In [ ]:

```
from google.colab import files
uploaded = files.upload()
```

Choose File **No file selected**

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving heart-statlog_csv (1).csv to heart-statlog_csv (1).csv
```

In [ ]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

In [ ]:

```
dt1 = pd.read_csv('heart-statlog_csv (1).csv')
dummy= pd.get_dummies(dt1['class'])
dt = pd.concat((dt1,dummy) , axis=1)
dt = dt.drop(['present','class' ], axis=1)
dt.rename(columns={"absent":"class"}, inplace = True)
dt.head()
```

Out[ ]:

	age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	resting_electrocardiographic_results	maxir
0	**70**	**1**	**4**	**130**	**322**	**0**	**2**	
1	**67**	**0**	**3**	**115**	**564**	**0**	**2**	
2	**57**	**1**	**2**	**124**	**261**	**0**	**0**	
3	**64**	**1**	**4**	**128**	**263**	**0**	**0**	
4	**74**	**0**	**2**	**120**	**269**	**0**	**2**	

In [ ]:

```
X = dt.drop(['class'],axis=1)
y = dt['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.2,shuf
fle=True, random_state=5)
```

In [ ]:

```
num_feats=11
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import RandomForestClassifier

embeded_rf_selector = SelectFromModel(RandomForestClassifier(n_estimators=100, criterion=
'gini'), max_features=num_feats)
embeded_rf_selector.fit(X, y)

embeded_rf_support = embeded_rf_selector.get_support()
embeded_rf_feature = X.loc[:,embeded_rf_support].columns.tolist()
print(str(len(embeded_rf_feature)), 'selected features')
print(embeded_rf_feature)
```

```
8 selected features
```

['age', 'chest', 'resting_blood_pressure', 'serum_cholestoral', 'maximum_heart_rate_achieved', 'oldpeak', 'number_of_major_vessels', 'thal']

In [ ]:

```
X = dt.drop(['class','fasting_blood_sugar','sex','resting_electrocardiographic_results',
'exercise_induced_angina'],axis=1)
y = dt['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.80)
#for a in range(1, 40):
k =10
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train,y_train)
y_pred_knn = knn.predict(X_test)
```

In [ ]:

```
from sklearn import metrics
from sklearn.metrics import classification_report, confusion_matrix,accuracy_score,log_lo
ss , roc_auc_score,  precision_score, recall_score, f1_score, matthews_corrcoef
CM=confusion_matrix(y_test,y_pred_knn)
sns.heatmap(CM, annot=True)
TN = CM[0][0]
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
specificity = TN/(TN+FP)
loss_log = log_loss(y_test, y_pred_knn)
acc= accuracy_score(y_test,y_pred_knn)
roc=roc_auc_score(y_test, y_pred_knn)
prec = precision_score(y_test, y_pred_knn)
rec = recall_score(y_test, y_pred_knn)
f1 = f1_score(y_test, y_pred_knn)
mathew = matthews_corrcoef(y_test,y_pred_knn)
model_results =pd.DataFrame([['FS+KNN',acc, prec,rec,specificity, f1,roc,mathew,loss_log]
])
model_results.columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity' ,'
F1-Score','Recall Score','Mathew coefficient','log_loss']
model_results.head()
```

Out[ ]:

	Model	Accuracy	Precision	Sensitivity	Specificity	F1-Score	Recall Score	Mathew coefficient	log_loss
0	FS+KNN	0.722222	0.84	0.65625	0.818182	0.736842	0.737216	0.46751	9.594164