In [57]:
```
import joblib
from xgboost import XGBClassifier
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, VotingC
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_validate, GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from lightgbm import LGBMClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,f1_score, classification_report, roc_auc_scor
df=sns.load_dataset('titanic')
pd.set_option('display.max_columns',None)
pd.set_option('display.width',500)
```

In [2]:
```
pip install xgboost
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: xgboost in c:\users\user\appdata\roaming\python\python31
1\site-packages (2.0.3)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (fro
m xgboost) (1.24.3)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-packages (fro
m xgboost) (1.11.1)
Note: you may need to restart the kernel to use updated packages.
```

In [3]:
```
pip install lightgbm
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: lightgbm in c:\users\user\appdata\roaming\python\python3
11\site-packages (4.3.0)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (fro
m lightgbm) (1.24.3)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-packages (fro
m lightgbm) (1.11.1)
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
df.head()
```

Out[4]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southa
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Che
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southa
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southa
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southa

In [5]:
```
p=['pclass','age','sibsp','fare']
X = df[p]
y = df['survived']
```

In [6]:
```
X
```

Out[6]:

	pclass	age	sibsp	fare
0	3	22.0	1	7.2500
1	1	38.0	1	71.2833
2	3	26.0	0	7.9250
3	1	35.0	1	53.1000
4	3	35.0	0	8.0500
...	...	...	...	...
886	2	27.0	0	13.0000
887	1	19.0	0	30.0000
888	3	NaN	1	23.4500
889	1	26.0	0	30.0000
890	3	32.0	0	7.7500

891 rows × 4 columns

In [7]:
```
y
```

Out[7]:
```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: survived, Length: 891, dtype: int64
```

In [8]:
```
y.isnull().values.any()
```

Out[8]:
```
False
```

In [9]:
```
updated_df = X
updated_df['age']=updated_df['age'].fillna(updated_df['age'].mean())
updated_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   pclass  891 non-null    int64  
 1   age     891 non-null    float64
 2   sibsp   891 non-null    int64  
 3   fare    891 non-null    float64
dtypes: float64(2), int64(2)
memory usage: 28.0 KB

C:\Users\User\AppData\Local\Temp\ipykernel_9872\925103184.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  updated_df['age']=updated_df['age'].fillna(updated_df['age'].mean())
```

In [ ]:
```
#5.Stacking & Ensemble Learning
```

In [34]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
```

In [52]:
```
log_model = LogisticRegression().fit(X_train, y_train)
```

In [55]:
```
y_pred = log_model.predict(X_test)
```

In [59]:
```
y_prob = log_model.predict_proba(X_train)[:,1]
```

In [ ]:
```

```

In [58]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.72      0.90      0.80       105
           1       0.77      0.50      0.61        74

    accuracy                           0.73       179
   macro avg       0.74      0.70      0.70       179
weighted avg       0.74      0.73      0.72       179
```

In [60]:
```
log_model = LogisticRegression().fit(X, y)
```

In [61]:
```
cv_results = cross_validate(log_model, X,y, cv=5, scoring=["accuracy","f1","roc_auc"])
```

In [62]:
```
cv_results
```

Out[62]:
```
{'fit_time': array([0.02063274, 0.00897288, 0.0087173 , 0.00951314, 0.01012135]),
 'score_time': array([0.00958538, 0.00602651, 0.00511909, 0.01014042, 0.0050025 ]),
 'test_accuracy': array([0.63128492, 0.68539326, 0.73033708, 0.73595506, 0.68539326]),
 'test_f1': array([0.4       , 0.5483871 , 0.54716981, 0.56074766, 0.51724138]),
 'test_roc_auc': array([0.62312253, 0.6993984 , 0.73415775, 0.79278075, 0.77961707])}
```

In [63]:
```
cv_results['test_accuracy'].mean()
```

Out[63]:
```
0.6936727135773021
```

In [64]:
```
cv_results['test_f1'].mean()
```

Out[64]:
```
0.514709190191339
```

In [65]:
```
cv_results['test_roc_auc'].mean()
```

Out[65]:
```
0.7258153000475673
```

In [ ]:
```

```