Albert Shevchuk

Dragutin Petkovic

CSC – 699 Artificial Intelligence

March 23, 2019

Applying Random Forest for Classification

This report will be evaluating Random Forest’s accuracy using a library function named Scikit from the programing language Python. We will be using Sabiha Barlaskar’s “Applying Improved Random Forest Explainability (RFEX 2.0) steps on synthetic data for variable features having a unimodal distribution” report as a basis to follow Step One on page 8 that states:

1. Apply Random Forest (RF) to establish base RF accuracy using all features of the data set for classification.

The synthetic test data set will also be used from Sabiha’s report referenced on page 15 with the link:

<https://drive.google.com/file/d/1KKZ6iK_0aBAesGZgwYnORHLPtGtfNXLM/view?usp=sharing>

The main objective of this exercise will be for learning and experimental purposes. The report will contain a brief outline of the steps and processes that are being executed. The report will evaluate the difference in randomized data for Random Forest’s (RF) Classifier. Currently the CSV data is sorted by the target Row with the first group 1. The list of testing will go as followed:

1. RF’s Classifier on CSV data “as is” with no randomizing of any kind.

2. RF’s Classifier on CSV data after randomizing only the CSV file using Microsoft Excel.

3. RF’s Classifier on CSV data and only randomizing while running RF with python.

4. RF’s Classifier on CSV data both randomizing while running RF with python and with Miscrosoft Excel before the test.

**INTRO**

First off we will be using Jupyter Notebook as the Integrated Development Environment of choice running on a 64bit version of Windows 7. For each item above the same set of commands will apply.

1. We will first import a python data analysis library pandas for data manipulation.
2. Using pandas we will read the csv file and store it into a variable named dataset
3. The data is going to be split into label features and attributes. We are going to use pandas .iloc to extract columns “F1 – F10” as features and column “target” as the attribute.
4. Import train\_test\_split, a quick utility that wraps input validation and splits the data to train and test.
5. We create 4 variables x\_train, x\_test, y\_train, y\_test and divide the data 44.2% to train and the rest to test. The reason behind choosing .442 as the splitting parameter is because at the 442nd row the target data is changed from a 1 to a 0. So we are basically splitting the train/test data to match the target results. (Note: this is specifically made to test #1 out of the list indicated.
6. Import RandomForestClassifier witch is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
7. We will apply the RandomForestClassifier to grow 100 trees and store the data into a variable named classifier.
8. Using fit will build the forest of trees from the tanning set(x, y).
9. Using the class predict() we can sample a vote by the trees iun the forest by their probability estimates.
10. Lastly import the necessary libraries to print out the confusion matrix, classification report, accuracy score and the out of bag score.

**TESTING**

1. In the first test sample business as usual and none of the data is randomized. We use the test size .442 to split the target group 1 from 0.

import pandas as pd

dataset = pd.read\_csv('C:/Users/Albert/Downloads/Synthetic\_data-variable\_values\_for\_features.csv')

x = dataset.iloc[16:1015, 0:9].values

y = dataset.iloc[16:1015, 10].values

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = .442)

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators = 100, oob\_score = True)

classifier.fit(x\_train, y\_train)

y\_pred = classifier.predict(x\_test)

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

print(confusion\_matrix(y\_test,y\_pred))

print(classification\_report(y\_test,y\_pred))

print('Accuracy Score: ', accuracy\_score(y\_test, y\_pred))

print('OOB Score: ',classifier.oob\_score\_)

[[258 0]

[ 8 176]]

precision recall f1-score support

0 0.97 1.00 0.98 258

1 1.00 0.96 0.98 184

micro avg 0.98 0.98 0.98 442

macro avg 0.98 0.98 0.98 442

weighted avg 0.98 0.98 0.98 442

Accuracy Score: 0.9819004524886877

OOB Score: 0.9892086330935251

1. For the second test sample the data is randomized using Microsoft excel spreadsheet. The data is randomized by creating random variables (RAND()) in an empty column next to target for all 1000 values then sorting a-z for all the rows based on the random values column. The test size is still .442.

import pandas as pd

dataset = pd.read\_csv('C:/Users/Albert/Downloads/Synthetic\_data-variable\_values\_for\_features-Rand().csv')

x = dataset.iloc[16:1015, 0:9].values

y = dataset.iloc[16:1015, 10].values

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = .442)

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators = 100, oob\_score = True)

classifier.fit(x\_train, y\_train)

y\_pred = classifier.predict(x\_test)

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

print(confusion\_matrix(y\_test,y\_pred))

print(classification\_report(y\_test,y\_pred))

print('Accuracy Score: ', accuracy\_score(y\_test, y\_pred))

print('OOB Score: ',classifier.oob\_score\_)

[[249 0]

[ 8 185]]

precision recall f1-score support

0 0.97 1.00 0.98 249

1 1.00 0.96 0.98 193

micro avg 0.98 0.98 0.98 442

macro avg 0.98 0.98 0.98 442

weighted avg 0.98 0.98 0.98 442

Accuracy Score: 0.9819004524886877

OOB Score: 0.987410071942446

1. For the third data sample the data will be randomized by the train\_test\_split library with the parameter called random\_state = 34 witch is the seed used by the random number generator.

import pandas as pd

dataset = pd.read\_csv('C:/Users/Albert/Downloads/Synthetic\_data-variable\_values\_for\_features.csv')

x = dataset.iloc[16:1015, 0:9].values

y = dataset.iloc[16:1015, 10].values

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = .442, random\_state = 34)

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators = 100, oob\_score = True)

classifier.fit(x\_train, y\_train)

y\_pred = classifier.predict(x\_test)

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

print(confusion\_matrix(y\_test,y\_pred))

print(classification\_report(y\_test,y\_pred))

print('Accuracy Score: ', accuracy\_score(y\_test, y\_pred))

print('OOB Score: ',classifier.oob\_score\_)

[[260 0]

[ 4 178]]

precision recall f1-score support

0 0.98 1.00 0.99 260

1 1.00 0.98 0.99 182

micro avg 0.99 0.99 0.99 442

macro avg 0.99 0.99 0.99 442

weighted avg 0.99 0.99 0.99 442

Accuracy Score: 0.9909502262443439

OOB Score: 0.9820143884892086

1. In the last test sample a combination of test sample 2 and 3 will be combined, having both the random number generator randomize the randomized data that’s fed into random forest.

import pandas as pd

dataset = pd.read\_csv('C:/Users/Albert/Downloads/Synthetic\_data-variable\_values\_for\_features-Rand().csv')

x = dataset.iloc[16:1015, 0:9].values

y = dataset.iloc[16:1015, 10].values

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = .442, random\_state= 34)

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators = 100, oob\_score = True)

classifier.fit(x\_train, y\_train)

y\_pred = classifier.predict(x\_test)

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

print(confusion\_matrix(y\_test,y\_pred))

print(classification\_report(y\_test,y\_pred))

print('Accuracy Score: ', accuracy\_score(y\_test, y\_pred))

print('OOB Score: ',classifier.oob\_score\_

[[257 0]

[ 6 179]]

precision recall f1-score support

0 0.98 1.00 0.99 257

1 1.00 0.97 0.98 185

micro avg 0.99 0.99 0.99 442

macro avg 0.99 0.98 0.99 442

weighted avg 0.99 0.99 0.99 442

Accuracy Score: 0.9864253393665159

OOB Score: 0.9838129496402878