DMW MINIPROJECT Rollno.-41239,41232

TITLE : Consider a labeled dataset belonging to an application domain. Apply suitable data preprocessing steps such as handling of null values, data reduction, discretization. For prediction of class labels of given data instances,build classifier models using different techniques (minimum 3), analyze the confusion matrix and compare these models. Also applycross validation while preparing the training and testing datasets.

PROBLEM STATEMENT : HEART DISEASE PREDICTION

S/W and H/W REQUIRED : Python,Jupyter,Matplotlib,Numpy,Pandas,ScikitLearn,64 bit OS, 8 GB RAM,500 GB HDD,Monitor,Keyboard.

OBJECTIVE : Students will be able to predict the heart disease whether it is present or not

THEORY:

1.Among all fatal disease, heart attacks diseases are considered as the most prevalent. Medical practitioners conduct different surveys on heart diseases and gather information of heart patients, their symptoms and disease progression. Increasingly are reported about patients with common diseases who have typical symptoms.

2.We have used the Heart Disease Dataset.It is based on UCI heart Disease Data Set [6] and we have 303 instances. According to UCI, “This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them.”We guess too many features will bring too much noise so people has done feature extraction and reduce 76 features to 14 features. To better understand the meaning of the features, we have the responsibility to explain some of the attributes of original dataset from UCI as follows:
• age: age in years
• sex: sex (1 = male; 0 = female)
• cp: chest pain type
-- Value 0: typical angina
-- Value 1: atypical angina
-- Value 2: non-anginal pain
-- Value 3: asymptomatic
• trestbps: resting blood pressure (in mm Hg on admission to the hospital)
• chol: serum cholestoral in mg/dl
• fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
• target: Heart disease (0 = no, 1 = yes)

METHODS USED

1) Logistic Regression

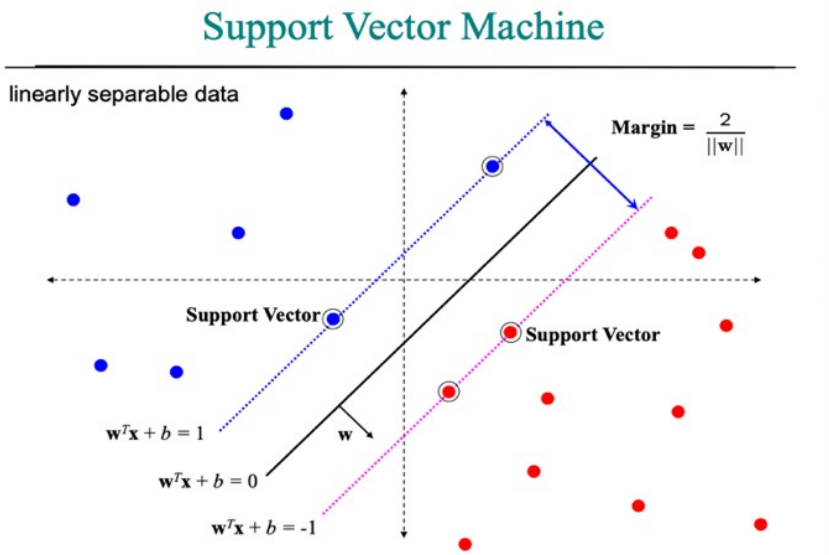
Logistic Regression is a supervised learning that computes the probabilities for classification problems with two outcomes. It can also be extended to predict several classes. In Logistic Regression model, we apply the sigmoid function, which is

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

This function successfully maps any number into the value between 0 and 1 and we can regard this value as the probability of predicting classes. For example, we have two classes and they are presence of heart disease and absence of disease. The accuracy score achieved using Logistic Regression is: 84 % which means the man has the 84% probability of having heart disease so we will predict that he has heart disease.

2)SVM (Support vector machine)

SVM aims to find a hyperplane in multiple dimensions (multiple features) that classifies the dataset. Here is a picture of classification by SVM.



The score for Support Vector Classifier is 83.0% with linear kernel.

3) KNN
KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry.

K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set.

The score for K Neighbors Classifier is 87.0% with 8 nieghbors.

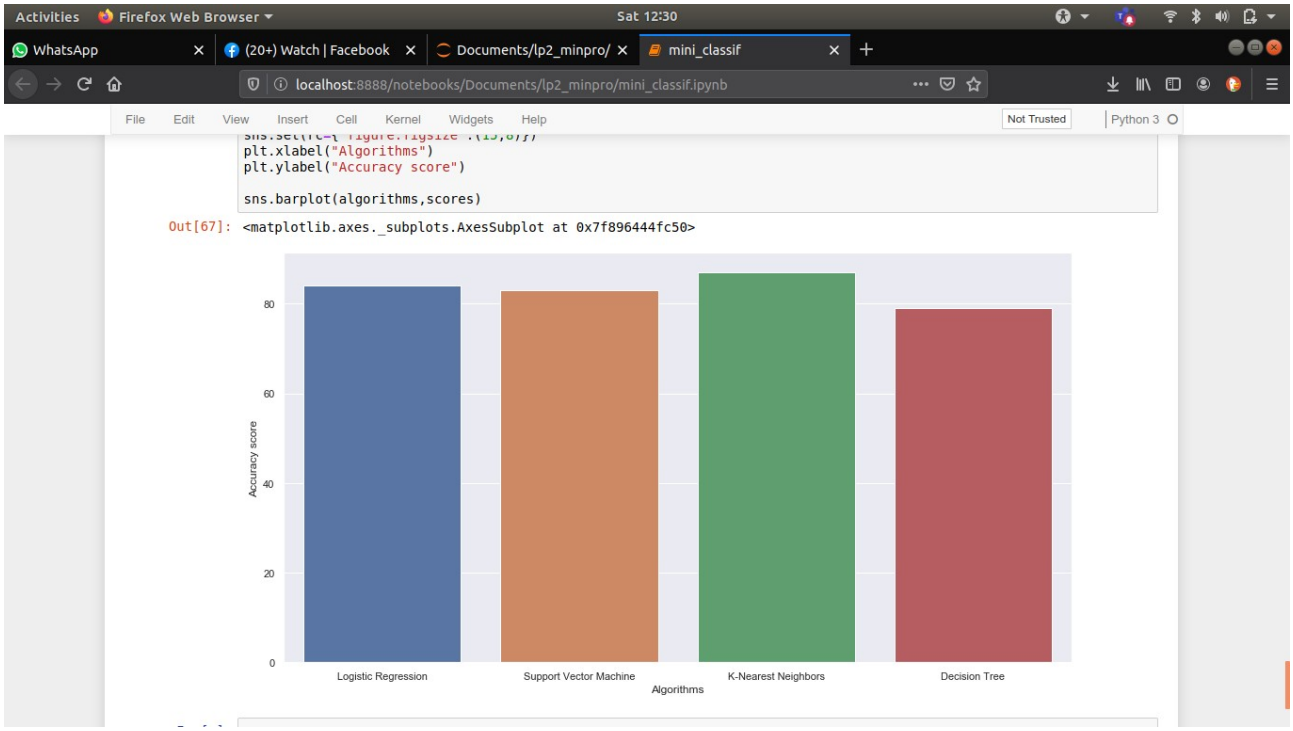
4) Decision Tree

The classification technique is a systematic approach to build classification models from an input dat set. For example, decision tree classifiers, rule-based classifiers, neural networks, support vector machines, and naive Bayes classifiers are different technique to solve a classification problem.

The decision tree classifiers organized a series of test questions and conditions in a tree structure. In the decision tree, the root and internal nodes contain attribute test conditions to separate recordes that have different characteristics.

The score for Decision Tree Classifier is 79.0% with [2, 4, 18] maximum features.

Conclusion:



Code -

1) Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import rcParams
from matplotlib.cm import rainbow
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

2) Read the Dataset

```
dataset = pd.read_csv('dataset.csv')
```

```
dataset.head(5)

3)Print the target variable count
rcParams['figure.figsize'] = 8,6
plt.bar(dataset['target'].unique(), dataset['target'].value_counts(), color = ['red', 'green'])
plt.xticks([0, 1])
plt.xlabel('Target Classes')
plt.ylabel('Count')
plt.title('Count of each Target Class')


4) Train Test split

y = dataset['target']
X = dataset.drop(['target'], axis = 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, random_state = 0)


5) Logistic Regression
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()

lr.fit(X_train,y_train)

Y_pred_lr = lr.predict(X_test)

6) KNN

knn_scores = []
for k in range(1,21):
    knn_classifier = KNeighborsClassifier(n_neighbors = k)
    knn_classifier.fit(X_train, y_train)
    knn_scores.append(knn_classifier.score(X_test, y_test))

plt.plot([k for k in range(1, 21)], knn_scores, color = 'red')
for i in range(1,21):
    plt.text(i, knn_scores[i-1], (i, knn_scores[i-1]))
plt.xticks([i for i in range(1, 21)])
plt.xlabel('Number of Neighbors (K)')
plt.ylabel('Scores')
plt.title('K Neighbors Classifier scores for different K values')


7) SVM

svc_scores = []
kernels = ['linear', 'poly', 'rbf', 'sigmoid']
for i in range(len(kernels)):
    svc_classifier = SVC(kernel = kernels[i])
    svc_classifier.fit(X_train, y_train)
    svc_scores.append(svc_classifier.score(X_test, y_test))
```

```
colors = rainbow(np.linspace(0, 1, len(kernels)))
plt.bar(kernels, svc_scores, color = colors)
for i in range(len(kernels)):
    plt.text(i, svc_scores[i], svc_scores[i])
plt.xlabel('Kernels')
plt.ylabel('Scores')
plt.title('Support Vector Classifier scores for different kernels')
```

8)Decision Tree

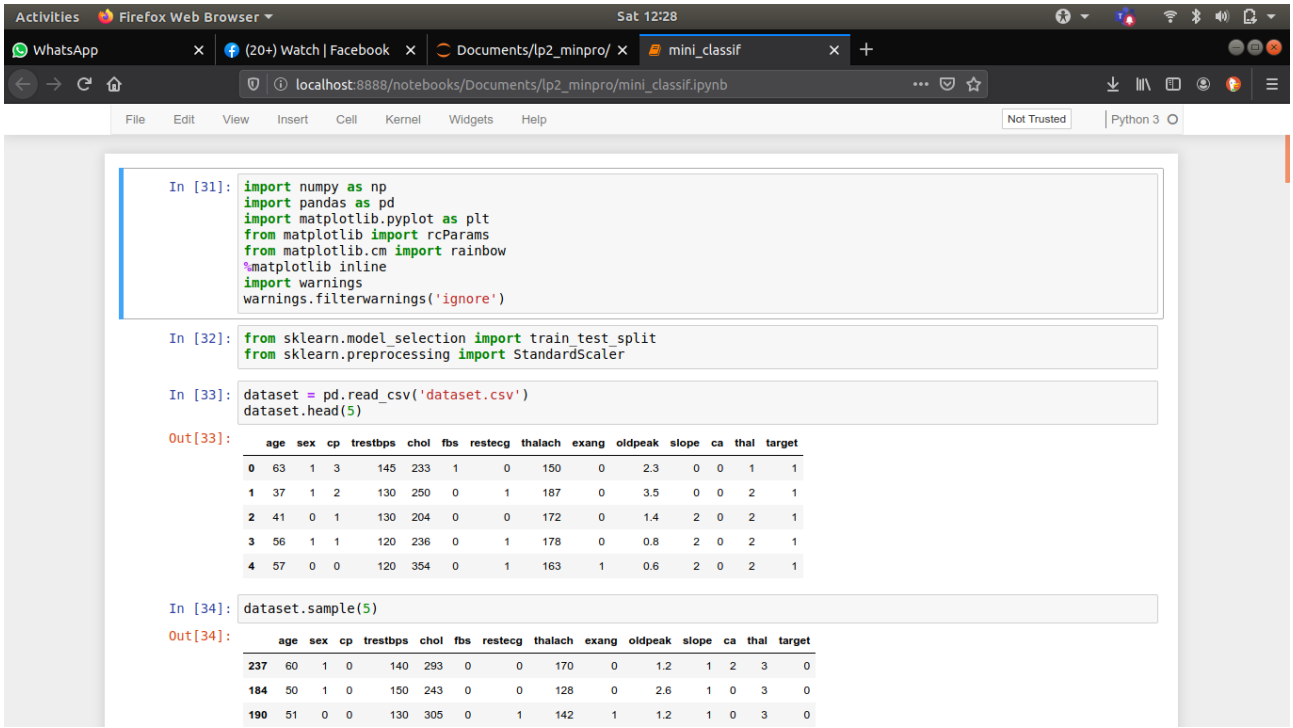
```
dt_scores = []
for i in range(1, len(X.columns) + 1):
    dt_classifier = DecisionTreeClassifier(max_features = i, random_state = 0)
    dt_classifier.fit(X_train, y_train)
    dt_scores.append(dt_classifier.score(X_test, y_test))
```

9)FINAL SCORE

```
scores = [score_lr,svm_score,knn_score,dt_score]
algorithms = ["Logistic Regression","Support Vector Machine","K-Nearest Neighbors","Decision Tree"]

for i in range(len(algorithms)):
    print("The accuracy score achieved using "+algorithms[i]+" is: "+str(scores[i])+" %")
```

Output -



Activities Firefox Web Browser Sat 12:28

WhatsApp | (20+) Watch | Facebook | Documents/lp2_minpro/ | mini_classif

localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

In [8]:
```
dataset.describe()
```

Out[8]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3

In [35]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
age         303 non-null int64
sex         303 non-null int64
cp          303 non-null int64
trestbps    303 non-null int64
chol        303 non-null int64
fbs         303 non-null int64
restecg     303 non-null int64
thalach     303 non-null int64
exang       303 non-null int64
oldpeak     303 non-null float64
slope       303 non-null int64
ca          303 non-null int64
thal        303 non-null int64
```

Activities Firefox Web Browser Sat 12:28

WhatsApp | (20+) Watch | Facebook | Documents/lp2_minpro/ | mini_classif

localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb

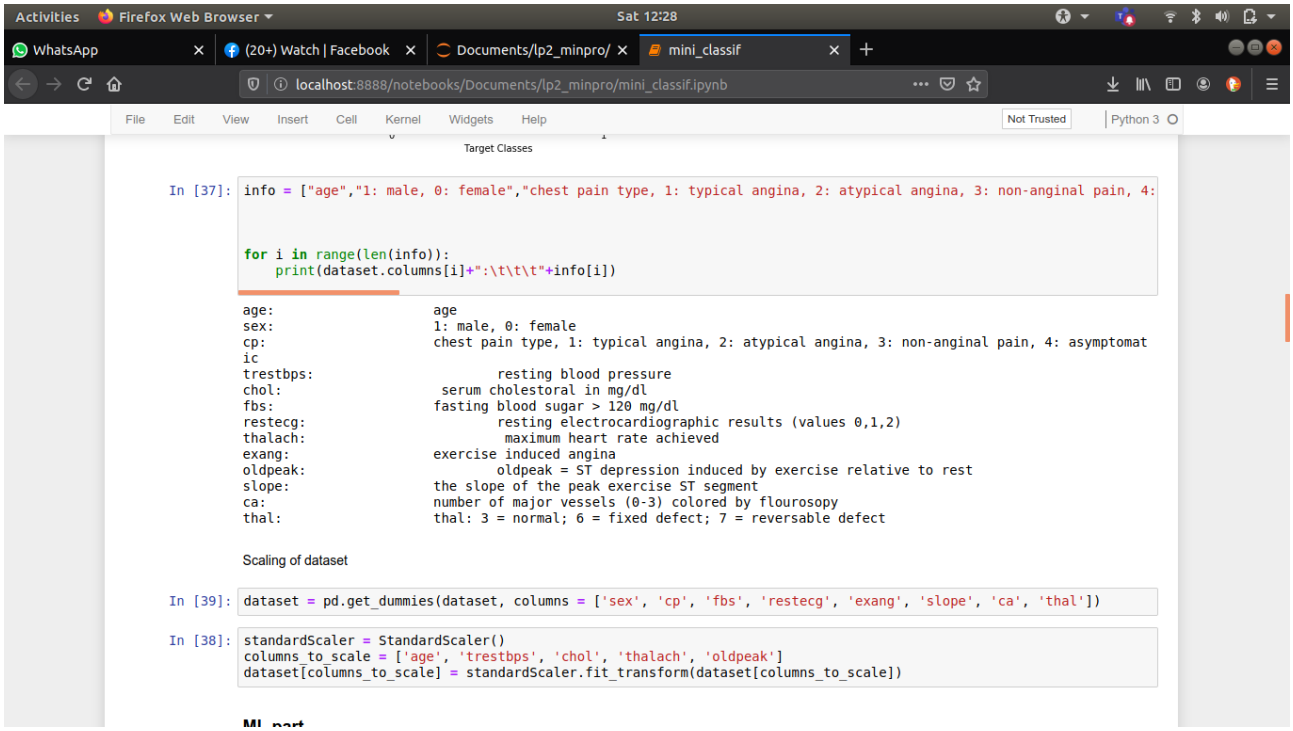
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

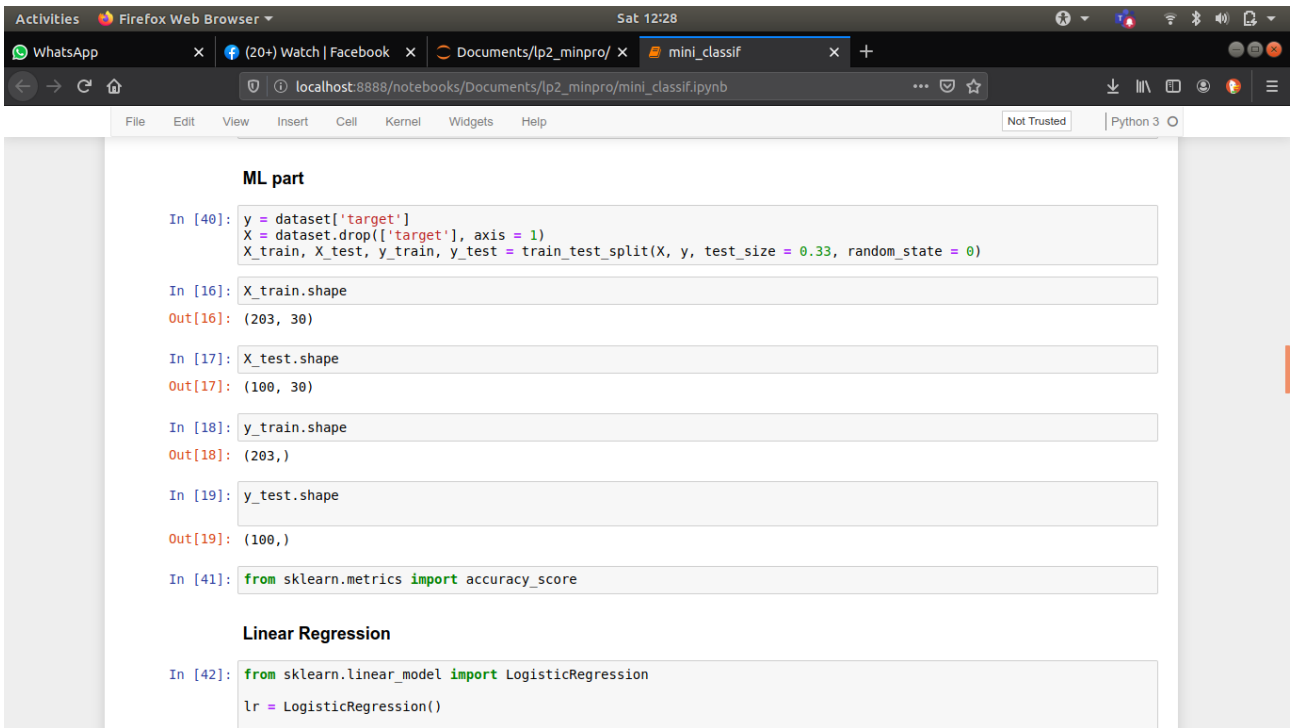
dataset has a total of 303 rows and there are no missing values. 13 features and one target value

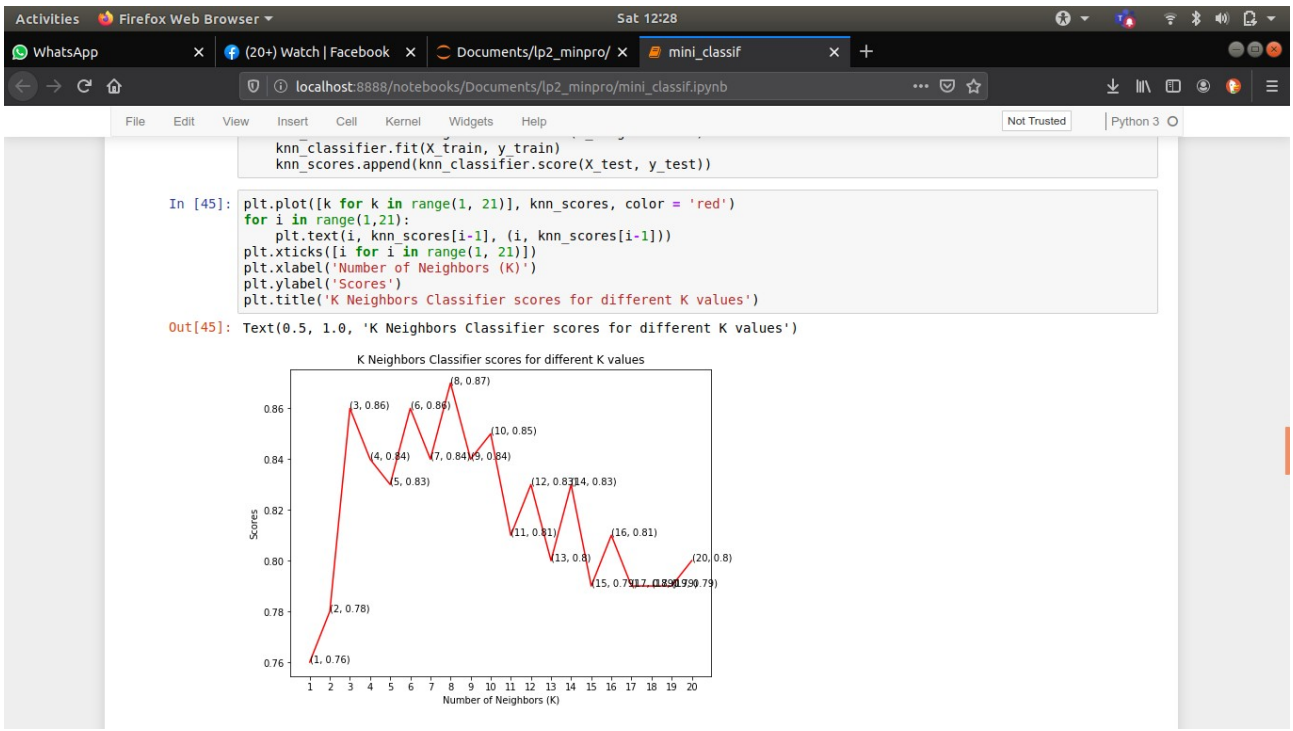
In [36]:
```
rcParams['figure.figsize'] = 8,6
plt.bar(dataset['target'].unique(), dataset['target'].value_counts(), color = ['red', 'green'])
plt.xticks([0, 1])
plt.xlabel('Target Classes')
plt.ylabel('Count')
plt.title('Count of each Target Class')
```

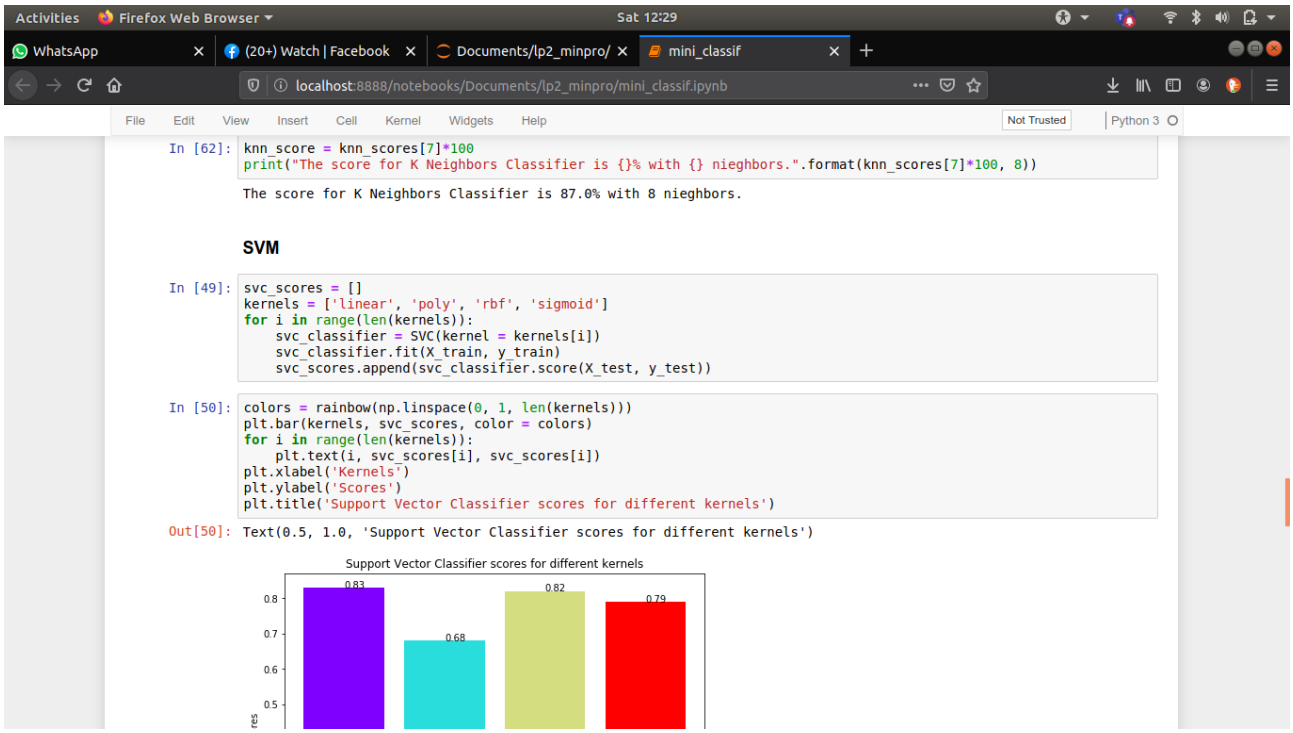
Out[36]:
```
Text(0.5, 1.0, 'Count of each Target Class')
```

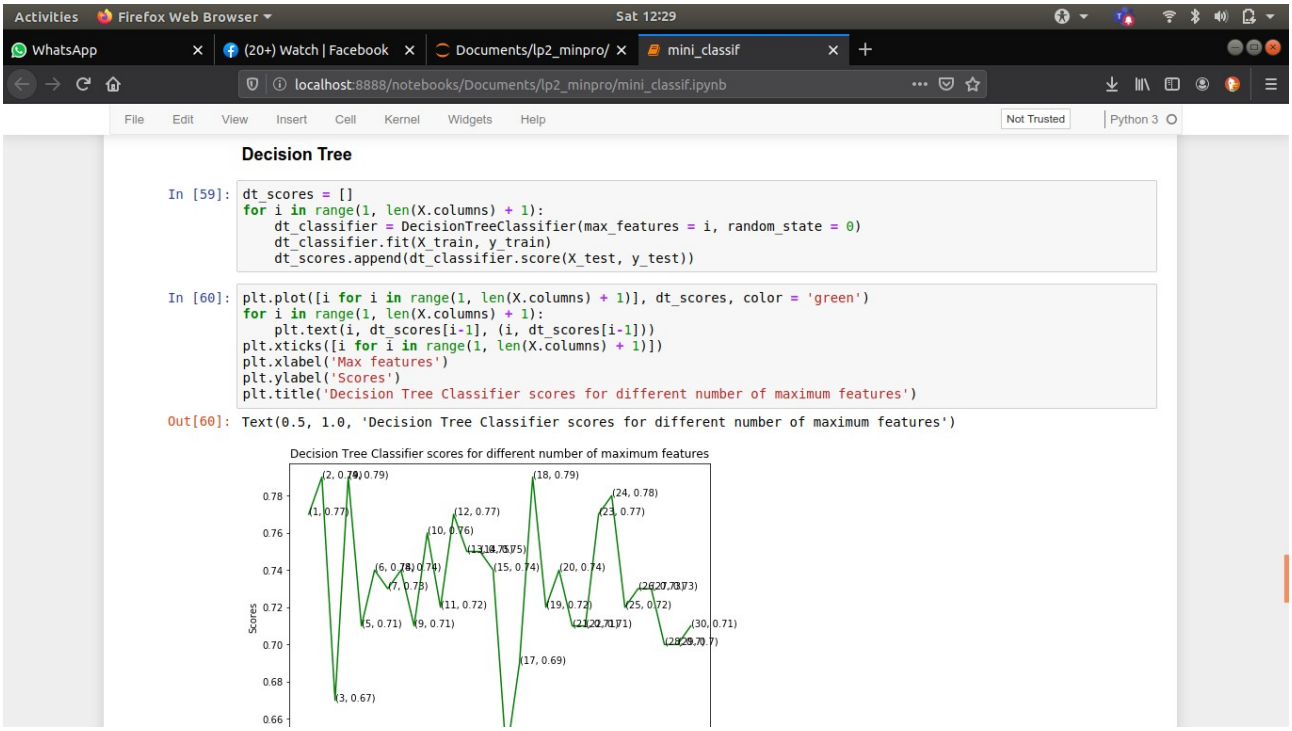
In [37]:
```
info = ["age","1: male, 0: female","chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4:
```

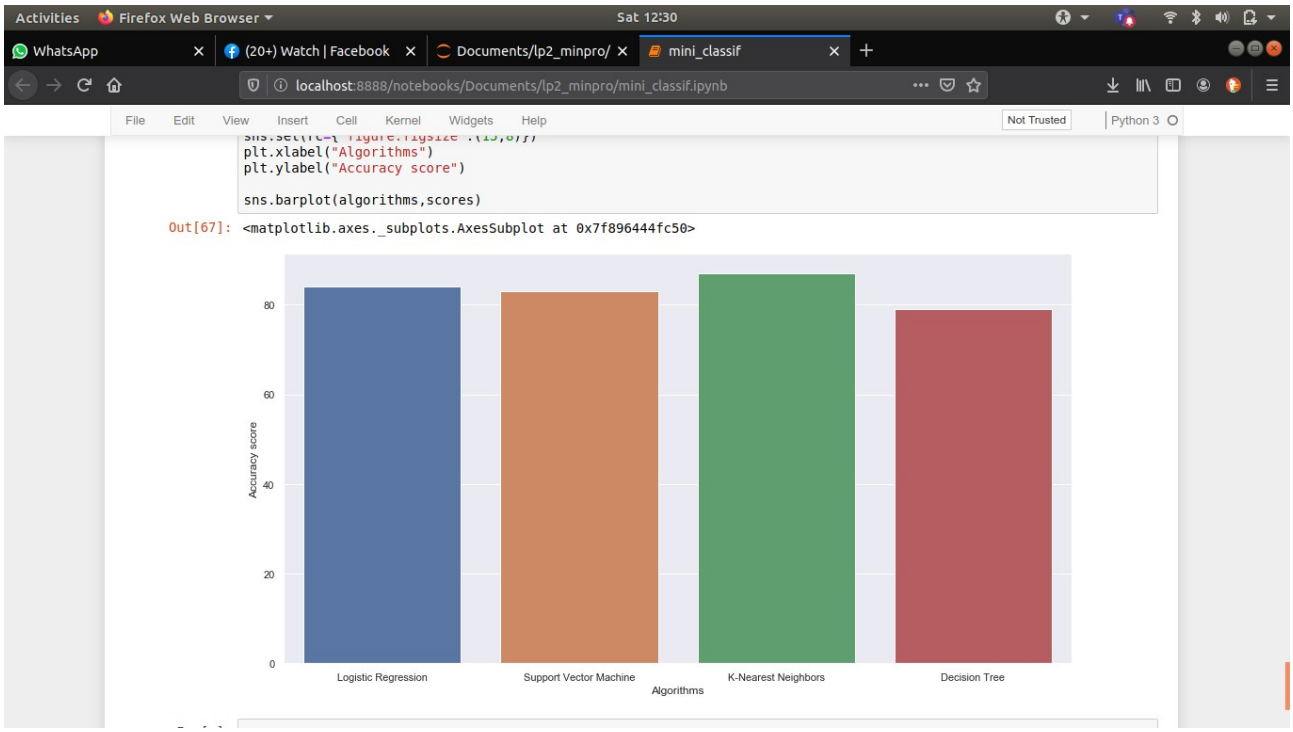

Activities
Firefox Web Browser
Sat 12:28
WhatsApp
(20+) Watch | Facebook
Documents/lp2_minpro/
mini_classif
localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Target Classes
In [37]:
info = ["age","1: male, 0: female","chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4:
for i in range(len(info)):
print(dataset.columns[i]+":\t\t\t"+info[i])
age: age
sex: 1: male, 0: female
cp: chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomat
ic
trestbps: resting blood pressure
chol: serum cholestoral in mg/dl
fbs: fasting blood sugar > 120 mg/dl
restecg: resting electrocardiographic results (values 0,1,2)
thalach: maximum heart rate achieved
exang: exercise induced angina
oldpeak: oldpeak = ST depression induced by exercise relative to rest
slope: the slope of the peak exercise ST segment
ca: number of major vessels (0-3) colored by flourosopy
thal: thal: 3 = normal; 6 = fixed defect; 7 = reversable defect
Scaling of dataset
In [39]:
dataset = pd.get_dummies(dataset, columns = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal'])
In [38]:
standardScaler = StandardScaler()
columns_to_scale = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
dataset[columns_to_scale] = standardScaler.fit_transform(dataset[columns_to_scale])


Activities
Firefox Web Browser
Sat 12:28
WhatsApp
(20+) Watch | Facebook
Documents/lp2_minpro/
mini_classif
localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
ML part
In [40]:
y = dataset['target']
X = dataset.drop(['target'], axis = 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, random_state = 0)
In [16]:
X_train.shape
Out[16]:
(203, 30)
In [17]:
X_test.shape
Out[17]:
(100, 30)
In [18]:
y_train.shape
Out[18]:
(203,)
In [19]:
y_test.shape
Out[19]:
(100,)
In [41]:
from sklearn.metrics import accuracy_score
Linear Regression
In [42]:
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()


Activities
Firefox Web Browser
Sat 12:28
WhatsApp
(20+) Watch | Facebook
Documents/lp2_minpro/
mini_classif
localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
knn_classifier.fit(X_train, y_train)
knn_scores.append(knn_classifier.score(X_test, y_test))
In [45]:
plt.plot([k for k in range(1, 21)], knn_scores, color = 'red')
for i in range(1,21):
plt.text(i, knn_scores[i-1], (i, knn_scores[i-1]))
plt.xticks([i for i in range(1, 21)])
plt.xlabel('Number of Neighbors (K)')
plt.ylabel('Scores')
plt.title('K Neighbors Classifier scores for different K values')
Out[45]:
Text(0.5, 1.0, 'K Neighbors Classifier scores for different K values')
K Neighbors Classifier scores for different K values
(8, 0.87)
(3, 0.86)
(6, 0.86)
(10, 0.85)
(4, 0.84)
(7, 0.84)
(9, 0.84)
(5, 0.83)
(12, 0.83)
(14, 0.83)
(11, 0.81)
(16, 0.81)
(13, 0.8)
(20, 0.8)
(2, 0.78)
(1, 0.76)
Scores
Number of Neighbors (K)


Activities
Firefox Web Browser
Sat 12:29
WhatsApp
(20+) Watch | Facebook
Documents/lp2_minpro/
mini_classif
localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
In [62]:
knn_score = knn_scores[7]*100
print("The score for K Neighbors Classifier is {}% with {} nieghbors.".format(knn_scores[7]*100, 8))
The score for K Neighbors Classifier is 87.0% with 8 nieghbors.
SVM
In [49]:
svc_scores = []
kernels = ['linear', 'poly', 'rbf', 'sigmoid']
for i in range(len(kernels)):
svc_classifier = SVC(kernel = kernels[i])
svc_classifier.fit(X_train, y_train)
svc_scores.append(svc_classifier.score(X_test, y_test))
In [50]:
colors = rainbow(np.linspace(0, 1, len(kernels)))
plt.bar(kernels, svc_scores, color = colors)
for i in range(len(kernels)):
plt.text(i, svc_scores[i], svc_scores[i])
plt.xlabel('Kernels')
plt.ylabel('Scores')
plt.title('Support Vector Classifier scores for different kernels')
Out[50]:
Text(0.5, 1.0, 'Support Vector Classifier scores for different kernels')
Support Vector Classifier scores for different kernels
0.83
0.68
0.82
0.79


Activities
Firefox Web Browser
Sat 12:29
WhatsApp
(20+) Watch | Facebook
Documents/lp2_minpro/
mini_classif
localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Decision Tree
In [59]:
dt_scores = []
for i in range(1, len(X.columns) + 1):
dt_classifier = DecisionTreeClassifier(max_features = i, random_state = 0)
dt_classifier.fit(X_train, y_train)
dt_scores.append(dt_classifier.score(X_test, y_test))
In [60]:
plt.plot([i for i in range(1, len(X.columns) + 1)], dt_scores, color = 'green')
for i in range(1, len(X.columns) + 1):
plt.text(i, dt_scores[i-1], (i, dt_scores[i-1]))
plt.xticks([i for i in range(1, len(X.columns) + 1)])
plt.xlabel('Max features')
plt.ylabel('Scores')
plt.title('Decision Tree Classifier scores for different number of maximum features')
Out[60]:
Text(0.5, 1.0, 'Decision Tree Classifier scores for different number of maximum features')
Decision Tree Classifier scores for different number of maximum features
Scores
(1, 0.77)
(3, 0.67)
(5, 0.71)
(9, 0.71)
(11, 0.72)
(12, 0.77)
(15, 0.74)
(17, 0.69)
(18, 0.79)
(19, 0.72)
(20, 0.74)
(23, 0.77)
(24, 0.78)
(25, 0.72)
(30, 0.71)


Activities
Firefox Web Browser
Sat 12:30
WhatsApp
(20+) Watch | Facebook
Documents/lp2_minpro/
mini_classif
localhost:8888/notebooks/Documents/lp2_minpro/mini_classif.ipynb
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
plt.xlabel("Algorithms")
plt.ylabel("Accuracy score")
sns.barplot(algorithms,scores)
Out[67]:
<matplotlib.axes._subplots.AxesSubplot at 0x7f896444fc50>
Accuracy score
0
20
40
60
80
Logistic Regression
Support Vector Machine
K-Nearest Neighbors
Decision Tree
Algorithms