ROAD ACCIDENT ANALYSIS

Abstract–

Road accidents are one of the major causes of death and disability all over the world. Road accidents are the ninth leading cause of death in 2004 and expected to be fifth leading cause of death by 2030 worldwide. Reason for road accidents can be environmental conditions such as weather, traffic on road, type of road, speed and light conditions. The paper addresses the in depth analysis that identifies as the contributory factors behind the road accidents and the quantification of the factors that affect the frequency and severity of accidents based on the crash data available. It also reviews about the factors involved in the background of accidents and the associated methodologies that are applied for a thorough analysis in the field of interest. The determination of various parameters that leads to the crash is investigated by applying various machine learning algorithms which helps us to obtain a reliable conclusion and reconstruction based on the needs. The performance of the system has been analysed by user review to accuracy of 80%.

INTRODUCTION -

Road accidents are a serious issue resulting in huge economic loss. There is a need to bring down the accident rate, for which a detailed analysis of the factors which are responsible for the accidents are to be analyzed. Road accidents are influenced by a large number of attributes such as road conditions, traffic flow, environmental state and user's behavior on the road [1]. Proper analysis of the road accident data provides a useful insight regarding the causes and consequences of accidents. It is also good that, if the number of accidents is predicted well in advance so as to encounter the issues. Predictive analytics is the latest field which can be applied to this accident data to predict the future accidents. Machine learning is the artificial intelligence technique, which creates a model that learns from the past data. Once the model is created whenever a new set of data is given to it, it can predict the approximate values. Many machine learning algorithms are available like regression, classification, clustering, recommender system, churn prediction etc. In these machine learning concepts are applied on the data, a more accurate forecasting can be made possible. Road accidents are unsure and irregular events and their analysis needs to be aware of the factors that affect them. Road accidents are defined by a set of attributes that are often different.

Proper analysis of the road accident data provides a useful insight regarding the causes and consequences of accidents. It is also good that, if the number of accidents is predicted well in advance so as to encounter the issues. Predictive analytics is the latest field which can be applied to this accident data to predict the future accidents [2].

Classification is an important data mining technique which analyzes the data and classifies the data into a predefined set of classes. There are one or more machine learning algorithms are available; they are linear regression, logistic regression, decision tree, SVM, KNN, Random Forest algorithm. Experimental results reveals that, Random Forest outperformed other algorithms with higher accuracy and a lower error rate. This research paper shows that four causes: fatal accident, major injury accident, minor injury accident, total accidents and year wise reports for the road accidents in India

METHODOLOGY

Random Forest:

Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because it's simplicity and the fact that it can be used for both classification and regression tasks.Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Random Forests are also very hard to beat in terms of performance.

Decision Tree:

A decision tree is on if the most frequently & widely used supervised machine learning algorithms that can perform both regression & classification tasks.The intuition behind the decision tree algorithm is simple ,yet also powerful. For each attribute in the dataset,the decision tree algorithm forms a node,where the most important attribute is palced at the root node.for evaluation we start at the root node & work our way down the tree by following the corresponding nod that meets our condition or"decision".this process continues until a leaf node is reached,which contains the prediction or the outcome of the decision tree.

KNN:

K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity.KNN has been used in statistical estimation and pattern recognition. A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If K = 1, then the case is simply assigned to the class of its nearest neighbor. KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry.

SVM:

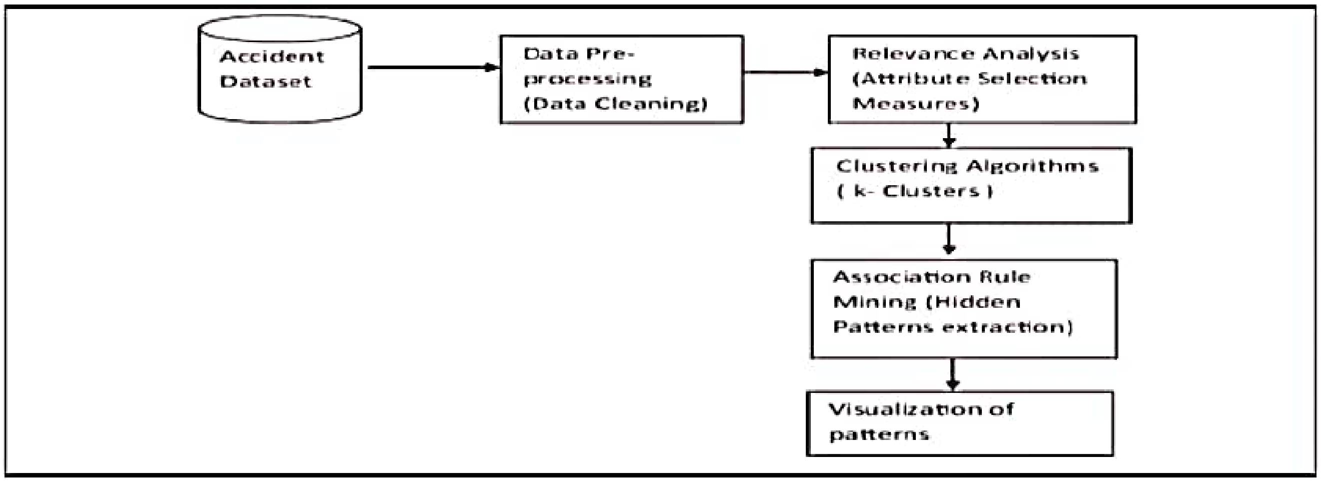
Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.It works really well with clear margin of separation.It is effective in high dimensional spaces.It is effective in cases where number of dimensions is greater than the number of samples.

PROPOSED ALGORITHM

Following steps are used in constructing system.

Database Creation :

In this step, A total of 30 attributes that focus on various criteria, such as accident-specific attributes, driver-specific attributes, circumstance-specific attributes, and other attributes given in the FIR report, form the input dataset.


Accident Dataset
Data Pre-processing (Data Cleaning)
Relevance Analysis (Attribute Selection Measures)
Clustering Algorithms (k- Clusters)
Association Rule Mining (Hidden Patterns extraction)
Visualization of patterns

Data Preprocessing :

After Database creation, Data pre-processing helps to remove noise, missing values, and inconsistencies. Missing values are replaced with NULL. Also each attribute data is discretized in order to make it appropriate for further analysis.

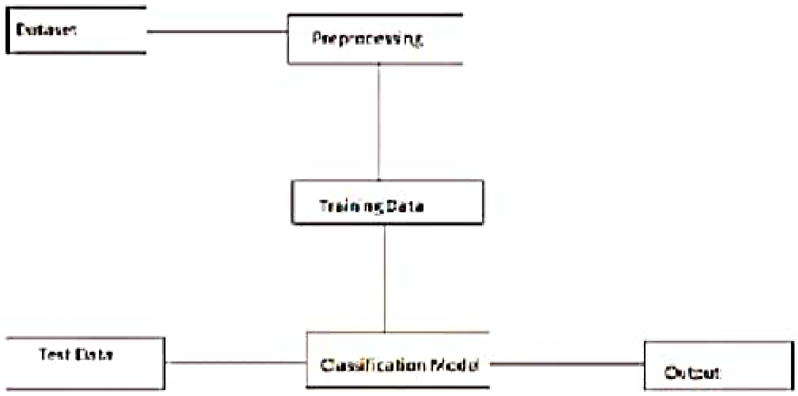
Data Cleaning:

In this step, The primary goal of data cleaning is to detect and remove errors and anomalies to increase the value of data in analytics and decision making. While it has been the focus of many researchers for several years, individual problems have been addressed separately. These include missing value imputation, outliers detection, transformations, integrity constraints violations detection and repair, consistent query answering, deduplication, and many other related problems such as profiling and constraints mining.

Performance Evaluation:

1. Performance Measure - The performance measure is the way you want to evaluate a solution to the problem.
2. Test and Train Datasets- From the transformed data, you will need to select a test set and a training set.
3. Cross Validation.

PROPOSED ARCHITECTURE



The main objective of this research is to investigate the role of human, vehicle, and infrastructure-related factors in accident severity by applying machine learning techniques on road accident data.The steps include data cleaning, data transformation, relevance analysis, clustering, association rules generation, and finally performance evaluation

PERFORMANCE ANALYSIS

The performance is being analyzed by using database survey. Multiple reviews from users have been collected on the basis of simplified text.

Algorithms	**Current Accuracy**	**Previous Accuracy**
Random Forest	80 %	78.6%
Decision Tree	61%	83.6%
KNN	72%	77%
Naïve Bayes	73%	83.7%
Linear Regression	29%	76.8%

CONCLUSION

In this study, the technique of association rules with a large set of accident's data to identify the reasons of road accidents were used.The results show that this model could provide good predictions against traffic accident with 80% correct rate.It should be noted that due to the constraints of data and research condition, there are still some factors, such as engine capacity, traffic flows, accident location etc., not used in the model and they should be taken into account in future study. The results of this study can be used in vehicle safety assistance driving and provide early warnings and proposals for safe driving.

Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
train=pd.read_csv('Road_train.csv')
test=pd.read_csv('Road_test.csv')
train.head()

#train['src']='train'
#test['src']='test'
data=pd.concat([train,test],ignore_index=True)
data.shape

#cheking Missing values
data.isnull().sum()

plt.figure(1)

data['Time']=data.Time.str.replace(':','').astype('float64')
data['Date']=data.Date.str.replace('/','').astype('float64')
#data['src']=data.src.str.replace(' ',' ').astype('float64')

import numpy as np
import pandas as pd
cols=['Date','Time']
for col in cols:
   data[col]=data[col].astype(dtype=np.float64)
data.info()

new_data=data.dropna(axis=0,how='any')

new_data.isnull()
print(new_data.isnull().sum())#Total missing value
threshold=len(new_data)*.1
threshold
threshold=len(new_data)*.1
new_data.dropna(thresh=threshold,axis=1)#drops a colm that has < valus that of threshold
# axis=1 drop col
print(new_data.isnull().sum())

y=(new_data.Road_Type)
X=new_data.drop("Road_Type",axis=1)
#print(y)

new_data.head()

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

```
from sklearn import datasets,linear_model
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn import linear_model
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA,TruncatedSVD
from sklearn import preprocessing

df_x=new_data.iloc[:,1:].values
df_y=new_data.iloc[:,0].values

x=new_data.iloc[:,1:].values
y=new_data.iloc[:,0].values

df = pd.DataFrame(data.drop(['accident_index'], axis = 1))
df.head()

df.dropna()

new_data=data.dropna(axis=0,how='any')

new_data.isnull()
print(new_data.isnull().sum())#Total missing value
threshold=len(new_data)*.1
threshold
threshold=len(new_data)*.1
new_data.dropna(thresh=threshold,axis=1)#drops a colm that has < valus that of threshold
# axis=1 drop col
print(new_data.isnull().sum())

y=(new_data.Road_Type)
X=new_data.drop("Road_Type",axis=1)
#print(y)

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)

new_data.dtypes

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn import datasets,linear_model
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn import linear_model
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA,TruncatedSVD
from sklearn import preprocessing

df_x=new_data.iloc[:,1:].values
df_y=new_data.iloc[:,0].values

x=new_data.iloc[:,1:].values
y=new_data.iloc[:,0].values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=4)

#rf=RandomForestClassifier(n_estimators=100)
#from sklearn import preprocessing
#le=preprocessing.LabelEncoder()
#rf.fit(x_train,y_train)
#pred=rf.predict(x_test)

reg=linear_model.LinearRegression()
reg.fit(x_train,y_train)
plt.style.use('fivethirtyeight')
plt.scatter(reg.predict(x_train),reg.predict(x_train)-y_train,color="green",s=10,label='Train
data')
plt.scatter(reg.predict(x_test),reg.predict(x_test)-y_test,color="blue",s=10,label='Train data')
plt.hlines(y=0,xmin=0,xmax=50,linewidth=2)
plt.legend(loc="upper right")
plt.title("Residual errors")
plt.show()

df = pd.DataFrame(data.drop(['accident_index'], axis = 1))
df.head()

df.dropna()
```

1.Random Forest

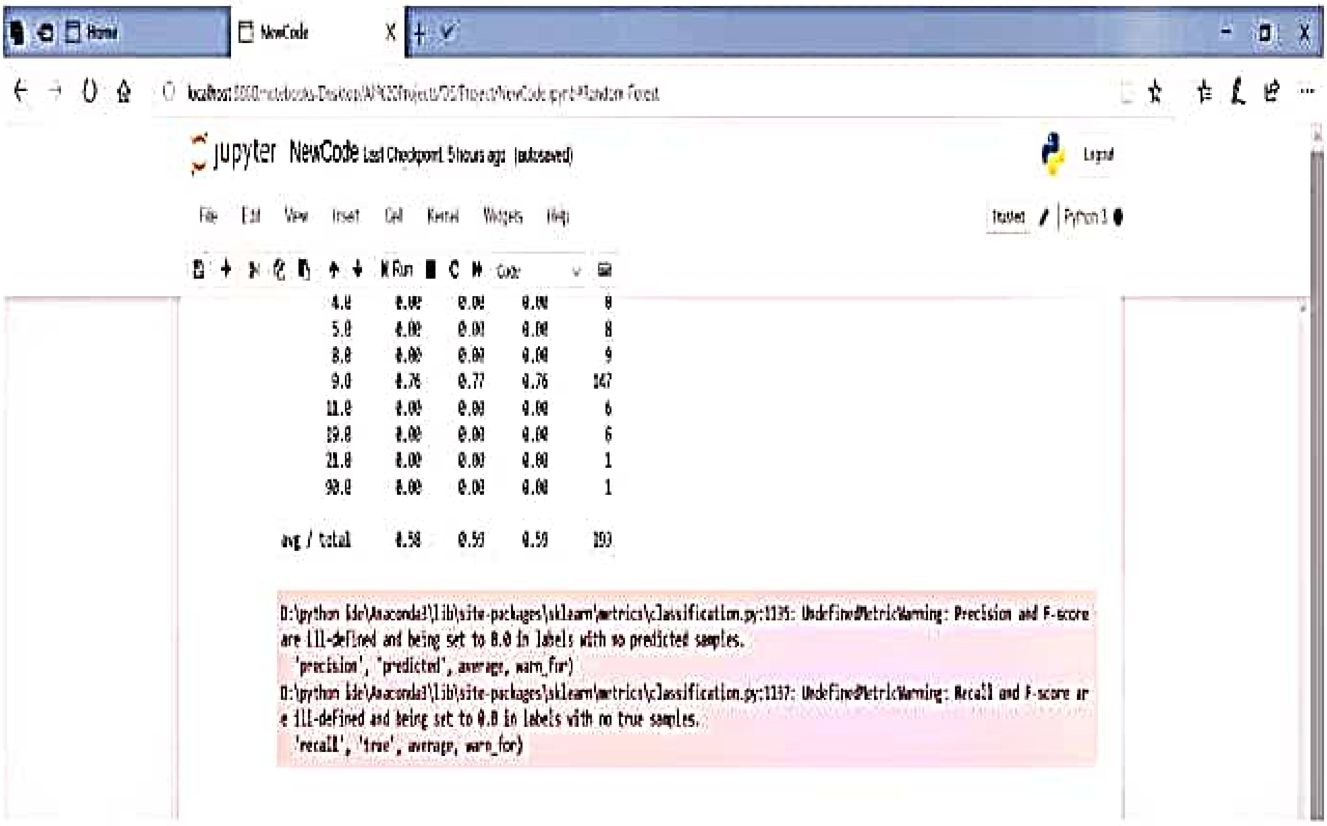
```
pca=PCA(n_components=16,whiten='True')
x=pca.fit(df_x).transform(df_x)
x_train,x_test,y_train,y_test=train_test_split(df_x,df_y,test_size=0.2,random_state=4)
rf=RandomForestClassifier(n_estimators=100)
rf.fit(x_train,y_train)
pred=rf.predict(x_test)
s=y_test
cnt=0
for i in range(len(pred)):
    if(pred[i]==s[i]):
        cnt=cnt+1
        #principle component analysis
cnt/float(len(pred))
```

2.Decision Tree For Classification

```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)

from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

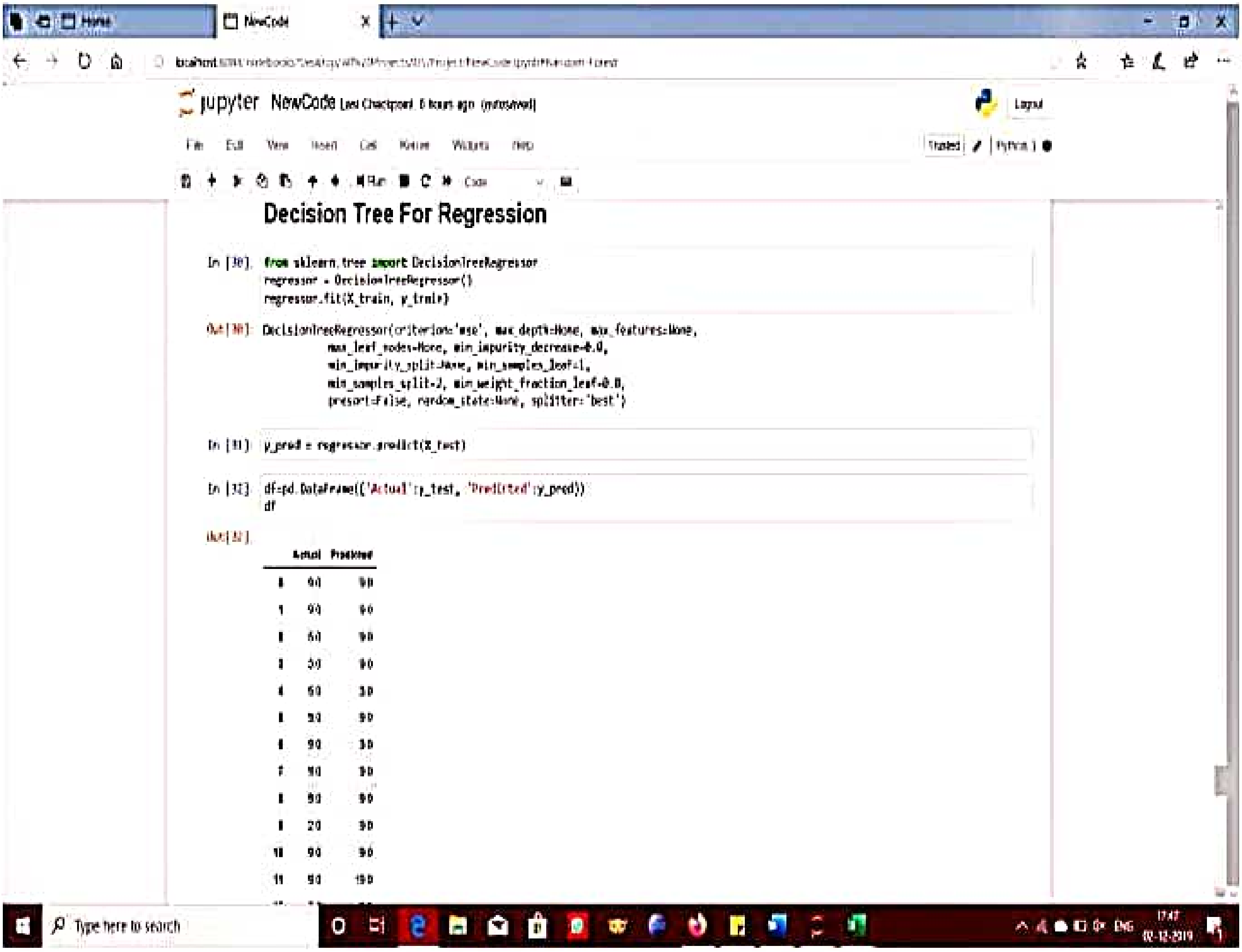
3.Decision Tree For Regression

```
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor()
regressor.fit(X_train, y_train)


y_pred = regressor.predict(X_test)

df=pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
df

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```



```
        4.0       0.00      0.00      0.00         8
        5.0       0.00      0.00      0.00         8
        8.0       0.00      0.00      0.00         9
        9.0       0.76      0.77      0.76       147
       11.0       0.00      0.00      0.00         6
       19.0       0.00      0.00      0.00         6
       21.0       0.00      0.00      0.00         1
       90.0       0.00      0.00      0.00         1

avg / total       0.58      0.59      0.59       193
```

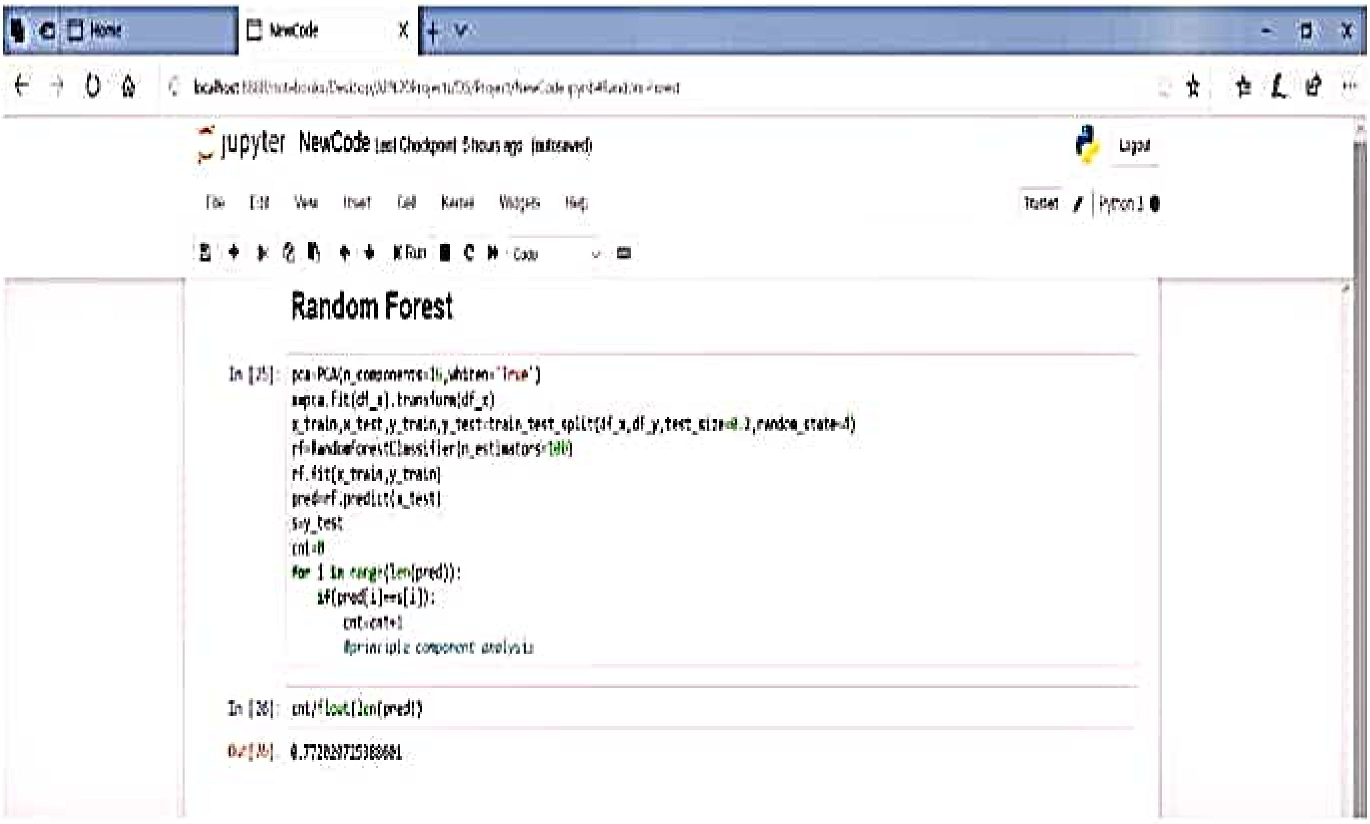
```
D:\python ide\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1135: UndefinedMetricWarning: Precision and F-score
are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
D:\python ide\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1137: UndefinedMetricWarning: Recall and F-score ar
e ill-defined and being set to 0.0 in labels with no true samples.
  'recall', 'true', average, warn_for)
```



Decision Tree For Regression

```
In [30]: from sklearn.tree import DecisionTreeRegressor
         regressor = DecisionTreeRegressor()
         regressor.fit(X_train, y_train)

Out[30]: DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
                    max_leaf_nodes=None, min_impurity_decrease=0.0,
                    min_impurity_split=None, min_samples_leaf=1,
                    min_samples_split=2, min_weight_fraction_leaf=0.0,
                    presort=False, random_state=None, splitter='best')

In [31]: y_pred = regressor.predict(X_test)

In [32]: df=pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
         df

Out[32]:
```

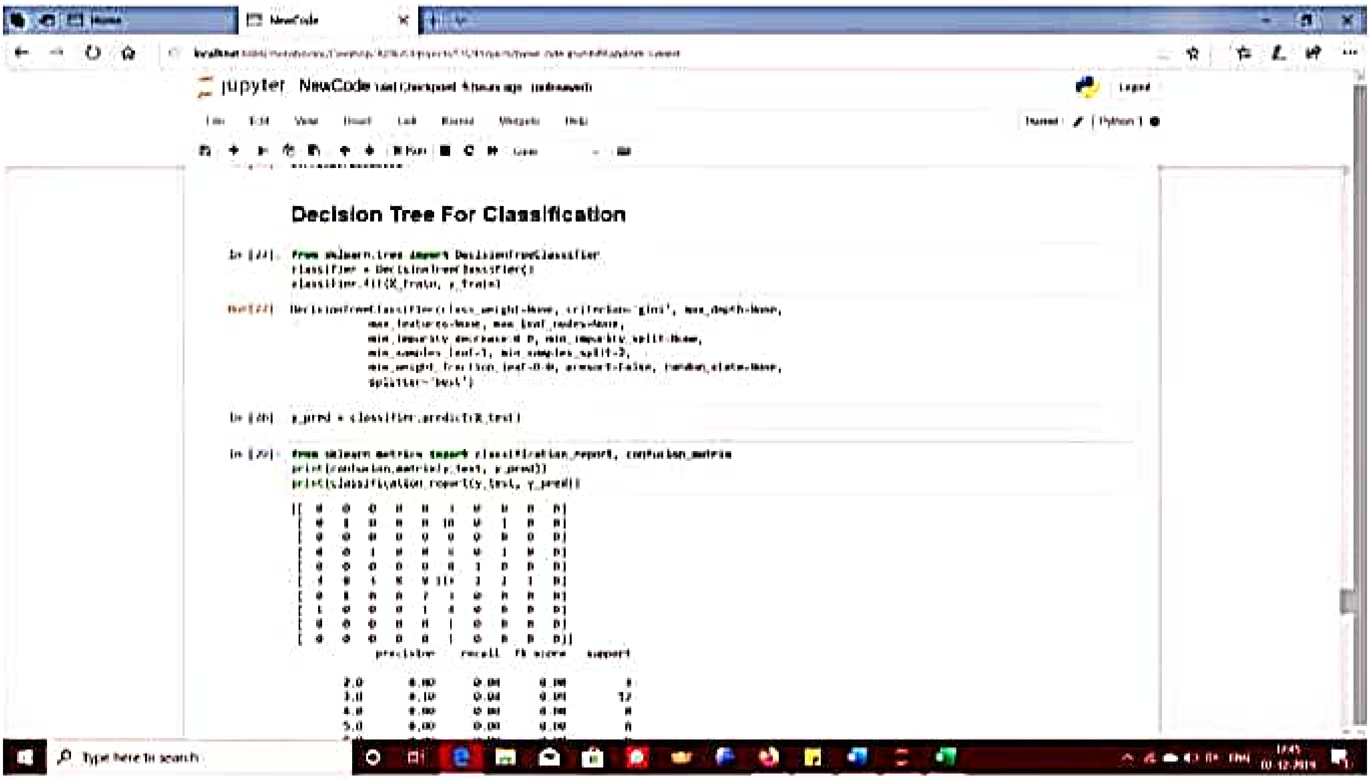
	Actual	Predicted
0	90	90
1	90	90
2	60	90
3	30	90
4	60	30
5	90	90
6	90	30
7	90	90
8	90	90
9	20	90
10	90	90
11	90	190



NewCode
jupyter NewCode Last Checkpoint: 5 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

Random Forest

```
In [25]: pca=PCA(n_components=16,whiten='True')
         x=pca.fit(df_x).transform(df_x)
         x_train,x_test,y_train,y_test=train_test_split(df_x,df_y,test_size=0.2,random_state=4)
         rf=RandomForestClassifier(n_estimators=100)
         rf.fit(x_train,y_train)
         pred=rf.predict(x_test)
         s=y_test
         cnt=0
         for i in range(len(pred)):
             if(pred[i]==s[i]):
                 cnt=cnt+1
                 #principle component analysis

In [26]: cnt/float(len(pred))

Out[26]: 0.7720207253886001
```



NewCode
jupyter NewCode Last Checkpoint: 5 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search

Decision Tree For Classification

```
In [27]: from sklearn.tree import DecisionTreeClassifier
         classifier = DecisionTreeClassifier()
         classifier.fit(X_train, y_train)

Out[27]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                     max_features=None, max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                     splitter='best')

In [28]: y_pred = classifier.predict(X_test)

In [29]: from sklearn.metrics import classification_report, confusion_matrix
         print(confusion_matrix(y_test, y_pred))
         print(classification_report(y_test, y_pred))

[[  0   0   0   0   0   1   0   0   0   0]
 [  0   1   0   0   0  10   0   1   0   0]
 [  0   0   0   0   0   0   0   0   0   0]
 [  0   0   1   0   0   6   0   1   0   0]
 [  0   0   0   0   0   0   1   0   0   0]
 [  4   0   1   5   9 115   2   2   1   0]
 [  0   1   0   0   7   1   0   0   0   0]
 [  1   0   0   0   1   4   0   0   0   0]
 [  0   0   0   0   0   1   0   0   0   0]
 [  0   0   0   0   0   1   0   0   0   0]]
              precision    recall  f1-score   support

         2.0       0.00      0.00      0.00         1
         3.0       0.10      0.08      0.09        12
         4.0       0.00      0.00      0.00         8
         5.0       0.00      0.00      0.00         8
```