: distribution of quality

1.2 Data preprocessing

To simplify our problem we assigned new class labels 0, 1 and 2. We assign class 0 (bad) to wines ranging in qualities 3-5, class 1 (medium) to wines labeled with quality 6 and class 2 (good) to wines with quality greater than 6. We further decided to use classification instead of regression. Though, it should be noted that we also examine algorithms (e.g. ridge classifier) that use regression to classify the data points and using regression should therefore yield similar results in our case. This doesn't come as a surprise, considering that the classes can obviously be linearly ordered.

1.3 Cross-Validating the data

The goal of cross validation is to achieve the accuracy of out-of-training data with using only the training data itself. So we don't have to separate the data into two different groups, instead we train the model on all of the data set.

In out project we use the cross_val_score helper function to calculate the mean and the accuracy[2] of the different models.

2 Ridge Classification

2.1 Description

Ridge Classifier implements Ridge regression. Ridge regression (one of it's many names) is a linear regularization method [3] that addresses some of the Ordinary Least Squares problems [4] in which if there is no unique answer the estimator will lead to over fitting or under fitting and the result will not be as accurate as by using other methods.

In our project we have implemented the RidgeClassifier function from the Scikit-learn library.

2.2 Cross-Validation

The resulted mean is 0.55655979654374510002, and the resulted accuracy is 0.56 (+/- 0.08).

[1] `http://archive.ics.uci.edu/ml/datasets/Wine+Quality`

[2] `https://scikit-learn.org/stable/modules/cross_validation.html#computing-cross-validated-metrics`

[3] `https://en.wikipedia.org/wiki/Tikhonov_regularization`

[4] `https://scikit-learn.org/stable/modules/linear_model.html#ridge-regression`

2.3 Results

3 Multi Layer Perceptron

3.1 Description

Multi Layer Perceptron (MLP) is a supervised learning algorithm and is a function approximator that can be used for classification and regression which is widely used in neural network algorithms [5]. In this structure, a single neuron is represented by a single perceptron and when considering many of them, they represent the entire neural network. In this method, multiple layers (two or more with one or more additional hidden layers) and non-linear methods are used.One layer is simply one row of neurons. A simple MLP with only one hidden layer is represented in Figure 2 (Figure credit [6]).

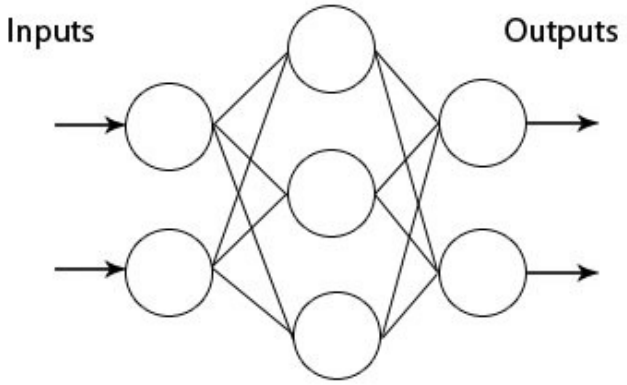

Figure 2: Simple MLP with one hidden layer

We have used the MLPClassifier function (rather than MLPRegressor although both could be used on our chosen data set) from Scikit-learn's Neural Network library.

3.2 Cross-Validation

Result = 0.5347151403973234 The resulted mean is 0.51490775676971511920, and the resulted accuracy is 0.51 (+/- 0.06).

[5] https://machinelearningmastery.com/neural-networks-crash-course/

[6] http://neuroph.sourceforge.net/tutorials/MultiLayerPerceptron.html

3.3 Results

4 Random Forest Classification

4.1 Description

Random Forest is also a supervised learning algorithm that is widely used for solving classification and regression problems. The idea behind this method is to increase the classification accuracy (a.k.a aim for lower variance) by using bootstrap aggregating (or bagging) algorithms. Generally speaking, the gross idea behind Random Forest Classification is to use the bagging method to split up a decision tree into two different trees. When one does that enough times, there is a large amount of trees (hence the name- "forest) which gives the possibility to gather more information for analysis and therefore better accuracy. [7].

In this project we used the RandomForestClassifier algorithm from the Scikit-learn library.

4.2 Cross-Validation

The resulted mean is 0.57390360843009313729, and the resulted accuracy is 0.57 (+/- 0.05).

4.3 Results

5 Gaussian Process Classification

5.1 Description

Gaussian Processes algorithms could be also used for both classification and regression problems. A Gaussian process is a stochastic process with a Gaussian normal distribution kernel. This type of algorithm takes all the points that are available and measures the similarity between them to predict a value for a value for a future point. The principle behind this classification method is the assumption that the tested data is normally distributed so this method should, in theory, work poorly on data that is not distributed in such a matter. In practice it could be observed that Gaussian processes work rather good on most data and the performance is not so accurate when the data set contain more than a dozen or a couple dozen of features. [8]. Luckily, our wine quality data set has only 11 features, which should reproduce satisfactory results.

[7] https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd

[8] https://www.kaggle.com/residentmario/gaussian-process-regression-and-classification

To implement the Gaussian Process Classification algorithm, we used the GaussianProcessClassifier method from the Scikit-learn library.

5.2 Cross-Validation

The resulted mean is 0.44876050113610305159, and the resulted accuracy is 0.45 (+/- 0.06).

5.3 Results

6 Comparison

7 Conclusion