ECEN 689 Challenge 4, Team 12

Shabarish Rajendra, Sayeed Alvi, and Fazia Batool

Abstract

This Challenge is based on the Wine Quality Data, which aggregates objective tests about various wines. The output is based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded wine quality between 0 (bad) and 10 (excellent). Two datasets are included, related to red and white vinho verde wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.).

The goal of this activity is to explore a linear regression to predict wine quality and to explore a decision tree to classify the type of wine (red or white).

I. Introduction

THE two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are munch more normal wines than excellent or poor ones). Outlier detection algorithms could be used to detect the few excellent or poor wines. Also, we are not sure if all input variables are relevant. So it could be interesting to test feature selection methods. [1].

Input variables (based on physicochemical tests):
1 - Fixed acidity
2 - Volatile acidity
3 - Citric acid
4 - Residual sugar
5 - Chlorides
6 - Free sulfur dioxide
7 - Total sulfur dioxide
8 - Density
9 - pH
10 - Sulphates
11 - Alcohol

Output variable (based on sensory data):
12 - Quality (score between 0 and 10)
Mathematically, wine quality is estimated based on a linear combinations of the input features,

$$\hat{y}_i = \alpha_0 + \sum_j xi, j \tag{1}$$

The coefficients $\{\alpha_i\}$ should be the same for every wine.

II. Methodology

Linear regression is a statistical approach to modelling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). In this framework, the relationships between input and output are modeled using linear predictor functions whose unknown model parameters are estimated from the data.

Gradient Descent is the process of minimizing a function by following the gradients of the cost function. This involves knowing the form of the cost as well as the derivative so that from a given point you know the gradient and can move in that direction, e.g. downhill towards the minimum value. Linear regression does provide a useful exercise for learning stochastic gradient descent which is an important algorithm used for minimizing cost functions by machine learning algorithms.

In Machine learning we can use a similar technique called stochastic gradient descent to minimize the error of a model on our training data.The way this works is that each training instance is shown to the model one at a time. The model makes a prediction for a training instance, the error is calculated and the model is updated in order to reduce the error for the next prediction.This procedure can be used to find the set of coefficients in a model that result in the smallest error for the model on the training data. Each iteration the coefficients, called weights (w) in machine learning language are updated using the equation:

w = w alpha * delta ; where w is the coefficient or weight being optimized, alpha is a learning rate that you must configure (e.g. 0.1) and gradient is the error for the model on the training data attributed to the weight.

A **decision tree** is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. **Random Forests** are an ensemble of k untrained Decision Trees (trees with only a root node) with M bootstrap samples (k and M do not have to be the same) trained using a variant of the random subspace method or feature bagging method. Note the method of training random forests is not quite as straightforward as applying bagging to a bunch of individual decision trees and then simply aggregating the output. The procedure for training a random forest is as follows:

1. At the current node, randomly select p features from available features D. The number of features p is usually much smaller than the total number of features D.

2. Compute the best split point for tree k using the specified splitting metric (Gini Impurity, Information Gain, etc.) and split the current node into daughter nodes and reduce the number of features D from this node on.

3. Repeat steps 1 to 2 until either a maximum tree depth l has been reached or the splitting metric reaches some extrema.

4. Repeat steps 1 to 3 for each tree k in the forest. Vote or aggregate on the output of each tree in the forest.

Compared with single decision trees, random forests split by selecting multiple feature variables instead of single features variables at each split point. Intuitively, the variable selection properties of decision trees can be drastically improved using this feature bagging procedure. Typically, the number of trees k is large, on the order of hundreds to thousands for large datasets with many features.

III. Results and Discussion

We are able to achieve RMSE of around 0.75 on the white wine testing data, and upon reusing the same model on the red wine test data, we achieve RMSE of around 0.77, which is really good considering feature reuse. This means that our model has done a good job on generalizing the white wine dataset. We have also observed that while performing gradient descent on the red wine training data we are able to achieve RMSE of around 0.65. and using that model on the white wine testing data yields a RMSE of 0.81, which means training the data on the red wine training data very slightly overfits the red wine data, whereas the model obtained by training on the white wine training data generalized the entire wine dataset better.

Furthermore, on training a classifier to distinguish red and white wines, we are able to achieve around 90 percent accuracy on the test data. This implies that the datasets of white and red wines have some uniqueness in their features that distinguish them to their own groups. So, if the white wine model were to overfit the white wine data, you would have a really poor performance (an RMSE of greater than 1) on the red wine testing data. But we find that we are achieving an almost similar performance on both the testing data. This implies that the model is really well generalized.

IV. Conclusions

There are several key insights from this analysis.Upon reusing the model, we are able to achieve good performances on both the datasets. This means that, the model has generalized the dataset well. And since the both the datasets are fairly distinguishable, this only solidifies the fact that the model is well generalized.

V. REFERENCES

1. P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. ”Modeling wine preferences by data mining from physicochemical properties”,In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

2. Wine quality Data, ”https://archive.ics.uci.edu/ml/datasets/Wine+Quality”