CSC6515 – Machine Learning for Big Data

Assignment 1

Task a - Decision Tree Classifier Code

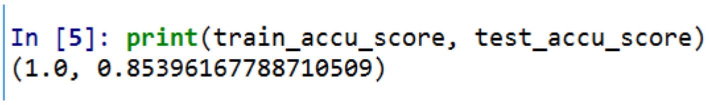
The code is self-explanatory, please read the comments on code

```
# -*- coding: utf-8 -*-
"""
Created on Sun Oct 16 17:47:58 2016
@author: Yamuna
"""

import pandas as pd
import os
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeClassifier
import sklearn.metrics

# Opening path where dataset is located
os.chdir("C:\\Users\\Yamuna\\Desktop\\Big Data\\")

# Read dataset
in_data = pd.read_csv("satellite.csv")

# Cleaning dataset
data_clean = in_data.dropna()
data_clean.dtypes
data_clean.describe()

# Defining predictors
predictors =
data_clean[['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z',
'AA','AB','AC','AD','AE','AF','AG','AH','AI','AJ']]

# Defining target data
targets = data_clean.AK

# Dividing train and test data
train_data, test_data, target_train, target_test = train_test_split(predictors,targets,test_size=.3)

# Shaping data
train_data.shape
test_data.shape
target_train.shape
target_test.shape

# Initialising Decision tree classifier
classifier=DecisionTreeClassifier()
classifier=classifier.fit(train_data,target_train)

# Predicting test and train data
test_predictions=classifier.predict(test_data)
train_predictions=classifier.predict(train_data)
```

```
#Train data confusion matrix and accuracy score
train_confu_mat = sklearn.metrics.confusion_matrix(target_train, train_predictions)
train_accu_score = sklearn.metrics.accuracy_score(target_train, train_predictions)

#Test data confusion matrix and accuracy score
test_confu_mat = sklearn.metrics.confusion_matrix(target_test, test_predictions)
test_accu_score = sklearn.metrics.accuracy_score(target_test, test_predictions)

#Visualising Decision Tree
from sklearn import tree
from io import BytesIO as StringIO
from IPython.display import Image
import pydotplus

out = StringIO()
tree.export_graphviz(classifier, out_file = out)
graph=pydotplus.graph_from_dot_data(out.getvalue())
Image(graph.create_png())
graph.write_pdf("C:\\Users\\Yamuna\\Desktop\\Big Data\\tree.pdf")
```

Output and Explanation

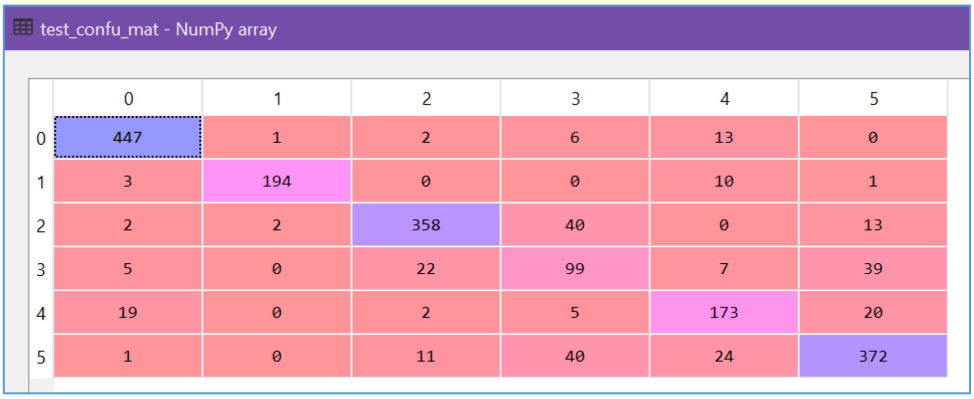
Accuracy scores



```
In [5]: print(train_accu_score, test_accu_score)
(1.0, 0.85396167788710509)
```

The train accuracy score is 100% and is larger than the test accuracy score (85% approx.). This is because, classifier was trained using the train data and hence the classification done on train data is accurate. The test data is slightly different and so the accuracy is not 100%

Test data Confusion Matrix

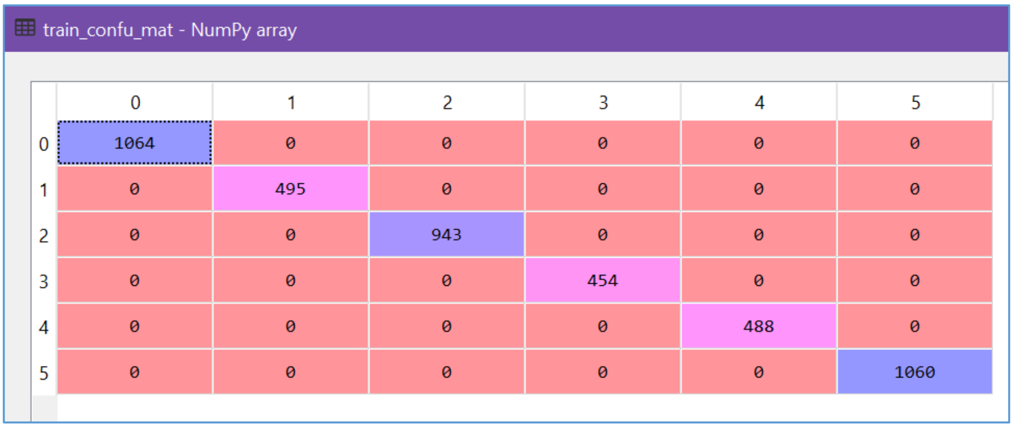


test_confu_mat - NumPy array

	0	1	2	3	4	5
0	447	1	2	6	13	0
1	3	194	0	0	10	1
2	2	2	358	40	0	13
3	5	0	22	99	7	39
4	19	0	2	5	173	20
5	1	0	11	40	24	372

This matrix above shows the Actual class along the Y axis and the Predicted class along the X axis. In the above confusion matrix of test data, 447 predictions are correctly predicted under class 0. Three predictions are done wrongly as class 0 which actually belongs to class 1 and so on. The predictions that are diagonal are the correctly predicted values for that class.

Train data Confusion Matrix

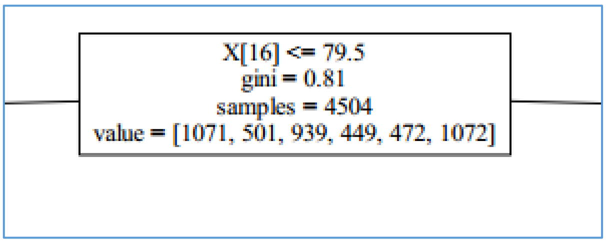

train_confu_mat - NumPy array

	0	1	2	3	4	5
0	1064	0	0	0	0	0
1	0	495	0	0	0	0
2	0	0	943	0	0	0
3	0	0	0	454	0	0
4	0	0	0	0	488	0
5	0	0	0	0	0	1060

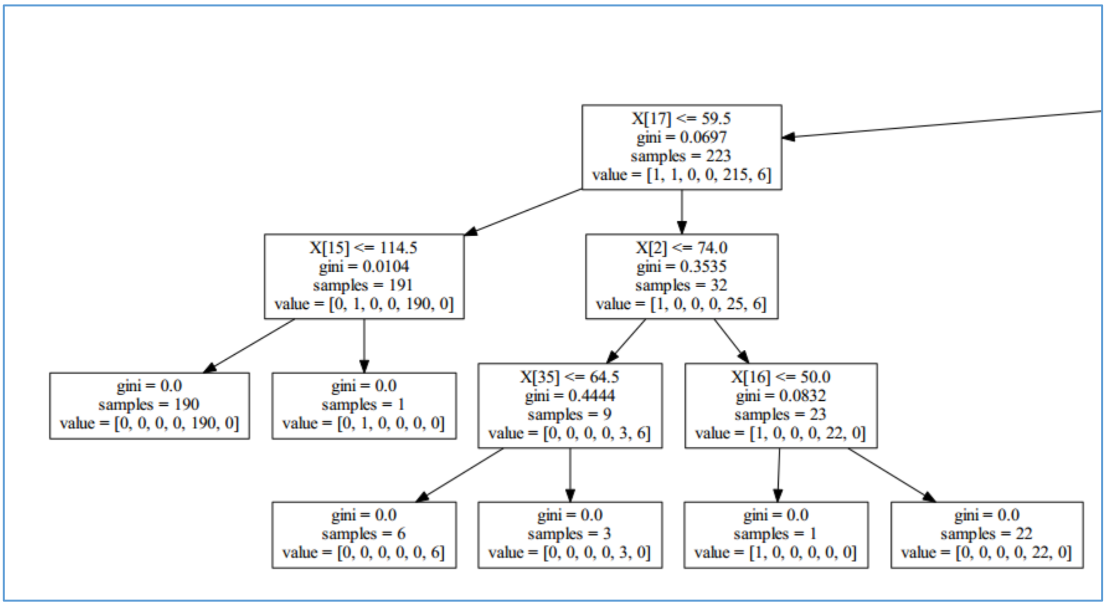
The confusion matrix for train data is very accurate and none of the classes are wrongly predicted. Since, the decision tree classifier was trained with the train data. And so is the accuracy.

Tree Visualization – Tree.pdf is attached with this submission which has the entire decision tree

Root



Sample Branch



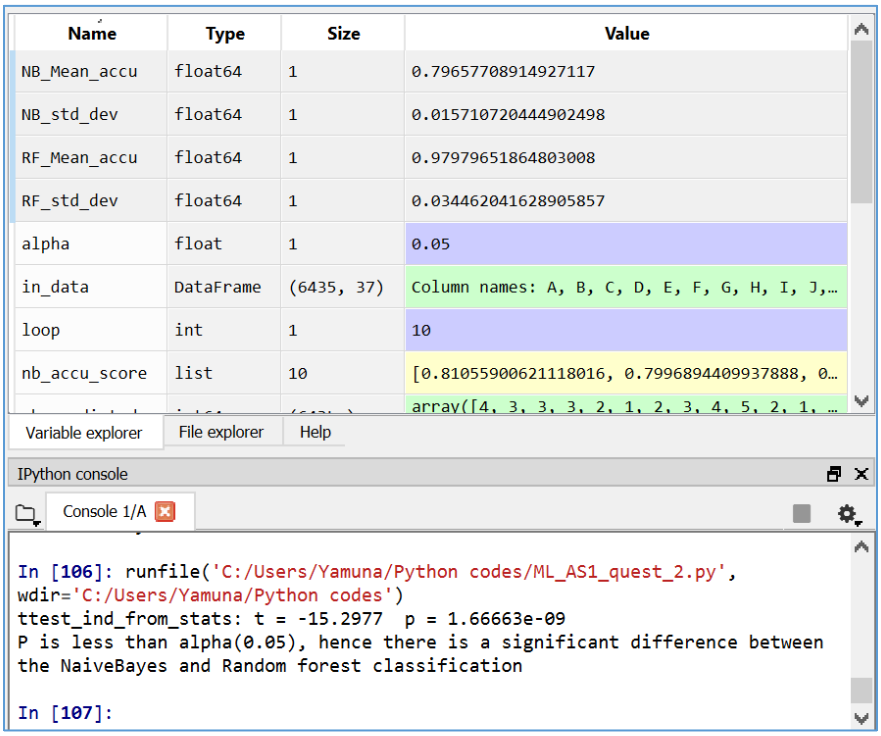
Task b – Naïve Bayes and Random Forest classifier code

```
# -*- coding: utf-8 -*-
"""
Created on Sun Oct 23 01:29:00 2016
@author: Yamuna
"""
```

```
import pandas as pd
import numpy as np
import os
import sklearn.metrics
from sklearn.ensemble import RandomForestClassifier
from sklearn import cross_validation
from sklearn.naive_bayes import GaussianNB

os.chdir("C:\\Users\\Yamuna\\Desktop\\Big Data\\")
#importing given dataset
in_data = pd.read_csv("satellite.csv")

# Initialising 10 fold cross validation
kf_total = cross_validation.KFold(len(in_data), n_folds=10, shuffle=False, random_state=None)

# Initialising Random Forest Classifier
rf_classifier=RandomForestClassifier(n_estimators=10,
                        max_features='auto',
                        max_depth=None,
                        min_samples_split=2,
                        random_state=0)

# Initialising Naive Bayes Classifier
nb_classifier = GaussianNB()

# Calculating accuracy scoreof naive bayes and Random forest for 10 folds
nb_accu_score=[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
rf_accu_score=[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
loop=0

# Looping for each fold
for train_index, test_index in kf_total:
    # Training the Naive Bayes Classifier for each fold with train data and train result
    nb_classifier.fit(np.array(in_data.ix[train_index[0]:train_index[len(train_index)-1],:36]),
np.array(in_data.ix[train_index[0]:train_index[len(train_index)-1],36:37]))
    # Predicting the test data using the trained Naive Bayes Classifier
    nb_predicted = nb_classifier.predict(np.array(in_data.ix[test_index[0]:test_index[len(test_index)-
1],:36]))
    # Calculating accuracy of Naive Bayes Classifier for each fold and storing it in an array
    nb_accu_score[loop] =
sklearn.metrics.accuracy_score(nb_predicted,np.array(in_data.ix[test_index[0]:test_index[len(test_i
ndex)-1],36:37]))
    # Training the Random Forest Classifier for each fold with train data and train result
    rf_classifier.fit(np.array(in_data.ix[train_index[0]:train_index[len(train_index)-1],:36]),
(np.array(in_data.ix[train_index[0]:train_index[len(train_index)-1],36:37])).ravel())
    # Predicting the test data using the trained Random Forest Classifier
    rf_predicted=rf_classifier.predict(np.array(in_data.ix[test_index[0]:test_index[len(test_index)-
1],:36]))
    # Calculating accuracy of Random Forest Classifier for each fold and storing it in an array
    rf_accu_score[loop] =
sklearn.metrics.accuracy_score(rf_predicted,np.array(in_data.ix[test_index[0]:test_index[len(test_in
dex)-1],36:37]))
    loop=loop+1
```

```
# Calculating Mean accuracy for NaiveBayes Classifier
NB_Mean_accu = np.mean(nb_accu_score)
# Calculating Mean Accuracy for Random Forest Classifier
RF_Mean_accu = np.mean(rf_accu_score)
# Calculating Standard deviation of NaiveBayes classifier accuracy
NB_std_dev=np.std(nb_accu_score, dtype=np.float64)
# Calculating Standard deviation of Random Forest Classifier accuracy
RF_std_dev=np.std(rf_accu_score, dtype=np.float64)

from scipy import stats
# Given alpha value in question
alpha=0.05

# Calculating statistical test for Mean and standard deviation of accuracy scores in NaiveBayes and
Random Forest
t, p = stats.ttest_ind_from_stats(NB_Mean_accu, NB_std_dev, 10,
                          RF_Mean_accu, RF_std_dev, 10,
                          equal_var=False)
print("ttest_ind_from_stats: t = %g  p = %g" % (t, p))

# Determinig the Statistical significance using ttest and alpha
if (p<alpha):
    print("P is less than alpha(0.05), hence there is a significant difference between the NaiveBayes
and Random forest classification")
else:
    print("There is no significant difference between the NaiveBayes and Random forest
classification")
```

Output and Explanation



Name	Type	Size	Value
NB_Mean_accu	float64	1	0.79657708914927117
NB_std_dev	float64	1	0.015710720444902498
RF_Mean_accu	float64	1	0.97979651864803008
RF_std_dev	float64	1	0.034462041628905857
alpha	float	1	0.05
in_data	DataFrame	(6435, 37)	Column names: A, B, C, D, E, F, G, H, I, J,…
loop	int	1	10
nb_accu_score	list	10	[0.81055900621118016, 0.7996894409937888, 0…
[illegible]	[illegible]	[illegible]	array([4, 3, 3, 3, 2, 1, 2, 3, 4, 5, 2, 1, …

```
In [106]: runfile('C:/Users/Yamuna/Python codes/ML_AS1_quest_2.py',
wdir='C:/Users/Yamuna/Python codes')
ttest_ind_from_stats: t = -15.2977  p = 1.66663e-09
P is less than alpha(0.05), hence there is a significant difference between
the NaiveBayes and Random forest classification

In [107]:
```

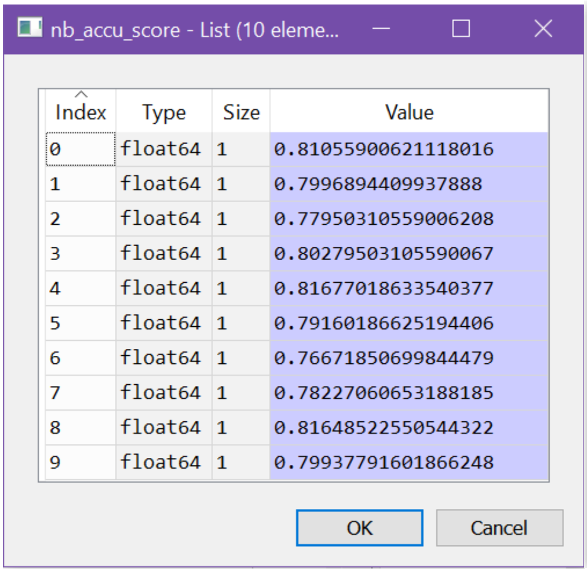
The value of ‘p’ from the statistical significance test is 1.66663e -09 and is less than the alpha value (0.05).

If p<0.05, then the observations are having statistically significant difference and

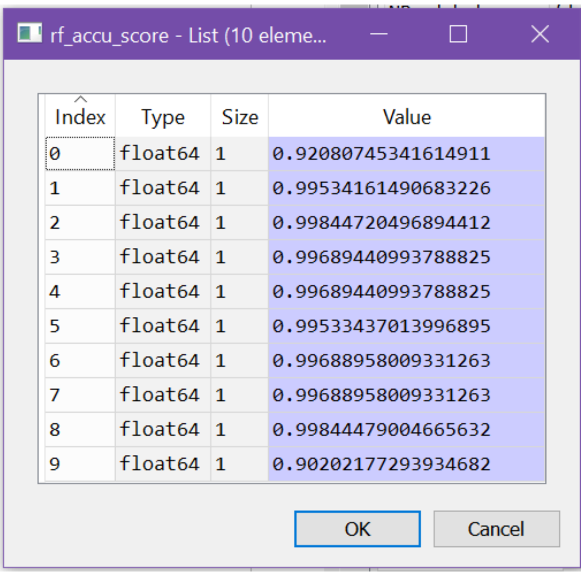
if p>0.05, then the observations are having statistically no significant difference.

Based on the output, Naïve Bayes and Random forest classifier are having statistically significant difference.

Naïve Bayes vs Random Forest 10 - Fold accuracy values



Index	Type	Size	Value
0	float64	1	0.81055900621118016
1	float64	1	0.7996894409937888
2	float64	1	0.77950310559006208
3	float64	1	0.80279503105590067
4	float64	1	0.81677018633540377
5	float64	1	0.79160186625194406
6	float64	1	0.76671850699844479
7	float64	1	0.78227060653188185
8	float64	1	0.81648522550544322
9	float64	1	0.79937791601866248



Index	Type	Size	Value
0	float64	1	0.92080745341614911
1	float64	1	0.99534161490683226
2	float64	1	0.99844720496894412
3	float64	1	0.99689440993788825
4	float64	1	0.99689440993788825
5	float64	1	0.99533437013996895
6	float64	1	0.99688958009331263
7	float64	1	0.99688958009331263
8	float64	1	0.99844479004665632
9	float64	1	0.90202177293934682