```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

df=pd.read_csv('downloads/heart_disease_uci.csv')

df
```

```
      id  age     sex        dataset               cp  trestbps   chol
fbs  \
0      1   63    Male      Cleveland   typical angina     145.0  233.0
True
1      2   67    Male      Cleveland     asymptomatic     160.0  286.0
False
2      3   67    Male      Cleveland     asymptomatic     120.0  229.0
False
3      4   37    Male      Cleveland      non-anginal     130.0  250.0
False
4      5   41  Female      Cleveland  atypical angina     130.0  204.0
False
..   ...  ...     ...            ...              ...       ...    ...
...
915  916   54  Female  VA Long Beach     asymptomatic     127.0  333.0
True
916  917   62    Male  VA Long Beach   typical angina       NaN  139.0
False
917  918   55    Male  VA Long Beach     asymptomatic     122.0  223.0
True
918  919   58    Male  VA Long Beach     asymptomatic       NaN  385.0
True
919  920   62    Male  VA Long Beach  atypical angina     120.0  254.0
False

              restecg  thalch  exang  oldpeak        slope   ca  \
0      lv hypertrophy   150.0  False      2.3  downsloping  0.0
1      lv hypertrophy   108.0   True      1.5         flat  3.0
2      lv hypertrophy   129.0   True      2.6         flat  2.0
3              normal   187.0  False      3.5  downsloping  0.0
4      lv hypertrophy   172.0  False      1.4    upsloping  0.0
..                ...     ...    ...      ...          ...  ...
915  st-t abnormality   154.0  False      0.0          NaN  NaN
916  st-t abnormality     NaN    NaN      NaN          NaN  NaN
917  st-t abnormality   100.0  False      0.0          NaN  NaN
918    lv hypertrophy     NaN    NaN      NaN          NaN  NaN
919    lv hypertrophy    93.0   True      0.0          NaN  NaN

                  thal  num
0         fixed defect    0
1               normal    2
2    reversable defect    1
3               normal    0
4               normal    0
```

```
..                 ...  ...
915                NaN    1
916                NaN    0
917       fixed defect    2
918                NaN    0
919                NaN    1

[920 rows x 16 columns]
```

```
df.head()
```

```
   id  age     sex    dataset               cp  trestbps   chol    fbs
\
0   1   63    Male  Cleveland   typical angina     145.0  233.0   True

1   2   67    Male  Cleveland     asymptomatic     160.0  286.0  False

2   3   67    Male  Cleveland     asymptomatic     120.0  229.0  False

3   4   37    Male  Cleveland      non-anginal     130.0  250.0  False

4   5   41  Female  Cleveland  atypical angina     130.0  204.0  False


          restecg  thalch  exang  oldpeak        slope   ca  \
0  lv hypertrophy   150.0  False      2.3  downsloping  0.0
1  lv hypertrophy   108.0   True      1.5         flat  3.0
2  lv hypertrophy   129.0   True      2.6         flat  2.0
3         normal   187.0  False      3.5  downsloping  0.0
4  lv hypertrophy   172.0  False      1.4    upsloping  0.0

                thal  num
0       fixed defect    0
1             normal    2
2  reversable defect    1
3             normal    0
4             normal    0
```

```
df.shape
```

```
(920, 16)
```

```
df.drop(['id','dataset'],axis=1,inplace=True)
```

```
df.isnull().sum()
```

```
age          0
sex          0
cp           0
trestbps     59
chol         30
```

```
fbs          90
restecg       2
thalch       55
exang        55
oldpeak      62
slope       309
ca          611
thal        486
num           0
dtype: int64
```

```
df=df.dropna()
```

```
df.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalch      0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
num         0
dtype: int64
```

```
x=df.drop("num",axis=1)
y=df.num
```

```
x.head()
```

```
   age     sex               cp  trestbps   chol    fbs
restecg  \
0   63    Male   typical angina     145.0  233.0   True  lv
hypertrophy
1   67    Male     asymptomatic     160.0  286.0  False  lv
hypertrophy
2   67    Male     asymptomatic     120.0  229.0  False  lv
hypertrophy
3   37    Male      non-anginal     130.0  250.0  False
normal
4   41  Female  atypical angina     130.0  204.0  False  lv
hypertrophy

   thalch  exang  oldpeak        slope   ca               thal
0   150.0  False      2.3  downsloping  0.0       fixed defect
```

```
1   108.0   True      1.5         flat  3.0             normal
2   129.0   True      2.6         flat  2.0  reversable defect
3   187.0  False      3.5  downsloping  0.0             normal
4   172.0  False      1.4    upsloping  0.0             normal
```

```
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()
x.sex=le.fit_transform(x.sex)
x.cp=le.fit_transform(x.cp)
x.fbs=le.fit_transform(x.fbs)
x.restecg=le.fit_transform(x.restecg)
x.exang=le.fit_transform(x.exang)
x.slope=le.fit_transform(x.slope)
x.thal=le.fit_transform(x.thal)
```

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10,8))
corr=x.corr()
plot=sns.heatmap(corr.round(2),annot=True)
plot.set_title("Correlation map")
```

```
Text(0.5, 1.0, 'Correlation map')
```


```
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

MNB=MultinomialNB()
KNN=KNeighborsClassifier(10)
SVM=SVC(C=3,kernel='poly')
LDS=LinearDiscriminantAnalysis()
DT=DecisionTreeClassifier()

from sklearn.model_selection import train_test_split
xTrain,xTest,yTrain,yTest=train_test_split(x,y,test_size=0.2)

import time
start=time.time()
KNN.fit(xTrain,yTrain)
KNNScore=KNN.score(xTest,yTest)
end=time.time()
KNNTime=end-start
print(f"KNN score:{KNNScore*100}%\nKNN time:{KNNTime} seconds")
```

```
KNN score:50.0%
KNN time:0.018689870834350586 seconds
```

```
start=time.time()
MNB.fit(xTrain,yTrain)
MNBScore=MNB.score(xTest,yTest)
end=time.time()
MNBTime=end-start
print(f"MNB score:{MNBScore*100}%\nMNB time:{MNBTime} seconds")
```

```
MNB score:58.333333333333336%
MNB time:0.017225027084350586 seconds
```

```
start=time.time()
SVM.fit(xTrain,yTrain)
SVMScore=SVM.score(xTest,yTest)
end=time.time()
SVMTime=end-start
print(f"SVM score:{SVMScore*100}%\nSVM time:{SVMTime} seconds")
```

```
SVM score:53.333333333333336%
SVM time:0.030105113983154297 seconds
```

```
start=time.time()
LDS.fit(xTrain,yTrain)
LDSScore=LDS.score(xTest,yTest)
end=time.time()
LDSTime=end-start
print(f"LDS score:{LDSScore*100}%\nLDS time:{LDSTime} seconds")
```

```
LDS score:60.0%
LDS time:0.02342510223388672 seconds
```

```
start=time.time()
DT.fit(xTrain,yTrain)
DTScore=DT.score(xTest,yTest)
end=time.time()
DTTime=end-start
print(f"DT score:{DTScore*100}%\nDT time:{DTTime} seconds")
```

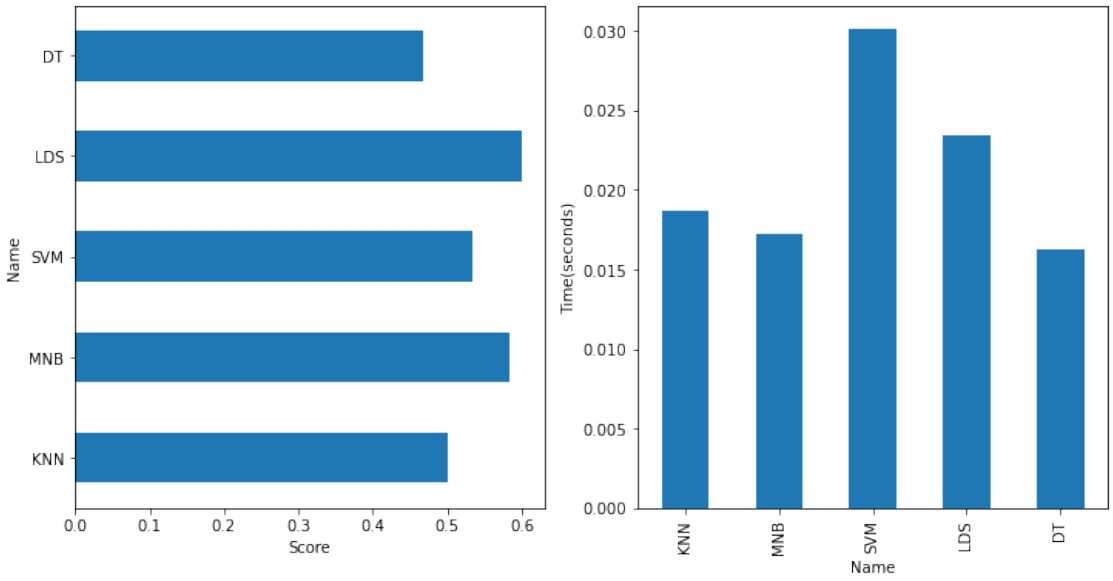
```
DT score:46.666666666666664%
DT time:0.016269207000732422 seconds
```

```
li=[["KNN",KNNScore,KNNTime],['MNB',MNBScore,MNBTime],
['SVM',SVMScore,SVMTime],["LDS",LDSScore,LDSTime],
['DT',DTScore,DTTime]]
newdf=pd.DataFrame(li,columns=["Name","Score","Time"])
newdf
```

```
  Name     Score      Time
0  KNN  0.500000  0.018690
1  MNB  0.583333  0.017225
```

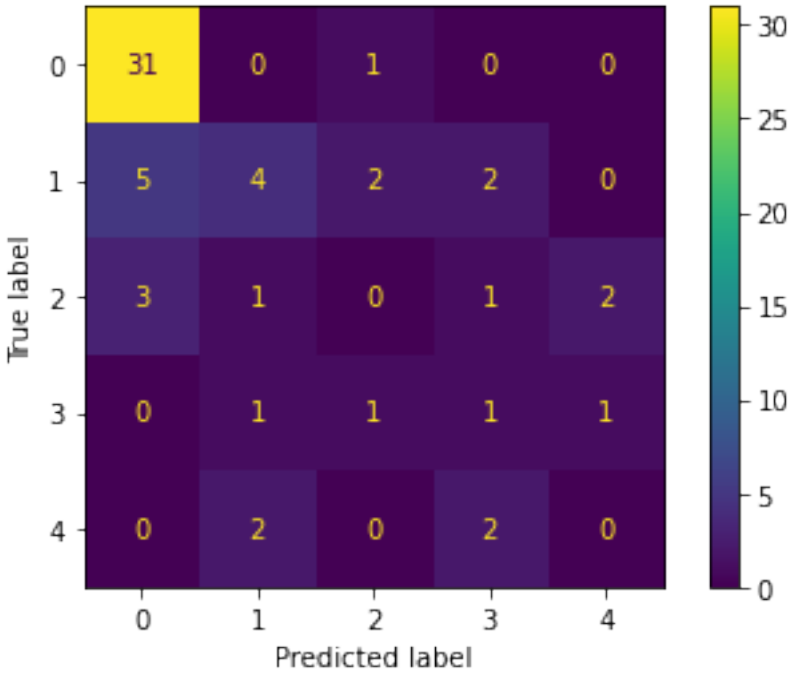
```
2  SVM  0.533333  0.030105
3  LDS  0.600000  0.023425
4   DT  0.466667  0.016269
```

```
fig, ax =plt.subplots(1,2)
newdf.plot.barh(x='Name',y='Score',ax=ax[0],legend=False)
ax[0].set_xlabel("Score")
newdf.plot.bar(x="Name",y="Time",ax=ax[1],legend=False)
ax[1].set_ylabel("Time(seconds)")
fig.set_size_inches(12, 6)
```




```
from sklearn.metrics import ConfusionMatrixDisplay
ConfusionMatrixDisplay.from_predictions(yTest,LDS.predict(xTest))
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7fed692ef8e0>
```

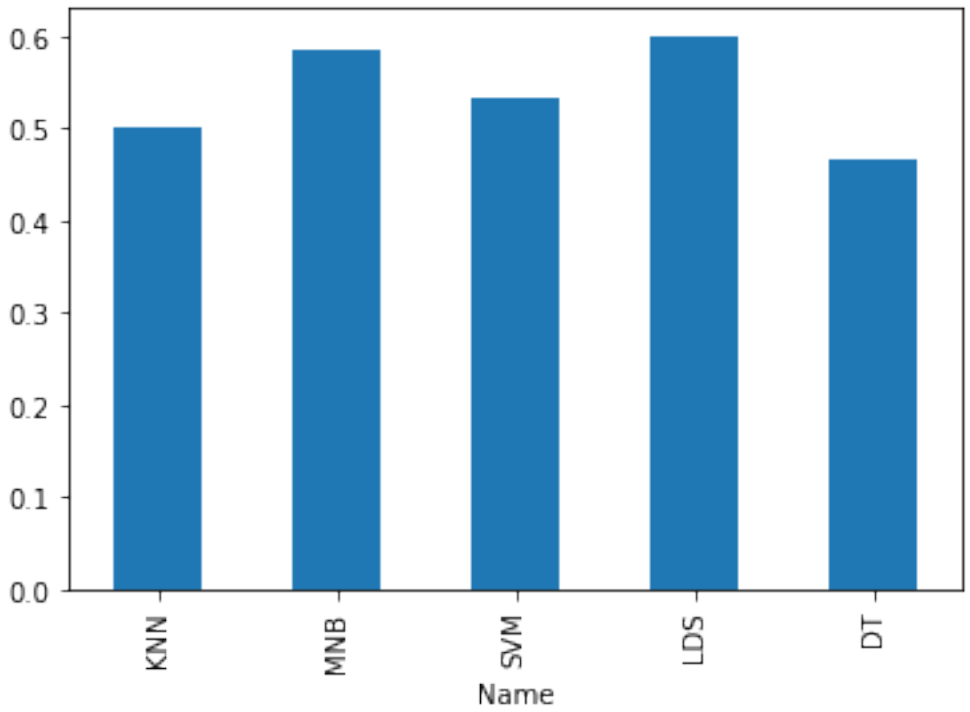



```
from sklearn.metrics import f1_score

KNNf1=f1_score(yTest,KNN.predict(xTest),average='micro')
SVMf1=f1_score(yTest,SVM.predict(xTest),average='micro')
MNBf1=f1_score(yTest,MNB.predict(xTest),average='micro')
LDSf1=f1_score(yTest,LDS.predict(xTest),average='micro')
DTf1=f1_score(yTest,DT.predict(xTest),average='micro')

li=[["KNN",KNNf1],['MNB',MNBf1],['SVM',SVMf1],["LDS",LDSf1],
['DT',DTf1]]
newdf=pd.DataFrame(li,columns=["Name","F1 Score"])
newdf
```

```
  Name  F1 Score
0  KNN  0.500000
1  MNB  0.583333
2  SVM  0.533333
3  LDS  0.600000
4   DT  0.466667
```

```
newdf.plot.bar(x="Name",y="F1 Score",legend=False)
```

```
<AxesSubplot:xlabel='Name'>
```


0.6
0.5
0.4
0.3
0.2
0.1
0.0
KNN
MNB
SVM
LDS
DT
Name