Heart Failure Prediction

June 12, 2024

```
[3]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[4]: %matplotlib inline
```

```
[5]: from sklearn.linear_model import LogisticRegression
     from sklearn.ensemble import RandomForestClassifier
```

```
[86]: from sklearn.model_selection import train_test_split, cross_val_score
      from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
      from sklearn.metrics import confusion_matrix, classification_report
      from sklearn.metrics import precision_score, recall_score, f1_score
      from sklearn.metrics import RocCurveDisplay
```

```
[11]: df = pd.read_csv("heart_failure_clinical_records_dataset.csv")
      df.shape
```

```
[11]: (299, 13)
```

```
[12]: df.head()
```

```
[12]:     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  time  DEATH_EVENT
```

```
0         0     4             1
1         0     6             1
2         1     7             1
3         0     7             1
4         0     8             1
```

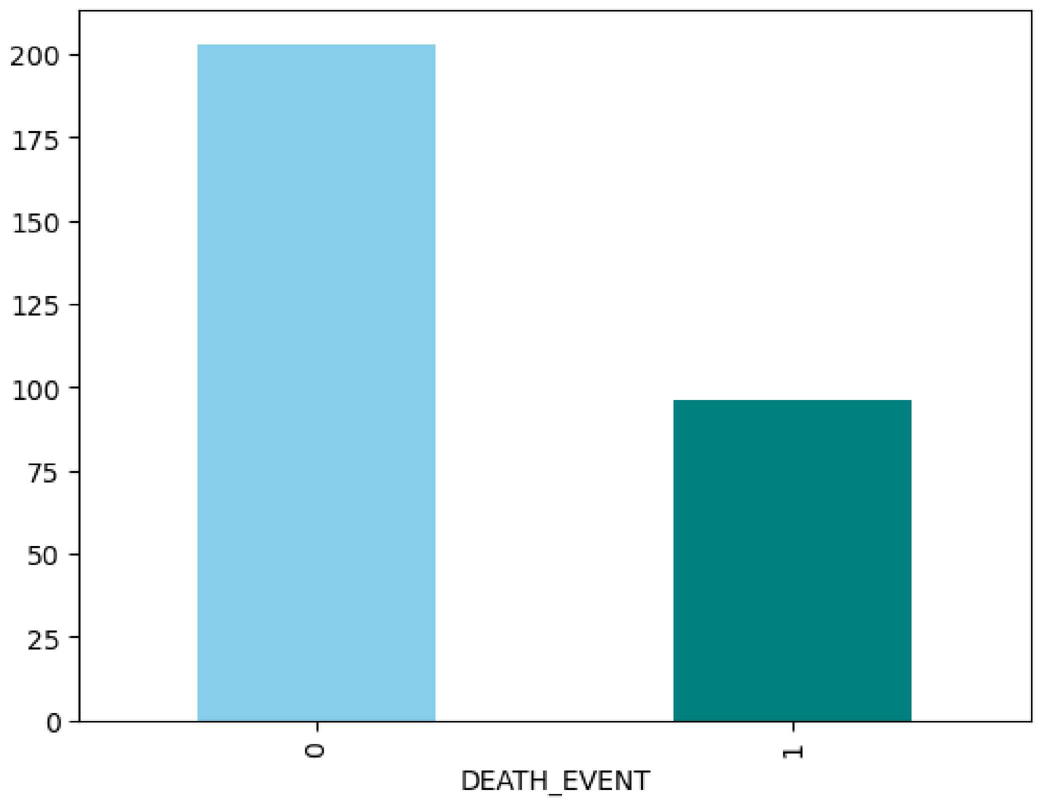
[13]:
```
df.tail()
```

[13]:
```
      age  anaemia  creatinine_phosphokinase  diabetes  ejection_fraction  \
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  \
294                    1   155000.0               1.1           143    1
295                    0   270000.0               1.2           139    0
296                    0   742000.0               0.8           138    0
297                    0   140000.0               1.4           140    1
298                    0   395000.0               1.6           136    1

     smoking  time  DEATH_EVENT
294        1   270            0
295        0   271            0
296        0   278            0
297        1   280            0
298        1   285            0
```

[14]:
```
df['DEATH_EVENT'].value_counts()
```

[14]:
```
DEATH_EVENT
0    203
1     96
Name: count, dtype: int64
```

[19]:
```
df["DEATH_EVENT"].value_counts().plot(kind="bar", color=["skyblue", "teal"]);
```



[20]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 299 entries, 0 to 298
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   age                       299 non-null    float64
 1   anaemia                   299 non-null    int64
 2   creatinine_phosphokinase  299 non-null    int64
 3   diabetes                  299 non-null    int64
 4   ejection_fraction         299 non-null    int64
 5   high_blood_pressure       299 non-null    int64
 6   platelets                 299 non-null    float64
 7   serum_creatinine          299 non-null    float64
 8   serum_sodium              299 non-null    int64
 9   sex                       299 non-null    int64
 10  smoking                   299 non-null    int64
 11  time                      299 non-null    int64
 12  DEATH_EVENT               299 non-null    int64
dtypes: float64(3), int64(10)
memory usage: 30.5 KB
```

[21]:
```
df.isna().sum()
```

[21]:
```
age                         0
anaemia                     0
creatinine_phosphokinase    0
diabetes                    0
ejection_fraction           0
high_blood_pressure         0
platelets                   0
serum_creatinine            0
serum_sodium                0
sex                         0
smoking                     0
time                        0
DEATH_EVENT                 0
dtype: int64
```

[22]:
```
df.describe()
```

[22]:
```
              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
```

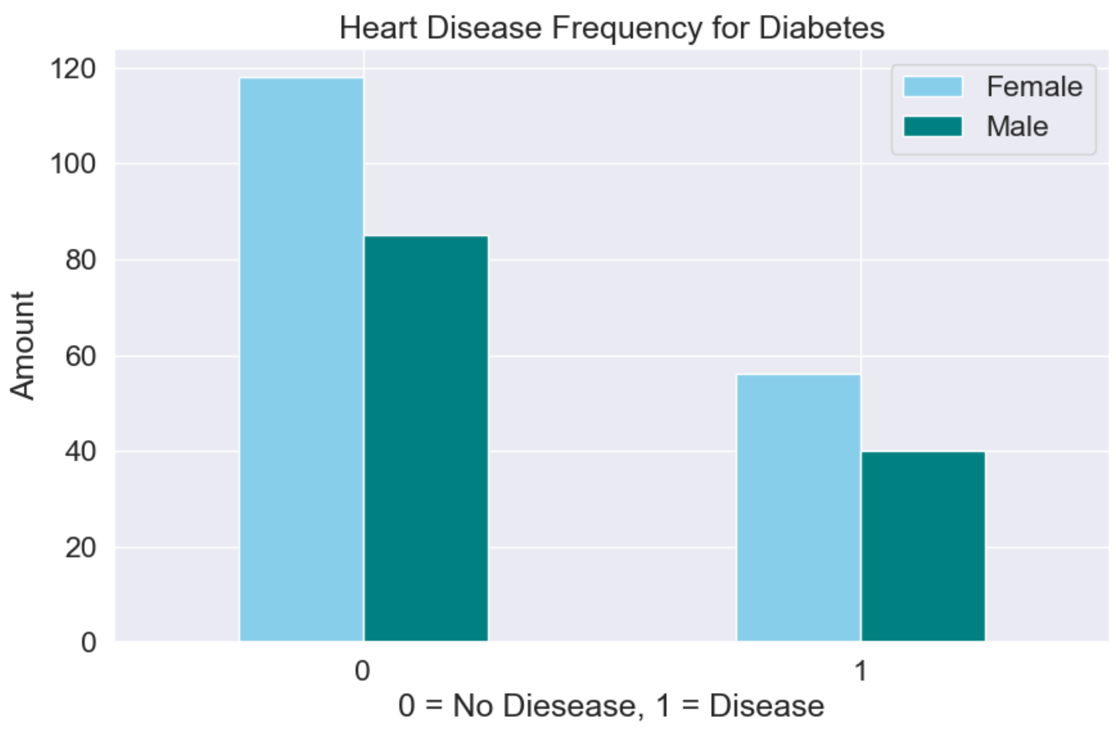
[23]:
```
df.diabetes.value_counts()
```

[23]:
```
diabetes
0    174
1    125
Name: count, dtype: int64
```

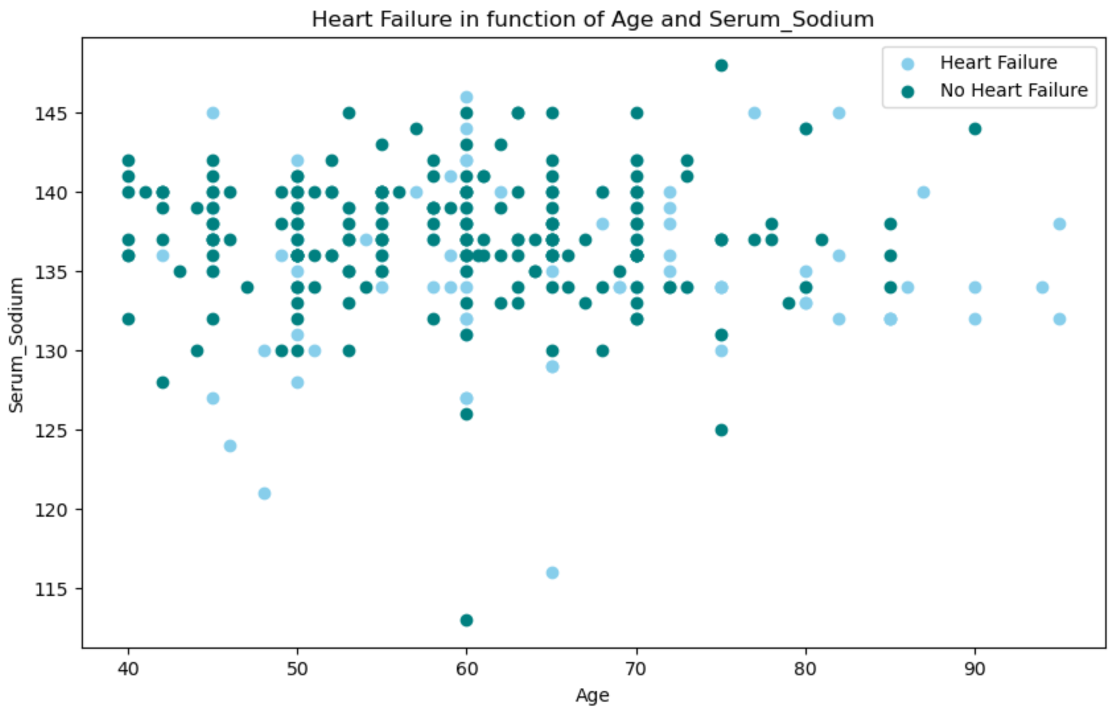
[24]:
```
pd.crosstab(df.DEATH_EVENT, df.diabetes)
```

[24]:
```
diabetes       0   1
DEATH_EVENT
0            118  85
1             56  40
```

[109]:
```
pd.crosstab(df.DEATH_EVENT, df.diabetes).plot(kind="bar",
                                    figsize=(10, 6),
                                    color=["skyblue", "teal"])

plt.title("Heart Disease Frequency for Diabetes")
plt.xlabel("0 = No Diesease, 1 = Disease")
plt.ylabel("Amount")
plt.legend(["Female", "Male"]);
plt.xticks(rotation=0);
```

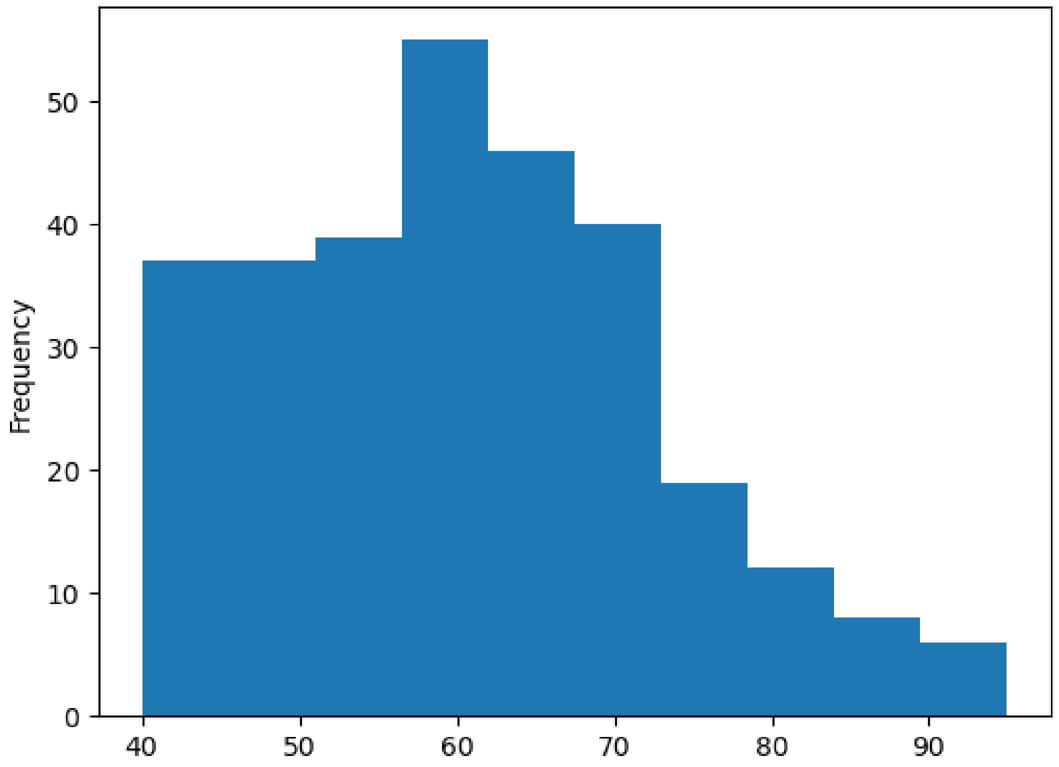


[30]:
```
plt.figure(figsize=(10, 6))

plt.scatter(df.age[df.DEATH_EVENT==1],
            df.serum_sodium[df.DEATH_EVENT==1],
            c="skyblue")

plt.scatter(df.age[df.DEATH_EVENT==0],
            df.serum_sodium[df.DEATH_EVENT==0],
            c="teal")

plt.title("Heart Failure in function of Age and Serum_Sodium")
plt.xlabel("Age")
plt.ylabel("Serum_Sodium")
plt.legend(["Heart Failure", "No Heart Failure"]);
```



```
[36]: df.age.plot.hist()
```

```
[36]: <Axes: ylabel='Frequency'>
```



[42]:
```
df.corr()
```

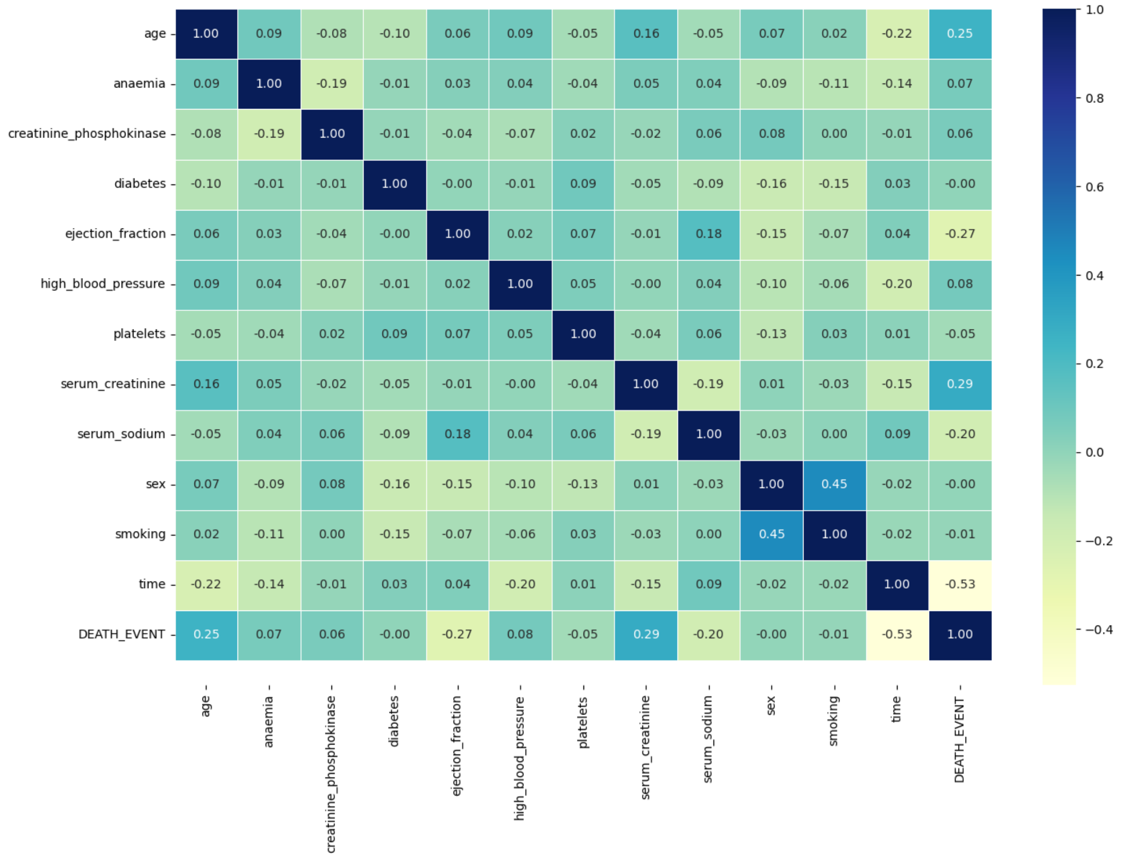
[42]:
```
                               age   anaemia  creatinine_phosphokinase  \
age                       1.000000  0.088006                 -0.081584
anaemia                   0.088006  1.000000                 -0.190741
creatinine_phosphokinase -0.081584 -0.190741                  1.000000
diabetes                 -0.101012 -0.012729                 -0.009639
ejection_fraction         0.060098  0.031557                 -0.044080
high_blood_pressure       0.093289  0.038182                 -0.070590
platelets                -0.052354 -0.043786                  0.024463
serum_creatinine          0.159187  0.052174                 -0.016408
serum_sodium             -0.045966  0.041882                  0.059550
sex                       0.065430 -0.094769                  0.079791
smoking                   0.018668 -0.107290                  0.002421
time                     -0.224068 -0.141414                 -0.009346
DEATH_EVENT               0.253729  0.066270                  0.062728

                          diabetes  ejection_fraction  high_blood_pressure  \
age                      -0.101012           0.060098             0.093289
anaemia                  -0.012729           0.031557             0.038182
creatinine_phosphokinase -0.009639          -0.044080            -0.070590
diabetes                  1.000000          -0.004850            -0.012732
```

```
ejection_fraction        -0.004850           1.000000             0.024445
high_blood_pressure      -0.012732           0.024445             1.000000
platelets                 0.092193           0.072177             0.049963
serum_creatinine         -0.046975          -0.011302            -0.004935
serum_sodium             -0.089551           0.175902             0.037109
sex                      -0.157730          -0.148386            -0.104615
smoking                  -0.147173          -0.067315            -0.055711
time                      0.033726           0.041729            -0.196439
DEATH_EVENT              -0.001943          -0.268603             0.079351

                          platelets  serum_creatinine  serum_sodium       sex  \
age                       -0.052354          0.159187     -0.045966  0.065430
anaemia                   -0.043786          0.052174      0.041882 -0.094769
creatinine_phosphokinase   0.024463         -0.016408      0.059550  0.079791
diabetes                   0.092193         -0.046975     -0.089551 -0.157730
ejection_fraction          0.072177         -0.011302      0.175902 -0.148386
high_blood_pressure        0.049963         -0.004935      0.037109 -0.104615
platelets                  1.000000         -0.041198      0.062125 -0.125120
serum_creatinine          -0.041198          1.000000     -0.189095  0.006970
serum_sodium               0.062125         -0.189095      1.000000 -0.027566
sex                       -0.125120          0.006970     -0.027566  1.000000
smoking                    0.028234         -0.027414      0.004813  0.445892
time                       0.010514         -0.149315      0.087640 -0.015608
DEATH_EVENT               -0.049139          0.294278     -0.195204 -0.004316

                           smoking      time  DEATH_EVENT
age                       0.018668 -0.224068     0.253729
anaemia                  -0.107290 -0.141414     0.066270
creatinine_phosphokinase  0.002421 -0.009346     0.062728
diabetes                 -0.147173  0.033726    -0.001943
ejection_fraction        -0.067315  0.041729    -0.268603
high_blood_pressure      -0.055711 -0.196439     0.079351
platelets                 0.028234  0.010514    -0.049139
serum_creatinine         -0.027414 -0.149315     0.294278
serum_sodium              0.004813  0.087640    -0.195204
sex                       0.445892 -0.015608    -0.004316
smoking                   1.000000 -0.022839    -0.012623
time                     -0.022839  1.000000    -0.526964
DEATH_EVENT              -0.012623 -0.526964     1.000000
```

[44]:
```
corr_matrix = df.corr()
fig, ax = plt.subplots(figsize=(15, 10))
ax = sns.heatmap(corr_matrix,
                 annot=True,
                 linewidths=0.5,
                 fmt=".2f",
                 cmap="YlGnBu");
```

```
bottom, top = ax.get_ylim()
ax.set_ylim(bottom + 0.5, top)
```

[44]:
```
(13.5, 0.0)
```



[56]:
```
X = df.drop("DEATH_EVENT", axis=1)

y = df["DEATH_EVENT"]
```

[57]:
```
X
```

[57]:
```
      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
0          0     4
1          0     6
2          1     7
3          0     7
4          0     8
..       ...   ...
294        1   270
295        0   271
296        0   278
297        1   280
298        1   285

[299 rows x 12 columns]
```

```
[48]: y
```

```
[48]: 0      1
      1      1
      2      1
      3      1
      4      1
            ..
      294    0
      295    0
      296    0
      297    0
      298    0
      Name: DEATH_EVENT, Length: 299, dtype: int64
```

[54]:
```
np.random.seed(42)

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

[55]:
```
X_train
```

[55]:
```
        age  anaemia  creatinine_phosphokinase  diabetes  ejection_fraction  \
6    75.000        1                       246         0                 15
183  75.000        0                        99         0                 38
185  60.667        1                       104         1                 30
146  52.000        0                       132         0                 30
30   94.000        0                       582         1                 38
..      ...      ...                       ...       ...                ...
188  60.667        1                       151         1                 40
71   58.000        0                       582         1                 35
106  55.000        0                       748         0                 45
270  44.000        0                       582         1                 30
102  80.000        0                       898         0                 25

     high_blood_pressure  platelets  serum_creatinine  serum_sodium  sex  \
6                      0  127000.00              1.20           137    1
183                    1  224000.00              2.50           134    1
185                    0  389000.00              1.50           136    1
146                    0  218000.00              0.70           136    1
30                     1  263358.03              1.83           134    1
..                   ...        ...               ...           ...  ...
188                    1  201000.00              1.00           136    0
71                     0  122000.00              0.90           139    1
106                    0  263000.00              1.30           137    1
270                    1  263358.03              1.60           130    1
102                    0  149000.00              1.10           144    1

     smoking  time
6          0    10
183        0   162
185        0   171
146        1   112
30         0    27
..       ...   ...
188        0   172
71         1    71
106        0    88
270        1   244
102        1    87
```

```
[239 rows x 12 columns]
```

[58]:
```
y_train
```

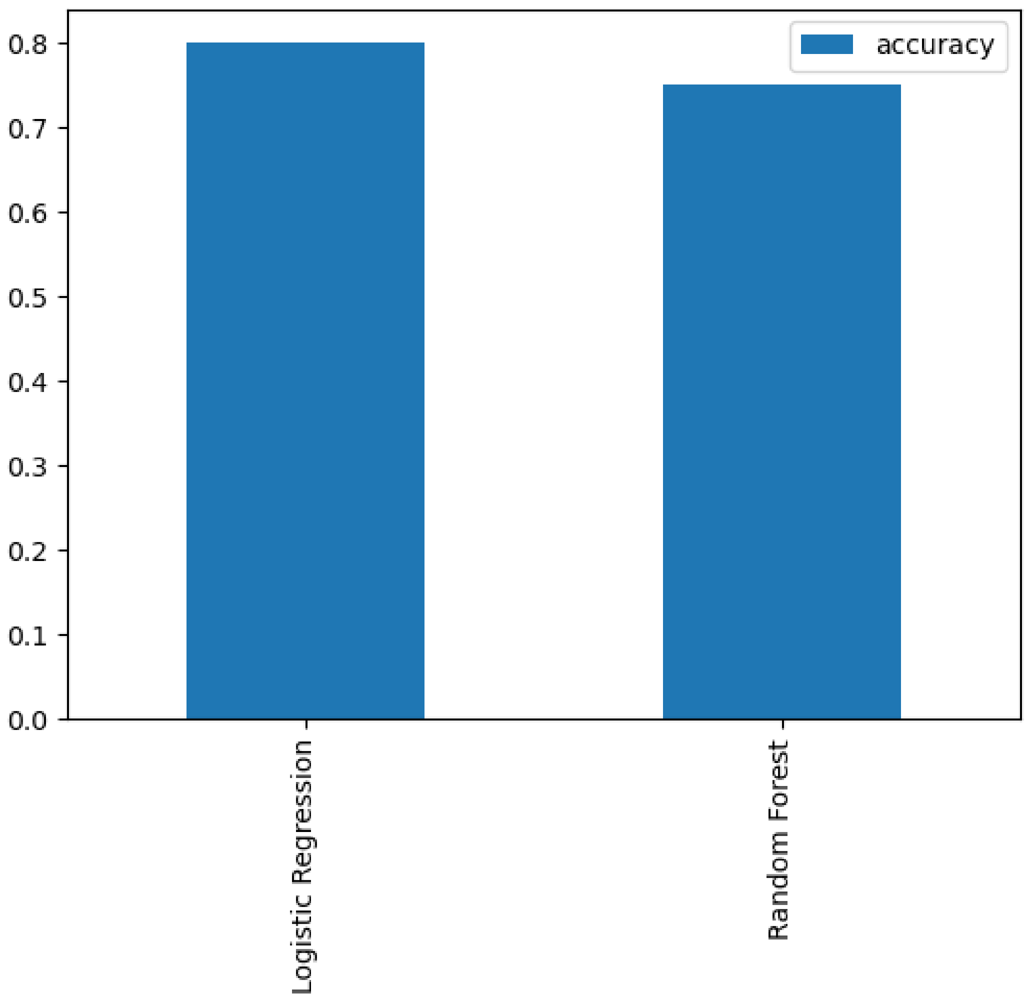
[58]:
```
6      1
183    1
185    1
146    0
30     1
      ..
188    0
71     0
106    0
270    0
102    0
Name: DEATH_EVENT, Length: 239, dtype: int64
```

[65]:
```
models = {"Logistic Regression": LogisticRegression(),
          "Random Forest": RandomForestClassifier()}

def fit_and_score(models, X_train, X_test, y_train, y_test):

    np.random.seed(42)

    model_scores = {}

    for name, model in models.items():
        model.fit(X_train, y_train)
        model_scores[name] = model.score(X_test, y_test)
    return model_scores
```

[66]:
```
model_scores = fit_and_score(models=models,
                             X_train=X_train,
                             X_test=X_test,
                             y_train=y_train,
                             y_test=y_test)


model_scores
```

[66]:
```
{'Logistic Regression': 0.8, 'Random Forest': 0.75}
```

[67]:
```
model_compare = pd.DataFrame(model_scores, index=["accuracy"])
model_compare.T.plot.bar();
```



[68]:
```
log_reg_grid = {"C": np.logspace(-4, 4, 20),
                "solver": ["liblinear"]}

rf_grid = {"n_estimators": np.arange(10, 1000, 50),
           "max_depth": [None, 3, 5, 10],
           "min_samples_split": np.arange(2, 20, 2),
           "min_samples_leaf": np.arange(1, 20, 2)}
```

[69]:
```
np.random.seed(42)

rs_log_reg = RandomizedSearchCV(LogisticRegression(),
                                param_distributions=log_reg_grid,
                                cv=5,
                                n_iter=20,
                                verbose=True)
```

```
rs_log_reg.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits
```

[69]:
```
RandomizedSearchCV(cv=5, estimator=LogisticRegression(), n_iter=20,
                   param_distributions={'C': array([1.00000000e-04,
2.63665090e-04, 6.95192796e-04, 1.83298071e-03,
       4.83293024e-03, 1.27427499e-02, 3.35981829e-02, 8.85866790e-02,
       2.33572147e-01, 6.15848211e-01, 1.62377674e+00, 4.28133240e+00,
       1.12883789e+01, 2.97635144e+01, 7.84759970e+01, 2.06913808e+02,
       5.45559478e+02, 1.43844989e+03, 3.79269019e+03, 1.00000000e+04]),
                                        'solver': ['liblinear']},
                   verbose=True)
```

[70]:
```
rs_log_reg.best_params_
```

[70]:
```
{'solver': 'liblinear', 'C': 0.08858667904100823}
```

[71]:
```
rs_log_reg.score(X_test, y_test)
```

[71]:
```
0.75
```

[72]:
```
np.random.seed(42)

rs_rf = RandomizedSearchCV(RandomForestClassifier(),
                           param_distributions=rf_grid,
                           cv=5,
                           n_iter=20,
                           verbose=True)

rs_rf.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits
```

[72]:
```
RandomizedSearchCV(cv=5, estimator=RandomForestClassifier(), n_iter=20,
                   param_distributions={'max_depth': [None, 3, 5, 10],
                                        'min_samples_leaf': array([ 1,  3,  5,
7,  9, 11, 13, 15, 17, 19]),
                                        'min_samples_split': array([ 2,  4,  6,
8, 10, 12, 14, 16, 18]),
                                        'n_estimators': array([ 10,  60, 110,
160, 210, 260, 310, 360, 410, 460, 510, 560, 610,
       660, 710, 760, 810, 860, 910, 960])},
                   verbose=True)
```

[74]:
```
rs_rf.best_params_
```

[74]:
```
{'n_estimators': 260,
 'min_samples_split': 16,
 'min_samples_leaf': 9,
 'max_depth': 10}
```

[75]:
```
rs_rf.score(X_test, y_test)
```

[75]:
```
0.75
```

[77]:
```
log_reg_grid = {"C": np.logspace(-4, 4, 30),
                "solver": ["liblinear"]}

gs_log_reg = GridSearchCV(LogisticRegression(),
                          param_grid=log_reg_grid,
                          cv=5,
                          verbose=True)

gs_log_reg.fit(X_train, y_train);
```

```
Fitting 5 folds for each of 30 candidates, totalling 150 fits
```

[79]:
```
gs_log_reg.best_params_
```

[79]:
```
{'C': 0.008531678524172805, 'solver': 'liblinear'}
```

[80]:
```
gs_log_reg.score(X_test, y_test)
```

[80]:
```
0.75
```

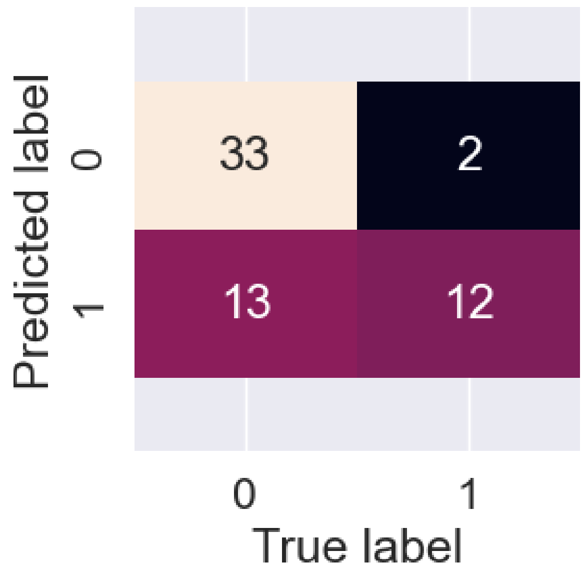
[82]:
```
y_preds = gs_log_reg.predict(X_test)
y_preds
```

[82]:
```
array([0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1], dtype=int64)
```

[96]:
```
print(confusion_matrix(y_test, y_preds))
```

```
[[33  2]
 [13 12]]
```

[97]:
```
sns.set(font_scale=1.5)

def plot_conf_mat(y_test, y_preds):

    fig, ax = plt.subplots(figsize=(3, 3))
    ax = sns.heatmap(confusion_matrix(y_test, y_preds),
                     annot=True,
                     cbar=False)
    plt.xlabel("True label")
    plt.ylabel("Predicted label")

    bottom, top = ax.get_ylim()
    ax.set_ylim(bottom + 0.5, top - 0.5)

plot_conf_mat(y_test, y_preds)
```



[98]:
```
print(classification_report(y_test, y_preds))
```

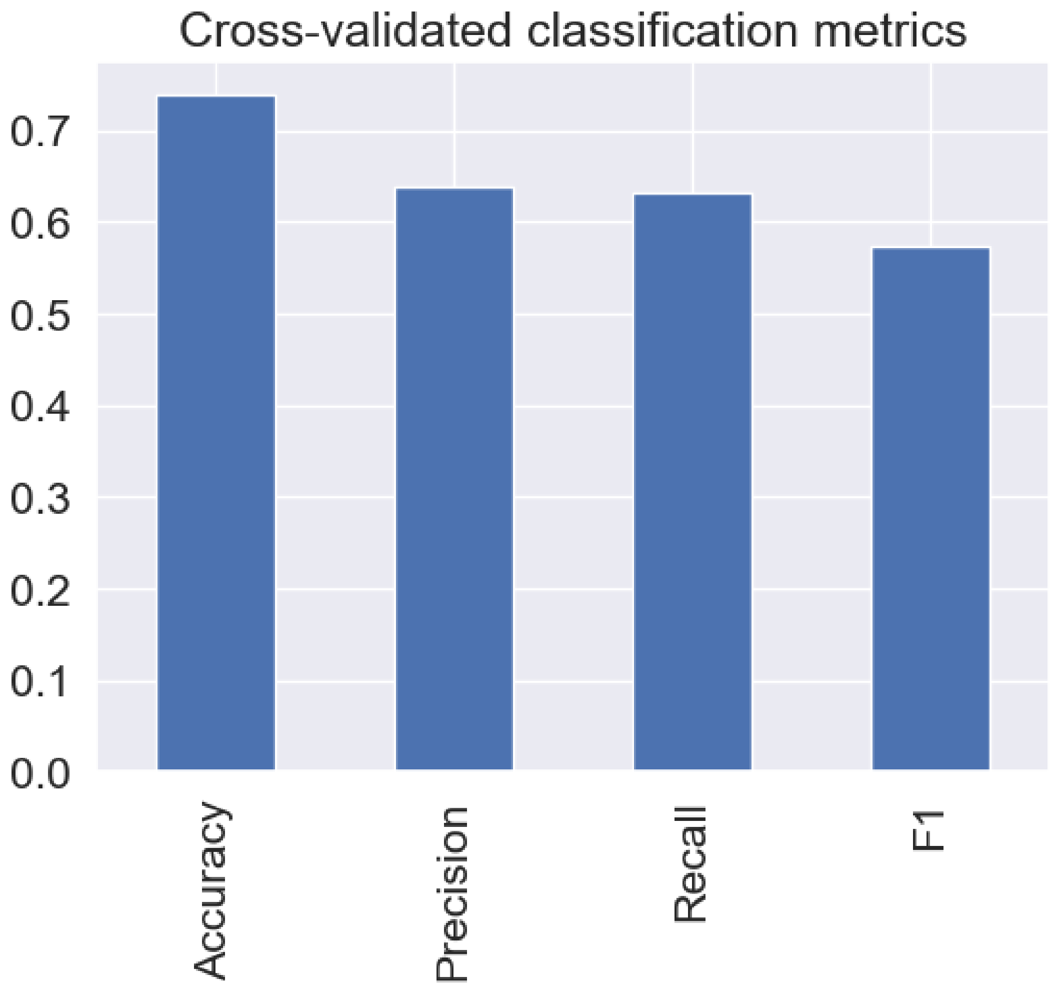
```
              precision    recall  f1-score   support

           0       0.72      0.94      0.81        35
           1       0.86      0.48      0.62        25

    accuracy                           0.75        60
   macro avg       0.79      0.71      0.72        60
weighted avg       0.78      0.75      0.73        60
```

[100]:
```
clf = LogisticRegression(C=0.008531678524172805,
                         solver="liblinear")
```

[110]:
```
cv_acc = cross_val_score(clf,
                         X,
                         y,
                         cv=5,
                         scoring="accuracy")
```

```
cv_acc = np.mean(cv_acc)
cv_acc

cv_precision = cross_val_score(clf,
                         X,
                         y,
                         cv=5,
                         scoring="precision")
cv_precision=np.mean(cv_precision)
cv_precision

cv_recall = cross_val_score(clf,
                         X,
                         y,
                         cv=5,
                         scoring="recall")
cv_recall = np.mean(cv_recall)
cv_recall

cv_f1 = cross_val_score(clf,
                         X,
                         y,
                         cv=5,
                         scoring="f1")
cv_f1 = np.mean(cv_f1)
cv_f1

cv_metrics = pd.DataFrame({"Accuracy": cv_acc,
                           "Precision": cv_precision,
                           "Recall": cv_recall,
                           "F1": cv_f1},
                          index=[0])

cv_metrics.T.plot.bar(title="Cross-validated classification metrics",
                      legend=False);
```

```
C:\Users\vidus\anaconda3.1\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 due to no predicted samples. Use
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```



```
[111]: clf = LogisticRegression(C=0.008531678524172805,
                                solver="liblinear")

       clf.fit(X_train, y_train);
```

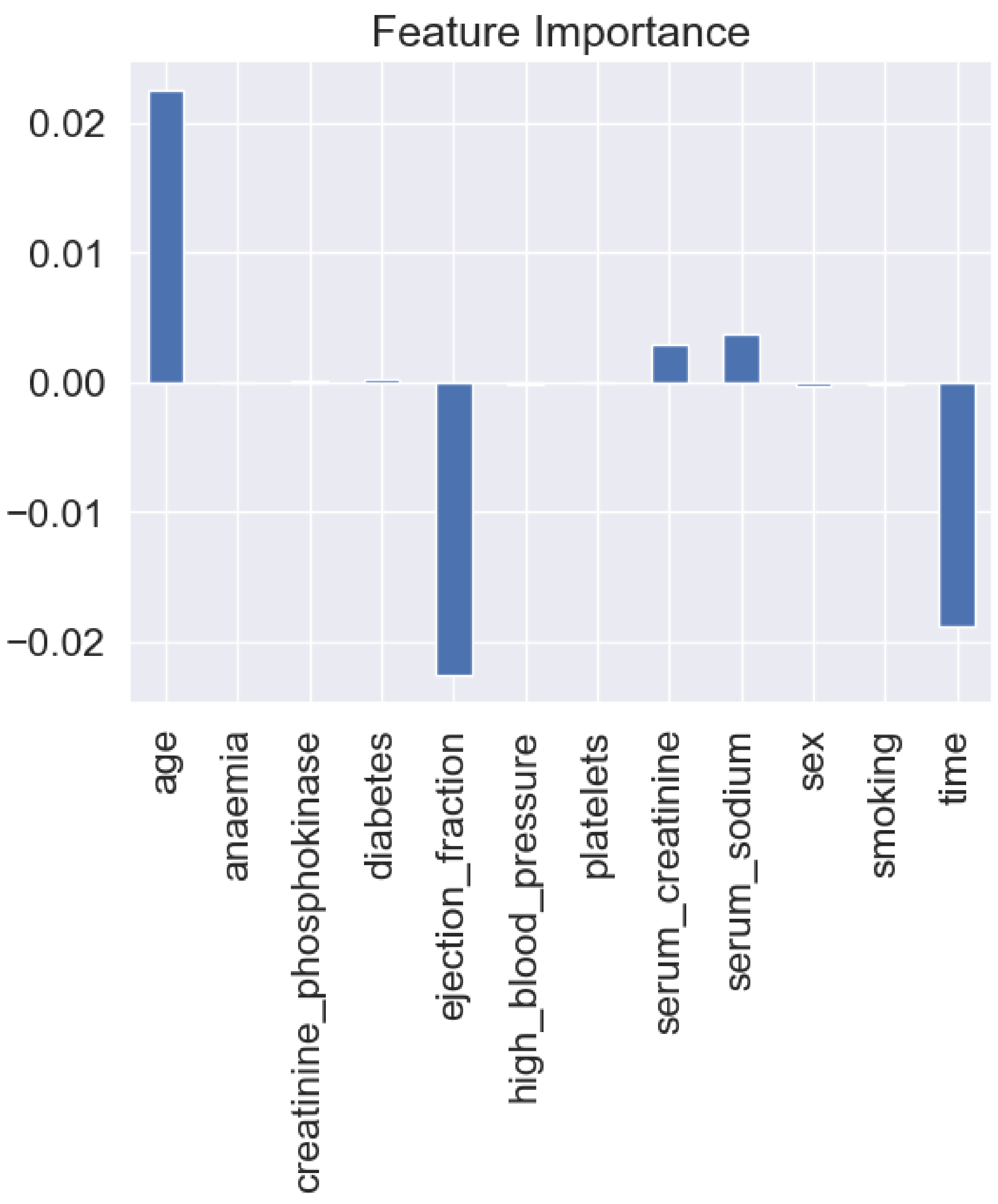
```
[106]: clf.coef_
```

```
[106]: array([[ 2.25664095e-02, -1.32373957e-05,  1.11736658e-04,
                 2.12224031e-04, -2.25512363e-02, -1.64240711e-04,
                 1.44950743e-07,  2.93717113e-03,  3.74278962e-03,
                -2.93417830e-04, -1.29592144e-04, -1.87936576e-02]])
```

```
[107]: feature_dict = dict(zip(df.columns, list(clf.coef_[0])))
       feature_dict
```

```
[107]: {'age': 0.02256640949058526,
        'anaemia': -1.3237395694908342e-05,
        'creatinine_phosphokinase': 0.0001117366583958865,
        'diabetes': 0.00021222403098543558,
        'ejection_fraction': -0.0225512363385351,
        'high_blood_pressure': -0.00016424071054663595,
        'platelets': 1.4495074316257639e-07,
        'serum_creatinine': 0.0029371711289282337,
        'serum_sodium': 0.0037427896151338,
        'sex': -0.00029341783017448317,
        'smoking': -0.0001295921442812619,
        'time': -0.018793657590705382}
```

```
[108]: feature_df = pd.DataFrame(feature_dict, index=[0])
       feature_df.T.plot.bar(title="Feature Importance", legend=False);
```


Feature Importance
0.02
0.01
0.00
−0.01
−0.02
age
anaemia
creatinine_phosphokinase
diabetes
ejection_fraction
high_blood_pressure
platelets
serum_creatinine
serum_sodium
sex
smoking
time

```
[ ]:
```