Heart attack

September 12, 2020

```
[107]: import numpy as np
       import pandas as pd
       import seaborn as sns
       import matplotlib.pyplot as plt
       import time
       from sklearn.model_selection import train_test_split
       from sklearn.ensemble import RandomForestClassifier
       from sklearn.metrics import f1_score,confusion_matrix
       from sklearn.metrics import accuracy_score
       from sklearn.feature_selection import SelectKBest
       from sklearn.feature_selection import chi2
       from sklearn.feature_selection import RFE
       from sklearn.datasets import make_regression


       def read_data(csv_file):
           try:
               return pd.read_csv(csv_file)
           except:
               print("The file is not found")
               return None



       Heart_attack_data_set = read_data("C:/Users/omri1/PycharmProjects/untitled2/
        ↪heart_attack.csv")
```

```
[108]: Heart_attack_data_set
```

```
[108]:        age  anaemia  creatinine_phosphokinase  diabetes  ejection_fraction  \
       0      75.0        0                       582         0                 20
       1      55.0        0                      7861         0                 38
       2      65.0        0                       146         0                 20
       3      50.0        1                       111         0                 20
       4      65.0        1                       160         1                 20
       ..      ...      ...                       ...       ...                ...
       294    62.0        0                        61         1                 38
       295    55.0        0                      1820         0                 38
```

```
296  45.0         0                          2060           1                      60
297  45.0         0                          2413           0                      38
298  50.0         0                           196           0                      45

     high_blood_pressure  platelets  serum_creatinine  serum_sodium  sex  \
0                      1  265000.00               1.9           130    1
1                      0  263358.03               1.1           136    1
2                      0  162000.00               1.3           129    1
3                      0  210000.00               1.9           137    1
4                      0  327000.00               2.7           116    0
..                   ...        ...               ...           ...  ...
294                    1  155000.00               1.1           143    1
295                    0  270000.00               1.2           139    0
296                    0  742000.00               0.8           138    0
297                    0  140000.00               1.4           140    1
298                    0  395000.00               1.6           136    1

     smoking  time  DEATH_EVENT
0          0     4            1
1          0     6            1
2          1     7            1
3          0     7            1
4          0     8            1
..       ...   ...          ...
294        1   270            0
295        0   271            0
296        0   278            0
297        1   280            0
298        1   285            0

[299 rows x 13 columns]
```

```python
[109]:  # Statistical analysis

        Heart_attack_data_set.describe()
```

```
[109]:              age      anaemia  creatinine_phosphokinase     diabetes  \
       count  299.000000  299.000000                299.000000  299.000000
       mean    60.833893    0.431438                581.839465    0.418060
       std     11.894809    0.496107                970.287881    0.494067
       min     40.000000    0.000000                 23.000000    0.000000
       25%     51.000000    0.000000                116.500000    0.000000
       50%     60.000000    0.000000                250.000000    0.000000
       75%     70.000000    1.000000                582.000000    1.000000
       max     95.000000    1.000000               7861.000000    1.000000

              ejection_fraction  high_blood_pressure        platelets  \
```

```
count        299.000000          299.000000       299.000000
mean          38.083612            0.351171   263358.029264
std           11.834841            0.478136    97804.236869
min           14.000000            0.000000    25100.000000
25%           30.000000            0.000000   212500.000000
50%           38.000000            0.000000   262000.000000
75%           45.000000            1.000000   303500.000000
max           80.000000            1.000000   850000.000000

       serum_creatinine  serum_sodium         sex    smoking         time  \
count         299.00000    299.000000  299.000000  299.00000   299.000000
mean            1.39388    136.625418    0.648829    0.32107   130.260870
std             1.03451      4.412477    0.478136    0.46767    77.614208
min             0.50000    113.000000    0.000000    0.00000     4.000000
25%             0.90000    134.000000    0.000000    0.00000    73.000000
50%             1.10000    137.000000    1.000000    0.00000   115.000000
75%             1.40000    140.000000    1.000000    1.00000   203.000000
max             9.40000    148.000000    1.000000    1.00000   285.000000

       DEATH_EVENT
count    299.00000
mean       0.32107
std        0.46767
min        0.00000
25%        0.00000
50%        0.00000
75%        1.00000
max        1.00000
```

```python
[110]: Heart_attack_data_set['sex'].value_counts().plot(kind="bar", title="Sex␣
        ↪Variable Distribution", alpha=0.5)
       plt.show()
```

Sex Variable Distribution
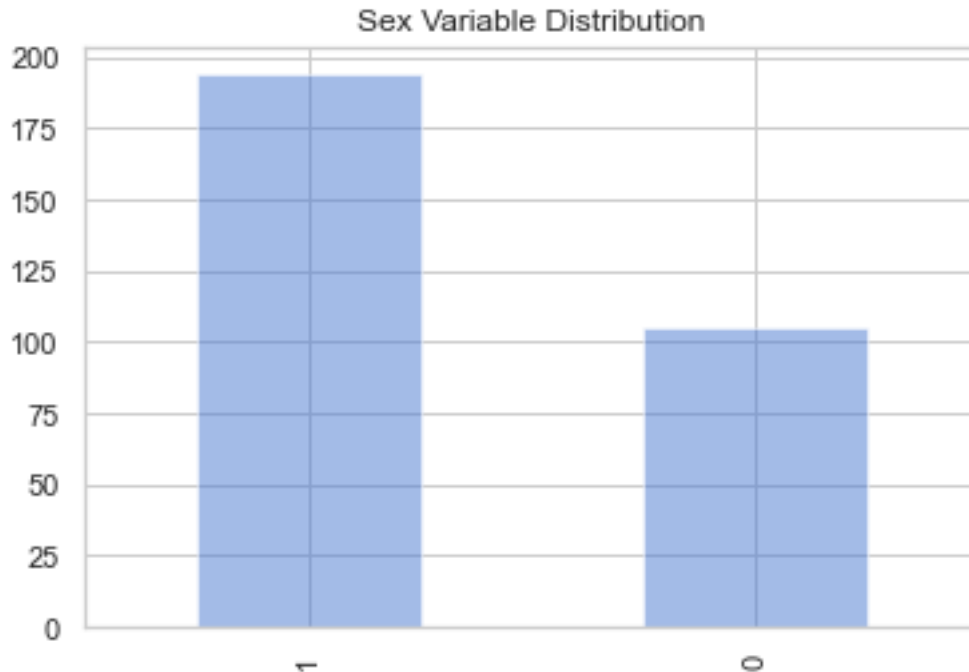

```
[111]:  def data_shape(data, label):
            print('Rows number of ' + label + " is: ", data.shape[0])
            print('Columns number of ' + label + ' is: ', data.shape[1])

        def data_columns(data):
            return list(data.columns)

        def describe_data(data):
            return data.describe()

        data_shape(Heart_attack_data_set, 'Heart attack data set')
        data_columns(Heart_attack_data_set)
        describe_data(Heart_attack_data_set)
```

```
Rows number of Heart attack data set is:  299
Columns number of Heart attack data set is:  13
```

[111]:

	age	anaemia	creatinine_phosphokinase	diabetes \
count	299.000000	299.000000	299.000000	299.000000
mean	60.833893	0.431438	581.839465	0.418060
std	11.894809	0.496107	970.287881	0.494067
min	40.000000	0.000000	23.000000	0.000000
25%	51.000000	0.000000	116.500000	0.000000
50%	60.000000	0.000000	250.000000	0.000000
75%	70.000000	1.000000	582.000000	1.000000

4

```
max       95.000000      1.000000                    7861.000000      1.000000
```

	ejection_fraction	high_blood_pressure	platelets	\
count	299.000000	299.000000	299.000000	
mean	38.083612	0.351171	263358.029264	
std	11.834841	0.478136	97804.236869	
min	14.000000	0.000000	25100.000000	
25%	30.000000	0.000000	212500.000000	
50%	38.000000	0.000000	262000.000000	
75%	45.000000	1.000000	303500.000000	
max	80.000000	1.000000	850000.000000	

	serum_creatinine	serum_sodium	sex	smoking	time	\
count	299.00000	299.000000	299.000000	299.00000	299.000000	
mean	1.39388	136.625418	0.648829	0.32107	130.260870	
std	1.03451	4.412477	0.478136	0.46767	77.614208	
min	0.50000	113.000000	0.000000	0.00000	4.000000	
25%	0.90000	134.000000	0.000000	0.00000	73.000000	
50%	1.10000	137.000000	1.000000	0.00000	115.000000	
75%	1.40000	140.000000	1.000000	1.00000	203.000000	
max	9.40000	148.000000	1.000000	1.00000	285.000000	

	DEATH_EVENT
count	299.00000
mean	0.32107
std	0.46767
min	0.00000
25%	0.00000
50%	0.00000
75%	1.00000
max	1.00000

```
[112]: sns.set(style="whitegrid", palette="muted")
       new_data = (Heart_attack_data_set - Heart_attack_data_set.mean()) /␣
        ↪(Heart_attack_data_set.std())
       new_data = pd.concat([Heart_attack_data_set['DEATH_EVENT'], new_data.iloc[:,0:
        ↪12]], axis=1)
       new_data
```

[112]:	DEATH_EVENT	age	anaemia	creatinine_phosphokinase	diabetes	\
0	1	1.190949	-0.869647	0.000165	-0.846161	
1	1	-0.490457	-0.869647	7.502063	-0.846161	
2	1	0.350246	-0.869647	-0.449186	-0.846161	
3	1	-0.910808	1.146046	-0.485257	-0.846161	
4	1	0.350246	1.146046	-0.434757	1.177856	
..	...	...	...	...	...	
294	0	0.098035	-0.869647	-0.536789	1.177856	

```
295              0 -0.490457 -0.869647              1.276075 -0.846161
296              0 -1.331160 -0.869647              1.523425  1.177856
297              0 -1.331160 -0.869647              1.887234 -0.846161
298              0 -0.910808 -0.869647             -0.397655 -0.846161

     ejection_fraction  high_blood_pressure      platelets  serum_creatinine  \
0            -1.527998             1.356997   1.678834e-02          0.489237
1            -0.007065            -0.734457   7.523047e-09         -0.284076
2            -1.527998            -0.734457  -1.036336e+00         -0.090748
3            -1.527998            -0.734457  -5.455595e-01          0.489237
4            -1.527998            -0.734457   6.507077e-01          1.262550
..                 ...                  ...            ...               ...
294          -0.007065             1.356997  -1.107907e+00         -0.284076
295          -0.007065            -0.734457   6.791087e-02         -0.187412
296           1.851853            -0.734457   4.893878e+00         -0.574068
297          -0.007065            -0.734457  -1.261275e+00          0.005916
298           0.584409            -0.734457   1.345974e+00          0.199244

     serum_sodium       sex   smoking      time
0       -1.501519  0.734457 -0.686531 -1.626775
1       -0.141739  0.734457 -0.686531 -1.601007
2       -1.728149  0.734457  1.451727 -1.588122
3        0.084892  0.734457 -0.686531 -1.588122
4       -4.674340 -1.356997 -0.686531 -1.575238
..            ...       ...       ...       ...
294      1.444672  0.734457  1.451727  1.800432
295      0.538152 -1.356997 -0.686531  1.813317
296      0.311522 -1.356997 -0.686531  1.903506
297      0.764782  0.734457  1.451727  1.929275
298     -0.141739  0.734457  1.451727  1.993696

[299 rows x 13 columns]
```

```
[113]: new_data = pd.melt(new_data, id_vars="DEATH_EVENT", var_name="features",
        ↪value_name='value')
       new_data
```

```
[113]:       DEATH_EVENT features      value
       0               1      age   1.190949
       1               1      age  -0.490457
       2               1      age   0.350246
       3               1      age  -0.910808
       4               1      age   0.350246
       ...           ...      ...        ...
       3583            0     time   1.800432
       3584            0     time   1.813317
       3585            0     time   1.903506
```

```
3586                0       time  1.929275
3587                0       time  1.993696

[3588 rows x 3 columns]
```

```
[114]:  # 0 = Death
        plt.figure(figsize=(10,10))
        sns.swarmplot(x="features", y="value", hue="DEATH_EVENT", data=new_data)
        plt.xticks(rotation=90)
```

```
[114]:  (array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11]),
         [Text(0, 0, 'age'),
          Text(1, 0, 'anaemia'),
          Text(2, 0, 'creatinine_phosphokinase'),
          Text(3, 0, 'diabetes'),
          Text(4, 0, 'ejection_fraction'),
          Text(5, 0, 'high_blood_pressure'),
          Text(6, 0, 'platelets'),
          Text(7, 0, 'serum_creatinine'),
          Text(8, 0, 'serum_sodium'),
          Text(9, 0, 'sex'),
          Text(10, 0, 'smoking'),
          Text(11, 0, 'time')])
```
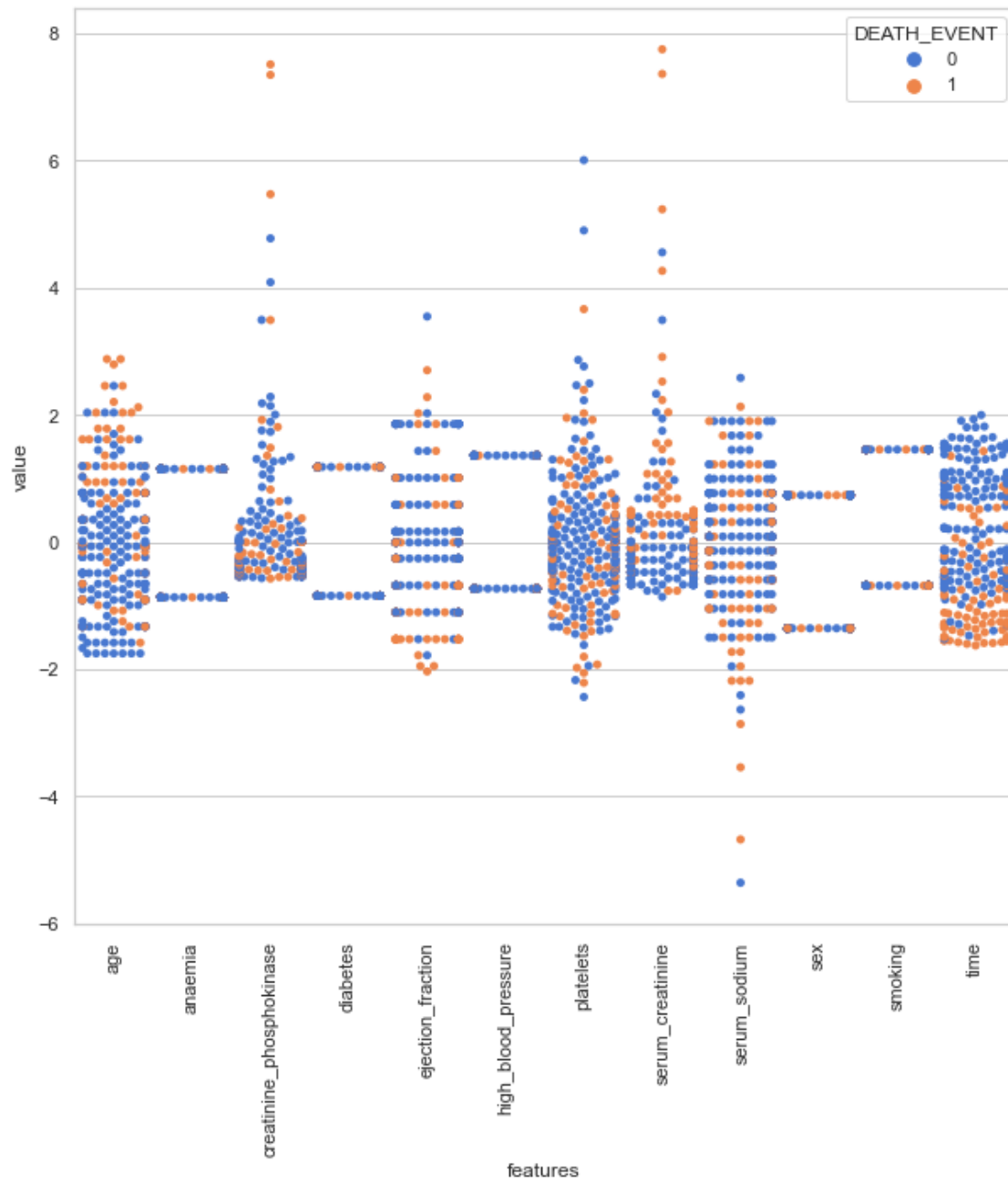
```
[115]: fig, ax = plt.subplots(figsize=(12,12))
       sns.heatmap(Heart_attack_data_set.corr(), annot = True, ax=ax)
       plt.title('Relations between columns', fontsize = 20)
       plt.show()
```

Relations between columns

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
age	1	0.088	-0.082	-0.1	0.06	0.093	-0.052	0.16	-0.046	0.065	0.019	-0.22	0.25
anaemia	0.088	1	-0.19	-0.013	0.032	0.038	-0.044	0.052	0.042	-0.095	-0.11	-0.14	0.066
creatinine_phosphokinase	-0.082	-0.19	1	-0.0096	-0.044	-0.071	0.024	-0.016	0.06	0.08	0.0024	-0.0093	0.063
diabetes	-0.1	-0.013	-0.0096	1	-0.0049	-0.013	0.092	-0.047	-0.09	-0.16	-0.15	0.034	-0.0019
ejection_fraction	0.06	0.032	-0.044	-0.0049	1	0.024	0.072	-0.011	0.18	-0.15	-0.067	0.042	-0.27
high_blood_pressure	0.093	0.038	-0.071	-0.013	0.024	1	0.05	-0.0049	0.037	-0.1	-0.056	-0.2	0.079
platelets	-0.052	-0.044	0.024	0.092	0.072	0.05	1	-0.041	0.062	-0.13	0.028	0.011	-0.049
serum_creatinine	0.16	0.052	-0.016	-0.047	-0.011	-0.0049	-0.041	1	-0.19	0.007	-0.027	-0.15	0.29
serum_sodium	-0.046	0.042	0.06	-0.09	0.18	0.037	0.062	-0.19	1	-0.028	0.0048	0.088	-0.2
sex	0.065	-0.095	0.08	-0.16	-0.15	-0.1	-0.13	0.007	-0.028	1	0.45	-0.016	-0.0043
smoking	0.019	-0.11	0.0024	-0.15	-0.067	-0.056	0.028	-0.027	0.0048	0.45	1	-0.023	-0.013
time	-0.22	-0.14	-0.0093	0.034	0.042	-0.2	0.011	-0.15	0.088	-0.016	-0.023	1	-0.53
DEATH_EVENT	0.25	0.066	0.063	-0.0019	-0.27	0.079	-0.049	0.29	-0.2	-0.0043	-0.013	-0.53	1

```
[116]: drop_list = ['time', 'DEATH_EVENT']
       fs_corr = Heart_attack_data_set.drop(columns=drop_list)
       fs_corr.head()
```

```
[116]:    age  anaemia  creatinine_phosphokinase  diabetes  ejection_fraction  \
       0  75.0        0                       582         0                 20
       1  55.0        0                      7861         0                 38
       2  65.0        0                       146         0                 20
       3  50.0        1                       111         0                 20
       4  65.0        1                       160         1                 20
```

```
      high_blood_pressure  platelets  serum_creatinine  serum_sodium  sex  \
0                       1  265000.00               1.9           130    1
1                       0  263358.03               1.1           136    1
2                       0  162000.00               1.3           129    1
3                       0  210000.00               1.9           137    1
4                       0  327000.00               2.7           116    0

    smoking
0        0
1        0
2        1
3        0
4        0
```

```
[117]:  y = Heart_attack_data_set['DEATH_EVENT']
        x_train, x_test, y_train, y_test = train_test_split(fs_corr, y, test_size=0.
         →2,random_state=42)
        clf_rf = RandomForestClassifier(n_estimators=20)
        clr_rf = clf_rf.fit(x_train,y_train)
```

```
[118]:  print('Accuracy',accuracy_score(y_test,clf_rf.predict(x_test)))
        cm = confusion_matrix(y_test,clf_rf.predict(x_test))
        sns.heatmap(cm,annot=True,fmt="d")
```

Accuracy 0.65

[118]: <AxesSubplot:>
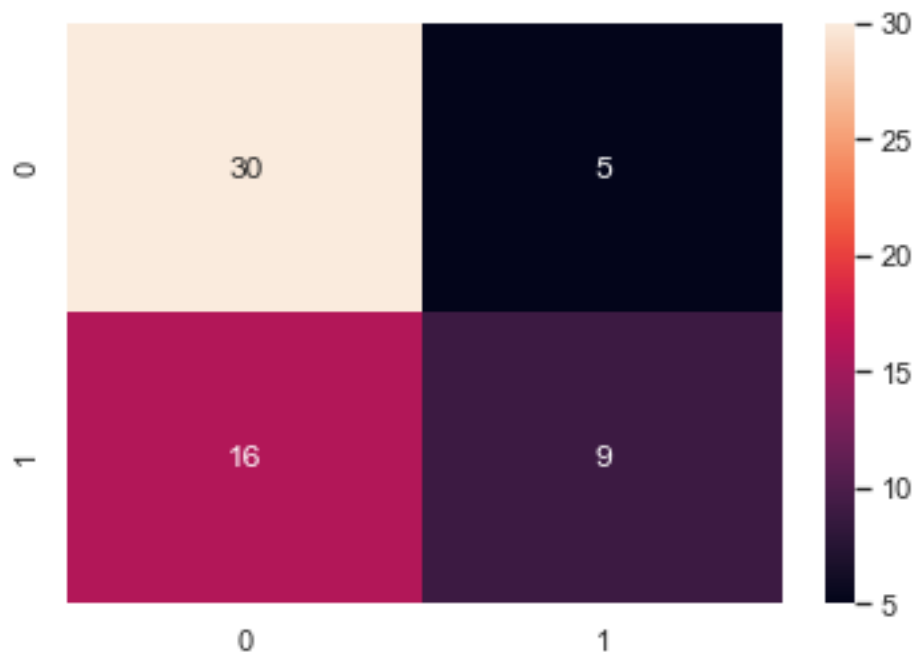
```python
[119]: K = range(1, len(x_train.columns))
       for k in K:
           select_feature = SelectKBest(chi2, k=k).fit(x_train, y_train)
           scores = zip(x_train.columns, select_feature.scores_)
           print("Selected K:", k)
           for i, (column, score) in enumerate(scores):
               if i < k:
                   print("Feature:", column, ", Score:", score)
           print("-----------")
```

```
Selected K: 1
Feature: age , Score: 46.9889849693994
-----------
Selected K: 2
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
-----------
Selected K: 3
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
-----------
Selected K: 4
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
-----------
Selected K: 5
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
Feature: ejection_fraction , Score: 55.896406551208116
-----------
Selected K: 6
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
Feature: ejection_fraction , Score: 55.896406551208116
Feature: high_blood_pressure , Score: 0.5289514866979651
-----------
Selected K: 7
Feature: age , Score: 46.9889849693994
```

```
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
Feature: ejection_fraction , Score: 55.896406551208116
Feature: high_blood_pressure , Score: 0.5289514866979651
Feature: platelets , Score: 27714.885624462317
-----------
Selected K: 8
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
Feature: ejection_fraction , Score: 55.896406551208116
Feature: high_blood_pressure , Score: 0.5289514866979651
Feature: platelets , Score: 27714.885624462317
Feature: serum_creatinine , Score: 18.105974482139235
-----------
Selected K: 9
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
Feature: ejection_fraction , Score: 55.896406551208116
Feature: high_blood_pressure , Score: 0.5289514866979651
Feature: platelets , Score: 27714.885624462317
Feature: serum_creatinine , Score: 18.105974482139235
Feature: serum_sodium , Score: 1.2352740846996069
-----------
Selected K: 10
Feature: age , Score: 46.9889849693994
Feature: anaemia , Score: 0.44381997110870225
Feature: creatinine_phosphokinase , Score: 460.053375481774
Feature: diabetes , Score: 0.0016732698597080864
Feature: ejection_fraction , Score: 55.896406551208116
Feature: high_blood_pressure , Score: 0.5289514866979651
Feature: platelets , Score: 27714.885624462317
Feature: serum_creatinine , Score: 18.105974482139235
Feature: serum_sodium , Score: 1.2352740846996069
Feature: sex , Score: 0.23498621599728284
-----------
```

```python
[120]: clf_rf_ = RandomForestClassifier(n_estimators=20)
       rfe = RFE(estimator=clf_rf_, n_features_to_select=5, step=1)
       rfe = rfe.fit(x_train, y_train)
```

```python
[121]: print('Chosen best 5 feature by RFE:',x_train.columns[rfe.support_])
```

```
Chosen best 5 feature by RFE: Index(['age', 'creatinine_phosphokinase',
```

```
          'ejection_fraction', 'platelets',
              'serum_creatinine'],
            dtype='object')
```

```
[122]:  clf_rf_ = RandomForestClassifier(n_estimators=20)
        clr_rf_ = clf_rf_.fit(x_train,y_train)
        importances = clr_rf_.feature_importances_
        std = np.std([tree.feature_importances_ for tree in clf_rf.estimators_],axis=0)
        indices = np.argsort(importances)[::-1]
```

```
[123]:  print("Feature ranking:")
        for f in range(x_train.shape[1]):
            print("%d. feature %d (%f)" % (f + 1, indices[f], importances[indices[f]]))
```

```
Feature ranking:
1. feature 7 (0.207291)
2. feature 4 (0.187294)
3. feature 0 (0.154374)
4. feature 2 (0.135270)
5. feature 8 (0.116074)
6. feature 6 (0.109168)
7. feature 3 (0.023244)
8. feature 1 (0.019160)
9. feature 9 (0.018682)
10. feature 5 (0.016550)
11. feature 10 (0.012894)
```

```
[124]:  plt.figure(1, figsize=(8, 8))
        plt.title("Feature importances")
        plt.bar(range(x_train.shape[1]), importances[indices],
        color="g", yerr=std[indices], align="center")
        plt.xticks(range(x_train.shape[1]), x_train.columns[indices],rotation=90)
        plt.xlim([-1, x_train.shape[1]])
        plt.show()
```

Feature importances
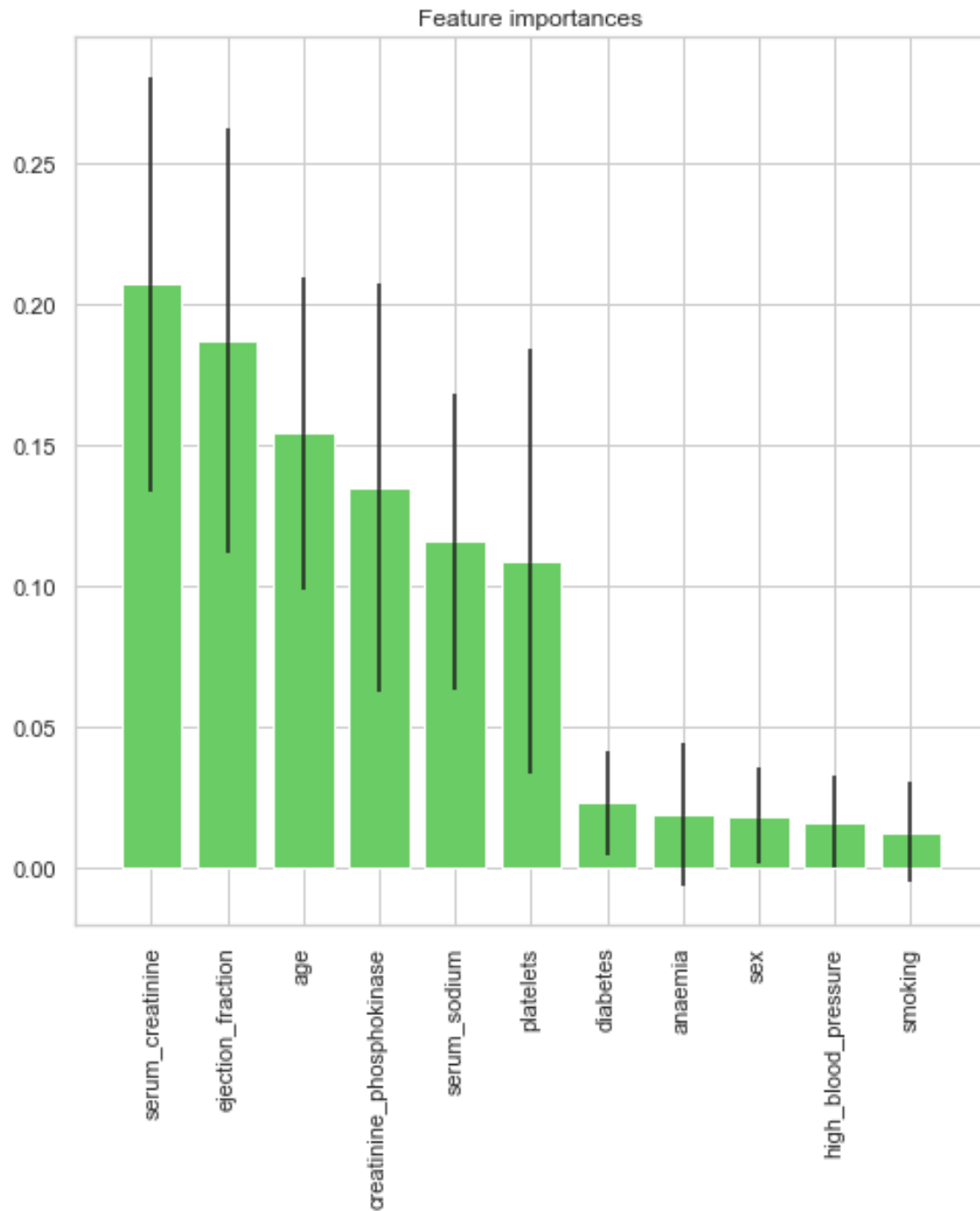

```
[125]: #SVM algorithm
       from sklearn.svm import SVC
       from sklearn.metrics import classification_report,confusion_matrix
       df_feat = Heart_attack_data_set
       df_feat.info()
```

```
X_train, X_test, y_train, y_test = train_test_split(df_feat, np.ravel(y),␣
 →test_size=0.30, random_state=101)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 299 entries, 0 to 298
Data columns (total 13 columns):
age                         299 non-null float64
anaemia                     299 non-null int64
creatinine_phosphokinase    299 non-null int64
diabetes                    299 non-null int64
ejection_fraction           299 non-null int64
high_blood_pressure         299 non-null int64
platelets                   299 non-null float64
serum_creatinine            299 non-null float64
serum_sodium                299 non-null int64
sex                         299 non-null int64
smoking                     299 non-null int64
time                        299 non-null int64
DEATH_EVENT                 299 non-null int64
dtypes: float64(3), int64(10)
memory usage: 30.5 KB
```

[126]:
```
model = SVC()
model.fit(X_train,y_train)
```

[126]: SVC()

[127]:
```
predictions = model.predict(X_test)
print(confusion_matrix(y_test,predictions))
```

```
[[62  0]
 [28  0]]
```

[128]: `print(classification_report(y_test,predictions))`

```
              precision    recall  f1-score   support

           0       0.69      1.00      0.82        62
           1       0.00      0.00      0.00        28

    accuracy                           0.69        90
   macro avg       0.34      0.50      0.41        90
weighted avg       0.47      0.69      0.56        90
```

```
C:\Users\omri1\Anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1221: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
```

```
        predicted samples. Use `zero_division` parameter to control this behavior.
          _warn_prf(average, modifier, msg_start, len(result))
```

```
[129]: #Gridsearch
       from sklearn.model_selection import GridSearchCV
       param_grid = {'C': [0.1,1, 10, 100, 1000], 'gamma': [1,0.1,0.01,0.001,0.0001],␣
       ↪'kernel': ['rbf']}
       grid = GridSearchCV(SVC(),param_grid,refit=True,verbose=3)
       grid.fit(X_train,y_train)
```

```
Fitting 5 folds for each of 25 candidates, totalling 125 fits
[CV] C=0.1, gamma=1, kernel=rbf …
[CV] … C=0.1, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=1, kernel=rbf …
[CV] … C=0.1, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=1, kernel=rbf …
[CV] … C=0.1, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=1, kernel=rbf …
[CV] … C=0.1, gamma=1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=0.1, gamma=1, kernel=rbf …
[CV] … C=0.1, gamma=1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf …
[CV] … C=0.1, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf …

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   0.0s remaining:   0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:   0.0s remaining:   0.0s

[CV] … C=0.1, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf …
[CV] … C=0.1, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf …
[CV] … C=0.1, gamma=0.1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf …
[CV] … C=0.1, gamma=0.1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf …
[CV] … C=0.1, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf …
[CV] … C=0.1, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf …
[CV] … C=0.1, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf …
[CV] … C=0.1, gamma=0.01, kernel=rbf, score=0.690, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf …
[CV] … C=0.1, gamma=0.01, kernel=rbf, score=0.683, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf …
[CV] … C=0.1, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf …
```

```
[CV] … C=0.1, gamma=0.001, kernel=rbf, score=0.667, total=    0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf …
[CV] … C=0.1, gamma=0.001, kernel=rbf, score=0.667, total=    0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf …
[CV] … C=0.1, gamma=0.001, kernel=rbf, score=0.690, total=    0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf …
[CV] … C=0.1, gamma=0.001, kernel=rbf, score=0.683, total=    0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf …
[CV] … C=0.1, gamma=0.0001, kernel=rbf, score=0.667, total=    0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf …
[CV] … C=0.1, gamma=0.0001, kernel=rbf, score=0.667, total=    0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf …
[CV] … C=0.1, gamma=0.0001, kernel=rbf, score=0.667, total=    0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf …
[CV] … C=0.1, gamma=0.0001, kernel=rbf, score=0.690, total=    0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf …
[CV] … C=0.1, gamma=0.0001, kernel=rbf, score=0.683, total=    0.0s
[CV] C=1, gamma=1, kernel=rbf …
[CV] … C=1, gamma=1, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=1, kernel=rbf …
[CV] … C=1, gamma=1, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=1, kernel=rbf …
[CV] … C=1, gamma=1, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=1, kernel=rbf …
[CV] … C=1, gamma=1, kernel=rbf, score=0.690, total=    0.0s
[CV] C=1, gamma=1, kernel=rbf …
[CV] … C=1, gamma=1, kernel=rbf, score=0.683, total=    0.0s
[CV] C=1, gamma=0.1, kernel=rbf …
[CV] … C=1, gamma=0.1, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=0.1, kernel=rbf …
[CV] … C=1, gamma=0.1, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=0.1, kernel=rbf …
[CV] … C=1, gamma=0.1, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=0.1, kernel=rbf …
[CV] … C=1, gamma=0.1, kernel=rbf, score=0.690, total=    0.0s
[CV] C=1, gamma=0.1, kernel=rbf …
[CV] … C=1, gamma=0.1, kernel=rbf, score=0.683, total=    0.0s
[CV] C=1, gamma=0.01, kernel=rbf …
[CV] … C=1, gamma=0.01, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=0.01, kernel=rbf …
[CV] … C=1, gamma=0.01, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=0.01, kernel=rbf …
[CV] … C=1, gamma=0.01, kernel=rbf, score=0.667, total=    0.0s
[CV] C=1, gamma=0.01, kernel=rbf …
[CV] … C=1, gamma=0.01, kernel=rbf, score=0.690, total=    0.0s
[CV] C=1, gamma=0.01, kernel=rbf …
[CV] … C=1, gamma=0.01, kernel=rbf, score=0.683, total=    0.0s
[CV] C=1, gamma=0.001, kernel=rbf …
```

```
[CV] … C=1, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf …
[CV] … C=1, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf …
[CV] … C=1, gamma=0.001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf …
[CV] … C=1, gamma=0.001, kernel=rbf, score=0.714, total=   0.0s
[CV] C=1, gamma=0.001, kernel=rbf …
[CV] … C=1, gamma=0.001, kernel=rbf, score=0.707, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf …
[CV] … C=1, gamma=0.0001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf …
[CV] … C=1, gamma=0.0001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf …
[CV] … C=1, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf …
[CV] … C=1, gamma=0.0001, kernel=rbf, score=0.738, total=   0.0s
[CV] C=1, gamma=0.0001, kernel=rbf …
[CV] … C=1, gamma=0.0001, kernel=rbf, score=0.707, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf …
[CV] … C=10, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf …
[CV] … C=10, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf …
[CV] … C=10, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf …
[CV] … C=10, gamma=1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=10, gamma=1, kernel=rbf …
[CV] … C=10, gamma=1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf …
[CV] … C=10, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf …
[CV] … C=10, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf …
[CV] … C=10, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf …
[CV] … C=10, gamma=0.1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=10, gamma=0.1, kernel=rbf …
[CV] … C=10, gamma=0.1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf …
[CV] … C=10, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf …
[CV] … C=10, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf …
[CV] … C=10, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf …
[CV] … C=10, gamma=0.01, kernel=rbf, score=0.690, total=   0.0s
[CV] C=10, gamma=0.01, kernel=rbf …
```

```
[CV] … C=10, gamma=0.01, kernel=rbf, score=0.683, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf …
[CV] … C=10, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf …
[CV] … C=10, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf …
[CV] … C=10, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf …
[CV] … C=10, gamma=0.001, kernel=rbf, score=0.714, total=   0.0s
[CV] C=10, gamma=0.001, kernel=rbf …
[CV] … C=10, gamma=0.001, kernel=rbf, score=0.707, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf …
[CV] … C=10, gamma=0.0001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf …
[CV] … C=10, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf …
[CV] … C=10, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf …
[CV] … C=10, gamma=0.0001, kernel=rbf, score=0.762, total=   0.0s
[CV] C=10, gamma=0.0001, kernel=rbf …
[CV] … C=10, gamma=0.0001, kernel=rbf, score=0.683, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf …
[CV] … C=100, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf …
[CV] … C=100, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf …
[CV] … C=100, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf …
[CV] … C=100, gamma=1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=100, gamma=1, kernel=rbf …
[CV] … C=100, gamma=1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf …
[CV] … C=100, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf …
[CV] … C=100, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf …
[CV] … C=100, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf …
[CV] … C=100, gamma=0.1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=100, gamma=0.1, kernel=rbf …
[CV] … C=100, gamma=0.1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf …
[CV] … C=100, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf …
[CV] … C=100, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf …
[CV] … C=100, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf …
```

```
[CV] … C=100, gamma=0.01, kernel=rbf, score=0.690, total=   0.0s
[CV] C=100, gamma=0.01, kernel=rbf …
[CV] … C=100, gamma=0.01, kernel=rbf, score=0.683, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf …
[CV] … C=100, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf …
[CV] … C=100, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf …
[CV] … C=100, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf …
[CV] … C=100, gamma=0.001, kernel=rbf, score=0.714, total=   0.0s
[CV] C=100, gamma=0.001, kernel=rbf …
[CV] … C=100, gamma=0.001, kernel=rbf, score=0.707, total=   0.0s
[CV] C=100, gamma=0.0001, kernel=rbf …
[CV] … C=100, gamma=0.0001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=100, gamma=0.0001, kernel=rbf …
[CV] … C=100, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=100, gamma=0.0001, kernel=rbf …
[CV] … C=100, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=100, gamma=0.0001, kernel=rbf …
[CV] … C=100, gamma=0.0001, kernel=rbf, score=0.762, total=   0.0s
[CV] C=100, gamma=0.0001, kernel=rbf …
[CV] … C=100, gamma=0.0001, kernel=rbf, score=0.683, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf …
[CV] … C=1000, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf …
[CV] … C=1000, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf …
[CV] … C=1000, gamma=1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf …
[CV] … C=1000, gamma=1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1000, gamma=1, kernel=rbf …
[CV] … C=1000, gamma=1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf …
[CV] … C=1000, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf …
[CV] … C=1000, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf …
[CV] … C=1000, gamma=0.1, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf …
[CV] … C=1000, gamma=0.1, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1000, gamma=0.1, kernel=rbf …
[CV] … C=1000, gamma=0.1, kernel=rbf, score=0.683, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf …
[CV] … C=1000, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf …
[CV] … C=1000, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf …
```

```
[CV] … C=1000, gamma=0.01, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf …
[CV] … C=1000, gamma=0.01, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1000, gamma=0.01, kernel=rbf …
[CV] … C=1000, gamma=0.01, kernel=rbf, score=0.683, total=   0.0s
[CV] C=1000, gamma=0.001, kernel=rbf …
[CV] … C=1000, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.001, kernel=rbf …
[CV] … C=1000, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.001, kernel=rbf …
[CV] … C=1000, gamma=0.001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.001, kernel=rbf …
[CV] … C=1000, gamma=0.001, kernel=rbf, score=0.714, total=   0.0s
[CV] C=1000, gamma=0.001, kernel=rbf …
[CV] … C=1000, gamma=0.001, kernel=rbf, score=0.707, total=   0.0s
[CV] C=1000, gamma=0.0001, kernel=rbf …
[CV] … C=1000, gamma=0.0001, kernel=rbf, score=0.667, total=   0.0s
[CV] C=1000, gamma=0.0001, kernel=rbf …
[CV] … C=1000, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1000, gamma=0.0001, kernel=rbf …
[CV] … C=1000, gamma=0.0001, kernel=rbf, score=0.690, total=   0.0s
[CV] C=1000, gamma=0.0001, kernel=rbf …
[CV] … C=1000, gamma=0.0001, kernel=rbf, score=0.762, total=   0.0s
[CV] C=1000, gamma=0.0001, kernel=rbf …
[CV] … C=1000, gamma=0.0001, kernel=rbf, score=0.683, total=   0.0s

[Parallel(n_jobs=1)]: Done 125 out of 125 | elapsed:    1.4s finished
```

[129]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.1, 1, 10, 100, 1000],
                         'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
                         'kernel': ['rbf']},
             verbose=3)
```

[130]:
```
grid.best_params_
grid.best_estimator_
grid_predictions = grid.predict(X_test)
print(confusion_matrix(y_test,grid_predictions))
print(classification_report(y_test,grid_predictions))
```

```
[[60  2]
 [26  2]]
              precision    recall  f1-score   support

           0       0.70      0.97      0.81        62
           1       0.50      0.07      0.12        28

    accuracy                           0.69        90
```

macro avg	0.60	0.52	0.47	90
weighted avg	0.64	0.69	0.60	90