KATHMANDU UNIVERSITY
SCHOOL OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MINI PROJECT REPORT

GPA ANALYSIS SYSTEM

Submitted By:

Saugat Adhikari
Roll No: 02
Group: CE

Submitted To:

Dr. Bal Krishna Bal
Assistant Professor
DoCSE

Submission Date : 22nd Jan 2018

Project Idea

GPA Analysis System is a prediction model type of Machine Learning System. The main idea behind the project is to analyze the CGPA obtained by the students and make the predictions on GPA based on the different factors which have direct or indirect effect on GPA obtained. The main purpose of this system is to analyze the students' academic performance based on these factors.

The extracted dataset was first used to summarize the data using visualization techniques. The data was plotted to understand each attributes and the relationship between the attributes. Then some models of the data were created and their accuracy on unseen data were estimated. Then the loaded dataset was splitted into two, 80% of them were used to train the models and 20% were held back as validation dataset. Cross Validation technique was used to test the accuracy of the models we created. Then analyzing the different algorithms, best model was identified. And finally the predictions on GPA was made based on the best model we selected before.

Data Set

For obtaining the data, a google form was created with various questions about the information needed for the project like GPA, daily study hours, Stream, SLC percentage, +2 percentage etc. The google form was circulated to different classes of Department of Computer Science and Engineering and requested the students to fill them. The response obtained from that was stored in excel sheet and the data from that excel sheet was extracted in the program using pandas.read_excel module.

Software and Tools

Programming Language: Python

IDE: PyCharm IDE

Tools: numpy, pandas, matplotlib, sci-kit learn

Experiments done and Outputs achieved

1. Data Visualization

The idea is to extract data from the dataset and **summarize** it at first. The shape of the data is first found out and then the required number of data from the datasets are printed along with the head. Then the data are **described** by displaying the total count, mean, minimum value, maximum value, standard deviation and quartile deviation. Then afterwards the **grouping** of the data is done on the basis of different attributes. Then afterwards the **Visualization** of the data is done. The GPA graph is first plotted to see how the GPA score is varying among the students. Then the **univariate plot** is done to understand each attribute which includes box and whisker plot, density plot and histogram and **multivariate plot** to understand the relationship between attributes which includes scatter matrix plot.

```
Data Shape
(135, 18)
   Stream  Semester  SLC_percent  PlusTwo_percent  KUCAT    CGPA  \
0      CE         4        85.60             78.6   1066  3.4867
1      CE         4        84.60             86.4   1040  3.6400
2      CE         4        86.12             82.2   1168  3.9700
3      CE         3        90.00             80.0    945  3.6400
4      CE         4        86.00             81.0   1113  3.8167
5      CE         4        88.75             79.8    997  2.8000
6      CE         4        89.88             79.0   1100  3.5900
7      CE         4        87.00             77.0   1028  2.8000
8      CE         4        86.70             74.0    980  3.5500
9      CE         4        70.00             74.0   1006  2.7967
10     CE         4        85.50             78.8    955  3.6267
11     CE         3        85.00             79.0   1164  3.4670
12     CE         4        88.75             72.6    975  3.4670
13     CE         3        87.10             80.8    972  2.9967
14     CE         4        81.00             71.0    829  2.9550
15     CE         8        90.88             83.4    900  3.8040
```

Fig1: Data Set and Data Shape

```
Data Description
         Semester  SLC_percent  PlusTwo_percent        KUCAT        CGPA  \
count  135.000000   135.000000       135.000000   135.000000  135.000000
mean     4.348148    84.384296        59.271931   953.607407    3.142573
std      1.901761     4.562438        31.909983   194.244003    0.449731
min      2.000000    70.000000         0.000000     0.000000    1.500000
25%      2.000000    81.375000        63.500000   900.000000    2.868500
50%      4.000000    85.000000        74.000000   964.000000    3.173000
75%      6.000000    87.500000        79.400000  1067.000000    3.458500
max      8.000000    99.000000        90.000000  1600.000000    4.000000

       Interactive level in lectures?
count                      135.000000
mean                         3.074074
std                          0.927439
min                          1.000000
25%                          3.000000
50%                          3.000000
75%                          4.000000
```

Fig2: Data description

```
Grouping of Data
Daily hrs of study
less than one hour    80
more than one hour    55
dtype: int64

Do you do your Assignments yourself?
No                                 29
Of selected subjects only          65
Sometimes                           2
Sometimes i do sometimes i copy     1
Yes                                38
dtype: int64
```

Fig3: Grouping of Data

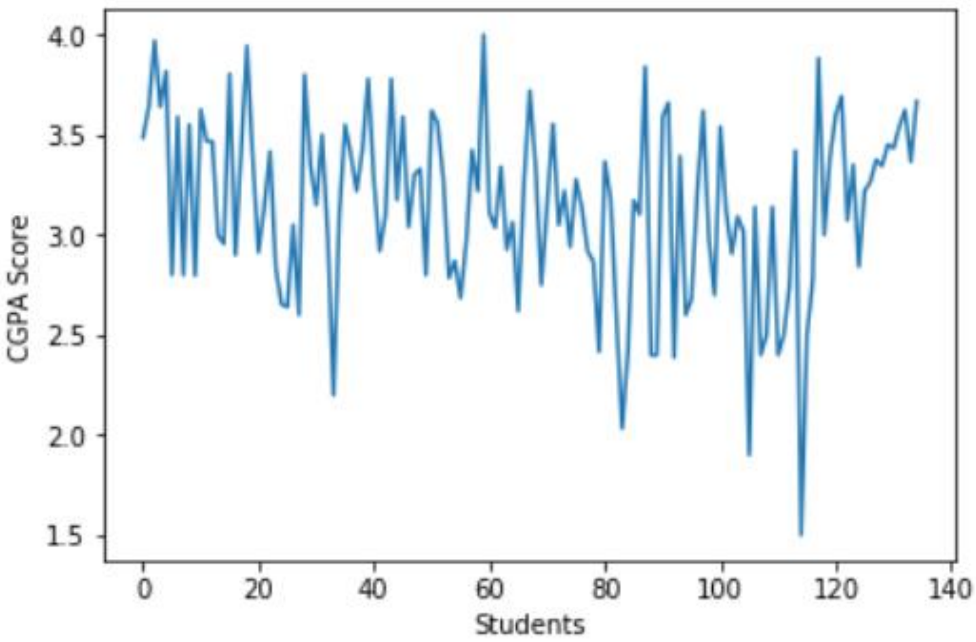

Fig4: CGPA Score Graph

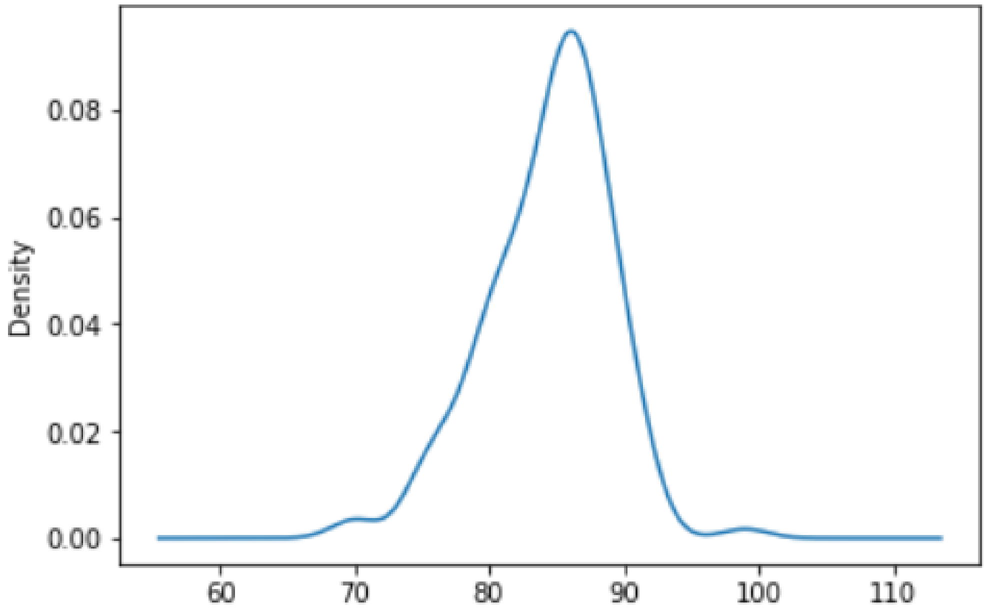

Fig5: Density plot of SLC %

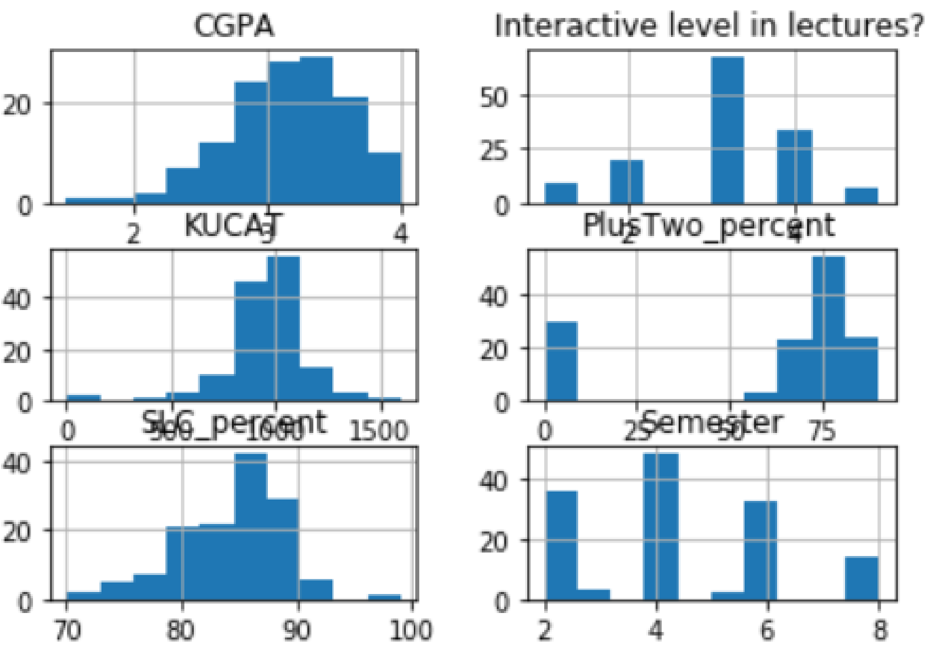

Fig6: Histogram plot

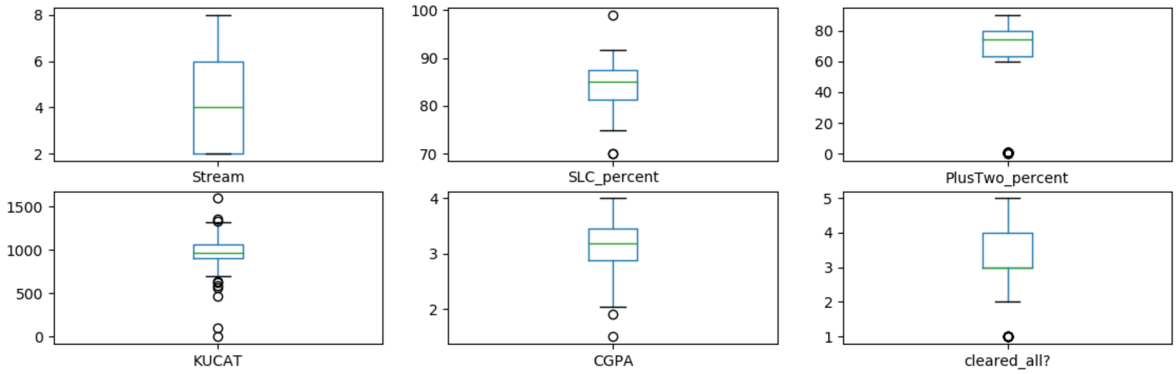

Fig7: Box and Whisker Plot

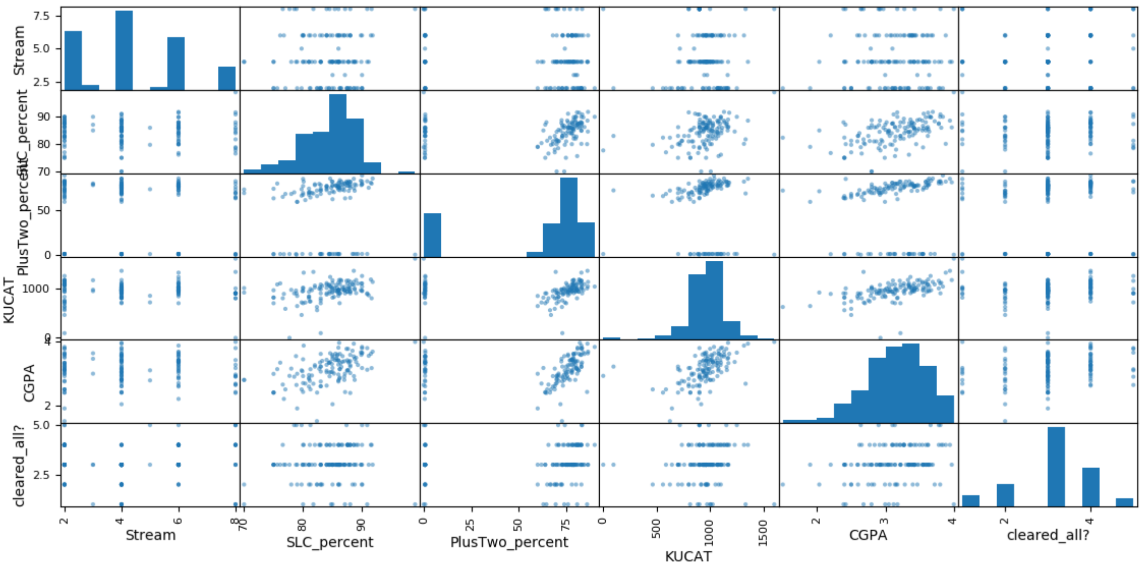

Fig8: Multivariate plot(Scatter Matrix)

2. Prediction using Linear Model

Then for **feature selection** on the obtained dataset **SelectKBest** class under **sklearn.feature_selection** module was used to select k best scoring features using **f_regression** as the scoring function. The training data and testing data were splitted and fetched using **train_test_split()** function under cross validation module. Now the **LinearRegression()** function under **linear_model** module was used to predict the GPA using the linear model. The **predicted GPA** was plotted in the graph as features vs predicted value. Looking at the graph, we could see that the machine predicted higher GPA for higher training values(SLC percentage, +2 percentage and KUCAT score) and lower GPA for lower training values (SLC percentage, +2 percentage and KUCAT). The predicted output value was similar to the original values we provided, thus the accuracy of the linear model was high.

```
(34, 6)
(101, 6)
(101,)
(34,)

-1.11022302463e-14

[ -3.76938639e-17   8.57860178e-17  -2.64027630e-19  -1.13606938e-18
   1.00000000e+00   7.34917875e-16]
```

Fig9: Train & Test data shape

```
Y_TEST
42      3.1000
36      3.4000
93      3.3940
5       2.8000
65      2.6200
108     2.5000
56      2.9700
17      3.4167
53      2.7850
120     3.6020
69      2.7500
119     3.3900
44      3.1750
59      4.0000
89      2.4000
115     2.5000
33      2.2000
```

Fig10: Sample Test Data

```
Y_PRED
[ 3.1     3.4     3.394   2.8     2.62    2.5     2.97    3.4167  2.785
  3.602   2.75    3.39    3.175   4.      2.4     2.5     2.2     2.867
  3.22    2.9     3.33    2.388   3.59    3.5     3.8167  3.344   3.      3.55
  2.685   3.0175  3.27    3.624   3.35    3.34  ]
```

Fig11: Predicted GPA Output

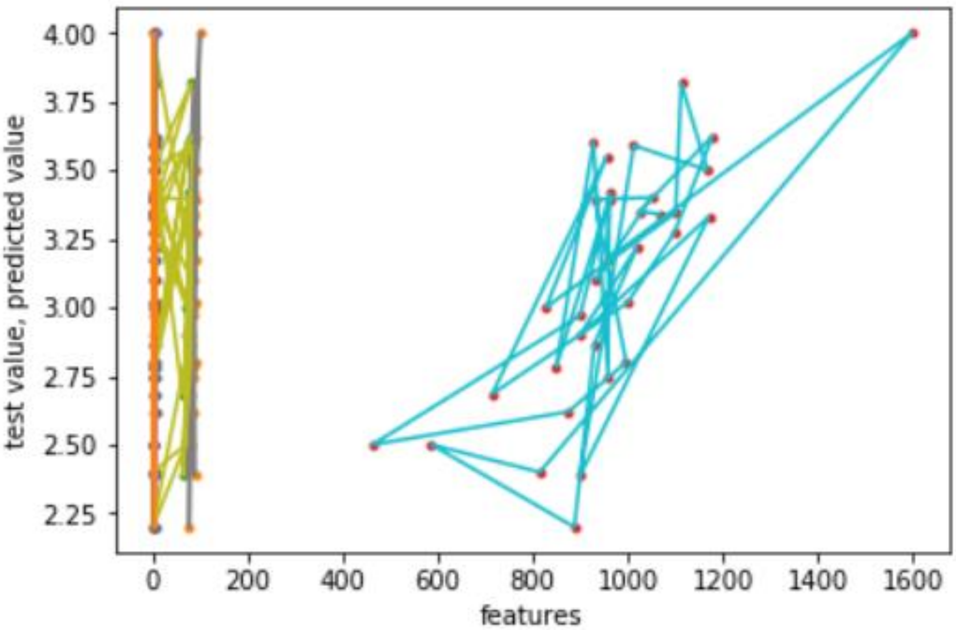

Fig12: Plot of selected features vs test value, predicted value

```
Mean Absolute Error
9.27362761746e-16

Mean Squared Error
1.25869717965e-30

Sqrt Mean Squared Error
1.12191674364e-15
```

Fig12: Obtained Errors

3. Creation of Validation Dataset

Then some **models** of the data were created to estimate their **accuracy** on the unseen data. Now the **Validation dataset** was created which contains 20% of the data. Remaining 80% of the data was used to **train** the models. To estimate the accuracy of the models, **10 fold Cross Validation** technique was used which splits our dataset into 10 parts, train on 9 and test on 1 and repeat for all combinations of train-test splits.

4. Building Models and Making Predictions

Then the models were created based on 6 different **algorithms** namely **Logistic Regression, Linear Discriminant Analysis, K-Nearest Neighbors, Classification and Regression Trees, Gaussian Naïve Bayes and Support Vector Machines**. The random number seed was reset before each run to ensure that the evaluation of each algorithm is performed using exactly the same data splits. It ensures the results are directly comparable. The plot for the **Algorithms Comparison** was then made. Looking at the accuracy score we find that **LDA algorithm has the highest accuracy** among these 6 algorithms with an accuracy of around **92.5%** so the LDA model was selected to make **predictions** on GPA status which was then plotted on the graph. Looking at the graph, we saw that for higher training values, obtained GPA status would be Outstanding, Excellent, Very Good and so on and for lower training values, obtained GPA status would be Good, Fair, Poor and Very Poor. The output thus matched with the algorithms' accuracy score. Then a plot of the model evaluation results was done and the spread and the mean accuracy of each model were compared. Then the confusion matrix and classification report of the predicted samples were calculated. The accuracy of the algorithm obtained from classification report thus matched with the accuracy calculated earlier.

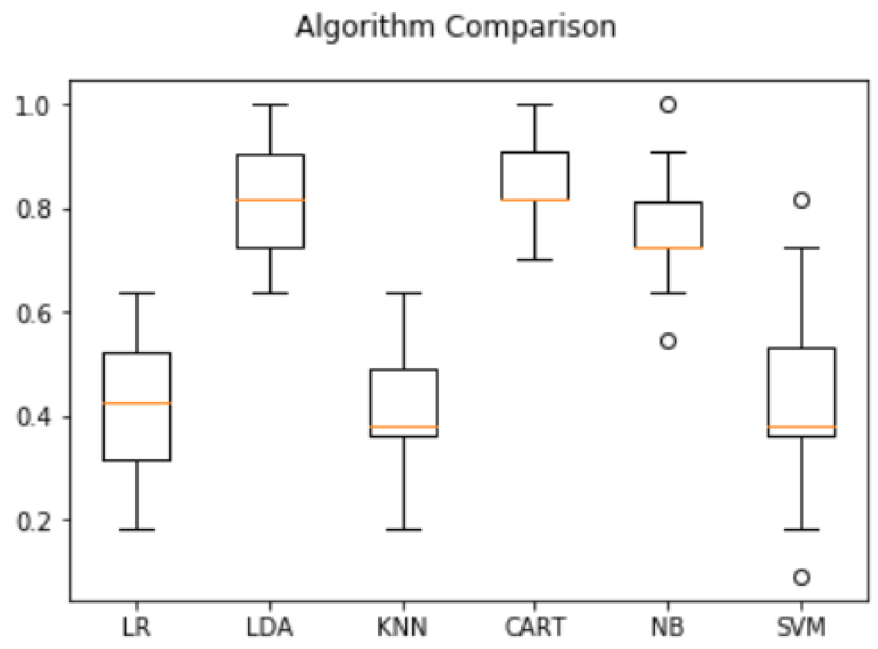

Fig13: Algorithm Comparison

```
Accuracy Score
0.925925925926
```

Fig14: Obtained Accuracy Score

```
Confusion Matrix
[[1 0 0 1 0 0 0]
 [0 5 0 0 0 0 0]
 [0 1 9 0 0 0 0]
 [0 0 0 1 0 0 0]
 [0 0 0 0 2 0 0]
 [0 0 0 0 0 6 0]
 [0 0 0 0 1 0 0]]
```

Fig15: Confusion Matrix

```
Classification Report
             precision    recall  f1-score   support

  Excellent       1.00      1.00      1.00         2
       Fair       0.83      1.00      0.91         5
       Good       1.00      0.90      0.95        10
Outstanding       1.00      1.00      1.00         1
       Poor       0.67      1.00      0.80         2
  Very Good       1.00      1.00      1.00         6
  Very Poor       0.00      0.00      0.00         1

avg / total       0.91      0.93      0.91        27

['Very Good' 'Poor' 'Fair' 'Good' 'Good' 'Fair' 'Very Good' 'Very Good'
 'Good' 'Outstanding' 'Good' 'Fair' 'Very Good' 'Fair' 'Excellent' 'Poor'
 'Good' 'Excellent' 'Fair' 'Good' 'Good' 'Fair' 'Good' 'Very Good' 'Good'
 'Poor' 'Very Good']
```

Fig16: Classification Report and Predicted Outputs

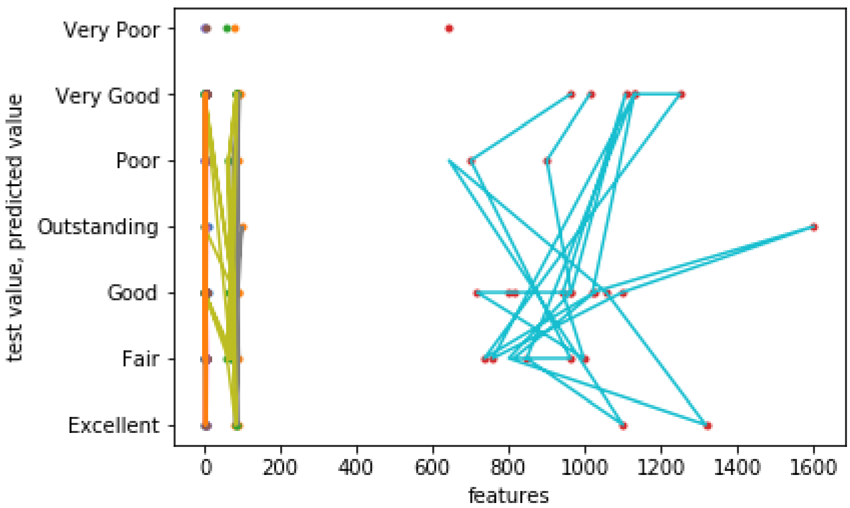

Fig17: Final plot of selected features vs predicted GPA status

Source Code

```
__author__ = 'Saugat'


import pandas
import numpy as np
import itertools
from pandas.plotting import scatter_matrix
from matplotlib import pyplot as plt
from sklearn import linear_model
from sklearn import model_selection
from sklearn.cross_validation import train_test_split
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_regression
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC


url = "E:\\Study\\7th sem\\ML - COMP 484\\ML_files\\data.xls"
names = ['Stream','Semester','SLC_percent',
         'PlusTwo_percent', 'KUCAT', 'CGPA', 'Interactive level in lectures?','GPA
Status', 'Seriousness in pre-exam breaks',
         'Daily hrs of study', 'Googling',
         'Personal_hobby','Do you do your Assignments yourself?', 'Attendance',
         'Interactive_in_lectures','Study_materials','Online_course',
         'Stay_KU']
dataset = pandas.read_excel(url, names=names)
# dataset2 = dataset['SLC_percent','PlusTwo_percent','KUCAT']

#shape

print("Data Shape")
print(dataset.shape)

#head

print(dataset.head(20))
# integer_dataset = dataset.loc(:,'SLC_percent':'CGPA')
# print(integer_dataset.head(10))
#
# #descriptions

print("Data Description")
print(dataset.describe())
print('')
#
# #class distribution

print("Grouping of Data")
```

```
print(dataset.groupby('Daily hrs of study').size())
print(' ')
print(dataset.groupby('Do you do your Assignments yourself?').size())

#Graph_CGPA

values = dataset.values
GPA = values[:,5]
plt.xlabel('Students')
plt.ylabel('CGPA Score')
plt.plot(GPA)
plt.show()
#


# #UNIVARIATE PLOTS
#box and whisker plots

dataset.plot(kind='box', subplots=True, layout=(3,3), sharex=False, sharey=False)
plt.show()
#

#Density Plot
dataset['SLC_percent'].plot(kind='density', subplots=True, layout=(1,1),
sharex=False )
plt.show()

#
plt.xlabel('Students')
plt.ylabel('KUCAT Score')
plt.plot(dataset['KUCAT'],'r')
# fig = plt.figure()
# ax = fig.add_subplot(111)
ax.set_xlabel("2013", fontsize=12)

#histograms

dataset.hist()
plt.show()
#
#MULTIVARIATE PLOTS

scatter_matrix(dataset)
plt.show()

#Prediction on GPA
# dataset2.drop('CGPA', axis=1, inplace=True)
array = dataset.values
dataset2 = array[:,1:7]
cgpa=dataset['CGPA']
# print("CGPA")
# print(cgpa)
X, y = dataset2, cgpa
X_new = SelectKBest(f_regression, k="all").fit(X,y)
X_new1 = X_new.transform(X)#transform(X, y)
# print (X_new1.shape)
# print (X_new1)

# print(X.columns[X_new.get_support()])
cgpa=dataset['CGPA']
X_train, X_test,y_train, y_test=train_test_split(X_new1,cgpa,random_state=1)

print (X_test.shape)
print (X_train.shape)
print (y_train.shape)
print (y_test.shape)
print("  ")
```

```python
linreg=linear_model.LinearRegression()
linreg.fit(X_train,y_train)

print(linreg.intercept_)
print("  ")
print(linreg.coef_)
print("  ")

y_pred=linreg.predict(X_test)
print (y_pred.shape)

print("Y_TEST")
print(y_test)
print("  ")

print("Y_PRED")
print(y_pred)
print("  ")

import matplotlib.pyplot as plt
get_ipython().magic('matplotlib inline')


plt.xlabel('features')
plt.ylabel('test value, predicted value')
plt.plot(X_test,y_test,'.',X_test,y_pred,'-')
plt.show()
print("  ")


from sklearn import metrics
import numpy as np
print("Mean Absolute Error")
print (metrics.mean_absolute_error(y_test,y_pred))
print("  ")
print("Mean Squared Error")
print (metrics.mean_squared_error(y_test,y_pred))
print("  ")
print("Sqrt Mean Squared Error")
print (np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
print("  ")


#split-out validation dataset

array = dataset.values
X = array[:,1:7]
Y = array[:,7]
# print(X)
# print(Y)
validation_size = 0.20
seed = 7
X_train, X_validation, Y_train, Y_validation = model_selection.train_test_split(X,
Y, test_size=validation_size, random_state=seed)

#test options and evaluation metric

seed = 7
scoring = 'accuracy'

#Spot Check Algorithms

models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
```

```python
models.append(('SVM', SVC()))


#evaluate each model in turn

results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=seed)
    cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold, 
scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)


#Compare Algorithms

fig = plt.figure()
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()

#make predictions on validation dataset

lr = LogisticRegression()
lda = LinearDiscriminantAnalysis()
knn = KNeighborsClassifier()
cart = DecisionTreeClassifier()
nb = GaussianNB()
svm = SVC()

lr.fit(X_train, Y_train)
lda.fit(X_train, Y_train)
knn.fit(X_train, Y_train)
cart.fit(X_train, Y_train)
nb.fit(X_train, Y_train)
svm.fit(X_train, Y_train)

predictions_lr = lr.predict(X_validation)
predictions_lda = lda.predict(X_validation)
predictions_knn = knn.predict(X_validation)
predictions = cart.predict(X_validation)
predictions_nb = nb.predict(X_validation)
predictions_svm = svm.predict(X_validation)

print("Accuracy Score LR")
print(accuracy_score(Y_validation, predictions_lr))
print("Accuracy Score LDA")
print(accuracy_score(Y_validation, predictions_lda))
print("Accuracy Score KNN")
print(accuracy_score(Y_validation, predictions_knn))
print("Accuracy Score CART")
print(accuracy_score(Y_validation, predictions))
print("Accuracy Score NB")
print(accuracy_score(Y_validation, predictions_nb))
print("Accuracy Score SVM")
print(accuracy_score(Y_validation, predictions_svm))


print("Confusion Matrix")
print(confusion_matrix(Y_validation, predictions))

print("Classification Report")
print(classification_report(Y_validation, predictions))
```

```
print(predictions)

plt.xlabel('features')
plt.ylabel('test value, predicted value')
plt.plot(X_validation,Y_validation,'.',X_validation,predictions,'-')
plt.show()
print("  ")
```

Lessons Learned

During the course of project, I got to learn many lessons. Working with pandas and sci-kit learn gave an insight to their possible usage in various Machine Learning projects. It was really nice to learn about datasets and their visualization to analyze how the rough data looks like. Training the machine with data given by us and predicting the outcome with high accuracy was the best part of the project. Since the dataset was relatively small as compared to the requirement for Machine Learning projects, feature selection had to be done to select the best scoring features. So, I realized that the dataset should be as large as possible for doing these types of projects. The analysis of various algorithms and their predicted outcome helped me learned about their accuracy and their use in various other projects in future.

Overall, it was a very good experience to learning Machine Learning with the completion of this project.

References

1. Pandas Data Analysis Library: https://pandas.pydata.org/
2. Sci-kit learn Documentation: https://pypi.python.org/pypi/scikit-learn/0.18
3. Linear Regression: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
4. Cross Validation and Model Selection: http://pythonforengineers.com/cross-validation-and-model-selection/
5. Feature Selection in Python: https://machinelearningmastery.com/feature-selection-machine-learning-python/