Implementation of the Random Forest Algorithm on Different Domains

Yiming Liu, Yi Ren

University of Southern California, Los Angeles, CA 90007
{liuyimin, yiren}@usc.edu

Abstract. Random forest is one of the most powerful machine-learning classification and regression. In this project, we implement the random forest classifier to get a further understanding the algorithm. In order to learn the approach of achieving better performance, we test it on different datasets, adjust different parameters to tune the random forest, and explore methods of improving the accuracy of the results. First, we predict the sex of abalone. Second, we use the algorithm to predict the category of iris. Third, we figure out the quality of wines based on several features. Finally, we apply random forest to classify a broad game dataset, which records some features of the game.

Keywords: random forest, machine learning, classification

1 Introduction

Random forest algorithm is a highly versatile machine learning method with numerous applications. It can handle a large number of features. Before the random forest, we want to introduce about the decision tree. Then, we would describe how a random forest works. Finally, we give four examples about how the random forest solves the problem in real world. By analyzing different versions of predictions and scores, we try to fit data into the best model and do further observations of the random forest learning strategy.

2 Decision Tree and Random Forest

(1) Decision tree

Decision tree is a tree data structure in computer science. It can be used as a decision tool to classify data. Just like other tree data structure, the decision tree has a root, and several nodes. Each edge in the tree is a judgment to decide which child should go next. When the condition arrive the leaf, you can get a result.

(2) Random Forest

Random Forest is a method for classification. Multiple decision trees construct the random forest in the training phase. For example, the data set has N instances, and M features. When we implement the algorithm, first, we need to decide the number of attributes m, m should smaller than M; second, get N samples for training data by get and return method, which means get a bootstrap sample; third, use the data that has not been gotten to evaluate; forth, for each node, random pick up m features, and calculate the best divide way; finally, pruning.
In our implementation of the random forest, the input is the training data and testing data, and the output is the accuracy of the test data.

(3) Introduction of the code
The core algorithm is composed by forest.py, tree.py and tools.py, which built by the class RandomForestClassifier. The main arguments which we may need to tune for better performance are as follows: n_estimators: The number of decision trees in the random forest.
max_features: Controls the number of features to randomly consider at each split.
max_depth: The maximum number of levels that the tree can grow downwards before forcefully becoming a leaf.
min_samples_split: The minimum number of samples needed at a node to justify a new node split.
bootstrap: A part of randomly chosen data to fit each tree on.

The main functions are described as follows:
Fit(self,X,y): Generate a forest by a random subset of data and features.
Predict (self, X): Predict the class of each sample in X.
Score (self, X, y): Return the accuracy of the prediction of X compared to y.

3 Using Random forest to predict the sex of abalone.

(1). We find a data set of abalone [1]. The data has 4177 instances, each instance include the attribute of Sex, Length, Diameter, Height, Whole weight, Shucked weight, Viscera weight, Shell weight, Rings. What we want to know here is whether we can predict the sex of an abalone if we have the parameter of the abalone.

(2). In the project, the y is the first attribute (sex), and the x is the other attribute. In the training phase, we want to get a model from training data. This model should contain enough information to classify the test data.

(3). Result: the result of the algorithm is not as good as we expect, which means it is very difficult for us to decide the sex of an abalone if we only have attribute like those. When we use 10 decision trees and the max depth of each tree is 10, the accu-

racy is 51.57%. And we use 50 decision trees and the max depth of each tree is 10, the accuracy is 52.91%.

4 Using Random forest to predict the category of iris

(1). This algorithm is also suitable for predict the category of iris. In the UCI Machine Learning Repository, there is a data set of iris [2]. The data has 150 instances, each instance include the attribute of sepal length, sepal width, petal length and petal width. There are three classes of Iris: Setosa, Versicolour and Virginica. What we want to know here is whether we can predict the category of Iris if we have the four parameters of an iris.

(2). In the project, the target y is the class (Setosa, Versicolour and Virginica), and the input x is the width or length.

(3). Result: The result of the algorithm is very good, which means it is very easy for us to decide the class of an iris if we only have attribute like those. When we use 10 decision trees and the max depth of each tree is 10, the accuracy is 97.37%.

5 Using Random forest to predict the quality of wines

(1). This algorithm is also suitable for predict the quality of wines. In the UCI Machine Learning Repository, there is a data set of wine quality [3]. The data has 4898 instances, each instance include the attribute of fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, soleplates alcohol and quality. The quality of wine is from 0 to 10. And there are two sets, one is for red wine and one is for white wine. What we want to know here is whether we can predict the quality of wines if we have those features.

(2). In the project, the target y is the quality, and the input x is the other variables.

(3). Result: The result of the algorithm is not very good, which means it is not very easy for us to decide the predict the quality of wine if we only have attribute like those. When we use 10 decision trees and the max depth of each tree is 10, the accuracy is 61.25%. When we use 30 decision trees and the max depth of each tree is 10, the accuracy is 64.75%. When we use 300 decision trees and the max depth of each tree is 20, the accuracy is 66.75%.

6 Using Random forest to predict the average rating of games

(1). This algorithm is also suitable for predict the average rating of games. The data set is coming from BoardGameGeek [4]. The data has 81312 instances, each instance

include the attribute of id, type, name, yearpubliced, minplayers, maxplayers, playingtime, minplaytime, maxplaytime, minage, users_rated, average_rating, bayes_average_rating, total_owners, total_weights and average_weight. The target column is average_rating with the range from 0 to 10. What we want to know here is whether we can predict the average_rating of wines if we have those features.

(2). In the project, the y is the unit digit of average_rating, and the x is the other variables except for bayes_average_rating, type and name because the bayes_average_rating is similar to average_rating.

(3). Result: The result of the algorithm is very bad, which means it is very hard for us to predict the average_rating of games if we only have attribute like those. When we use 10 decision trees and the max depth of each tree is 10, the accuracy is 34.2%. When we use 30 decision trees and the max depth of each tree is 10, the accuracy is 39.75%. When we use 300 decision trees and the max depth of each tree is 20, the accuracy is 40.1%.

7 Analysis of the result:

As the results show, the random forest classifier model gives different performances on different datasets.

What we analysis is that: if the attributes are objective and describe the all aspect of the statue of the theme, then the result might be good. For example, the iris data is objective, and it covers all the aspect of the iris, so the accuracy is high. For the wine quality, different people have different views, so although the data is objective, but the result is not as good as we expect. For the abalone, although the data is objective, but it is not cover all the aspect or it is not cover the main features of difference between male and female abalone.

For the game rating, the opinions on the quality of a broad game differ from person to person. Even the playtime can influence the rating, but it is not main part. We can assume that the most import of a game is the logic of the game. If the game is interesting, the players must be attractive by its content. After using “corr” method from Pandas library to get the correlations for the average_rating column, we see that the average_weight and id columns correlate best to rating. However, the best correlation score is about 0.351which means these attributes are not tightly related to the target.

8 References

1. UCI data repository. Abalone. http://archive.ics.uci.edu/ml/datasets/Abalone?pagewanted=all
2. UCI data repository. Iris. https://archive.ics.uci.edu/ml/datasets/Iris
3. UCI data repository. Wine quality. https://archive.ics.uci.edu/ml/datasets/Wine+Quality
4. BoardgameGeek. http://www.boardgamegeek.com/

Division of the project:
Both of the members completed and optimized the code together and complete report together.