oolr8ormg

April 25, 2025

[2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

tt = sns.load_dataset("titanic")
tt.head(10)
```

[2]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True	NaN	Queenstown	no	True
6	0	1	male	54.0	0	0	51.8625	S	First	man	True	E	Southampton	no	True
7	0	3	male	2.0	3	1	21.0750	S	Third	child	False	NaN	Southampton	no	False
8	1	3	female	27.0	0	2	11.1333	S	Third	woman	False	NaN	Southampton	yes	False
9	1	2	female	14.0	1	0	30.0708	C	Second	child	False	NaN	Cherbourg	yes	False

[3]:
```
del tt["alive"]
del tt["class"]

tt
```

[3]:
```
     survived  pclass     sex   age  sibsp  parch     fare embarked    who  \
0           0       3    male  22.0      1      0   7.2500        S    man
1           1       1  female  38.0      1      0  71.2833        C  woman
2           1       3  female  26.0      0      0   7.9250        S  woman
3           1       1  female  35.0      1      0  53.1000        S  woman
4           0       3    male  35.0      0      0   8.0500        S    man
..        ...     ...     ...   ...    ...    ...      ...      ...    ...
886         0       2    male  27.0      0      0  13.0000        S    man
887         1       1  female  19.0      0      0  30.0000        S  woman
888         0       3  female   NaN      1      2  23.4500        S  woman
889         1       1    male  26.0      0      0  30.0000        C    man
890         0       3    male  32.0      0      0   7.7500        Q    man

     adult_male deck  embark_town  alone
0          True  NaN  Southampton  False
1         False    C    Cherbourg  False
2         False  NaN  Southampton   True
3         False    C  Southampton  False
4          True  NaN  Southampton   True
..          ...  ...          ...    ...
886        True  NaN  Southampton   True
887       False    B  Southampton   True
888       False  NaN  Southampton  False
889        True    C    Cherbourg   True
890        True  NaN   Queenstown   True

[891 rows x 13 columns]
```

[4]:
```
tt['pclass'].value_counts()
```

[4]:
```
pclass
3    491
1    216
2    184
Name: count, dtype: int64
```

[5]:
```
tt.isnull().sum()
```

[5]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
who              0
```

```
adult_male       0
deck           688
embark_town      2
alone            0
dtype: int64
```

[6]:
```
# filling null values in age
tt['age'] = tt['age'].fillna(tt['age'].mode()[0])
# filling null values in embarked
tt['embarked'] = tt['embarked'].fillna(tt['embarked'].mode()[0])
#
tt['embark_town'] = tt['embark_town'].fillna(tt['embark_town'].mode()[0])
#
tt['deck'] = tt['deck'].fillna(tt['deck'].mode()[0])

tt
```

[6]:
```
     survived  pclass     sex   age  sibsp  parch     fare embarked    who  \
0           0       3    male  22.0      1      0   7.2500        S    man
1           1       1  female  38.0      1      0  71.2833        C  woman
2           1       3  female  26.0      0      0   7.9250        S  woman
3           1       1  female  35.0      1      0  53.1000        S  woman
4           0       3    male  35.0      0      0   8.0500        S    man
..        ...     ...     ...   ...    ...    ...      ...      ...    ...
886         0       2    male  27.0      0      0  13.0000        S    man
887         1       1  female  19.0      0      0  30.0000        S  woman
888         0       3  female  24.0      1      2  23.4500        S  woman
889         1       1    male  26.0      0      0  30.0000        C    man
890         0       3    male  32.0      0      0   7.7500        Q    man

     adult_male deck  embark_town  alone
0          True    C  Southampton  False
1         False    C    Cherbourg  False
2         False    C  Southampton   True
3         False    C  Southampton  False
4          True    C  Southampton   True
..          ...  ...          ...    ...
886        True    C  Southampton   True
887       False    B  Southampton   True
888       False    C  Southampton  False
889        True    C    Cherbourg   True
890        True    C   Queenstown   True

[891 rows x 13 columns]
```

[7]:
```
tt.isnull().sum()
```

[7]:
```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
who            0
adult_male     0
deck           0
embark_town    0
alone          0
dtype: int64
```

[8]:
```
tt['family_size'] = tt['sibsp'] + tt['parch'] + 1

del tt['sibsp']
del tt['parch']

tt
```

[8]:

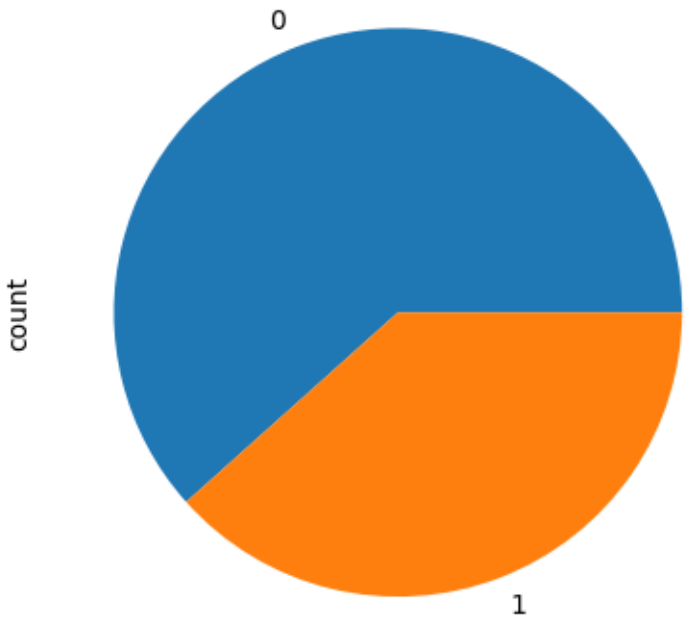
	survived	pclass	sex	age	fare	embarked	who	adult_male	deck	\
0	0	3	male	22.0	7.2500	S	man	True	C	
1	1	1	female	38.0	71.2833	C	woman	False	C	
2	1	3	female	26.0	7.9250	S	woman	False	C	
3	1	1	female	35.0	53.1000	S	woman	False	C	
4	0	3	male	35.0	8.0500	S	man	True	C	
..	…	…	…	…	…	…	…	…	…	
886	0	2	male	27.0	13.0000	S	man	True	C	
887	1	1	female	19.0	30.0000	S	woman	False	B	
888	0	3	female	24.0	23.4500	S	woman	False	C	
889	1	1	male	26.0	30.0000	C	man	True	C	
890	0	3	male	32.0	7.7500	Q	man	True	C	

	embark_town	alone	family_size
0	Southampton	False	2
1	Cherbourg	False	2
2	Southampton	True	1
3	Southampton	False	2
4	Southampton	True	1
..	…	…	…
886	Southampton	True	1
887	Southampton	True	1
888	Southampton	False	4
889	Cherbourg	True	1
890	Queenstown	True	1

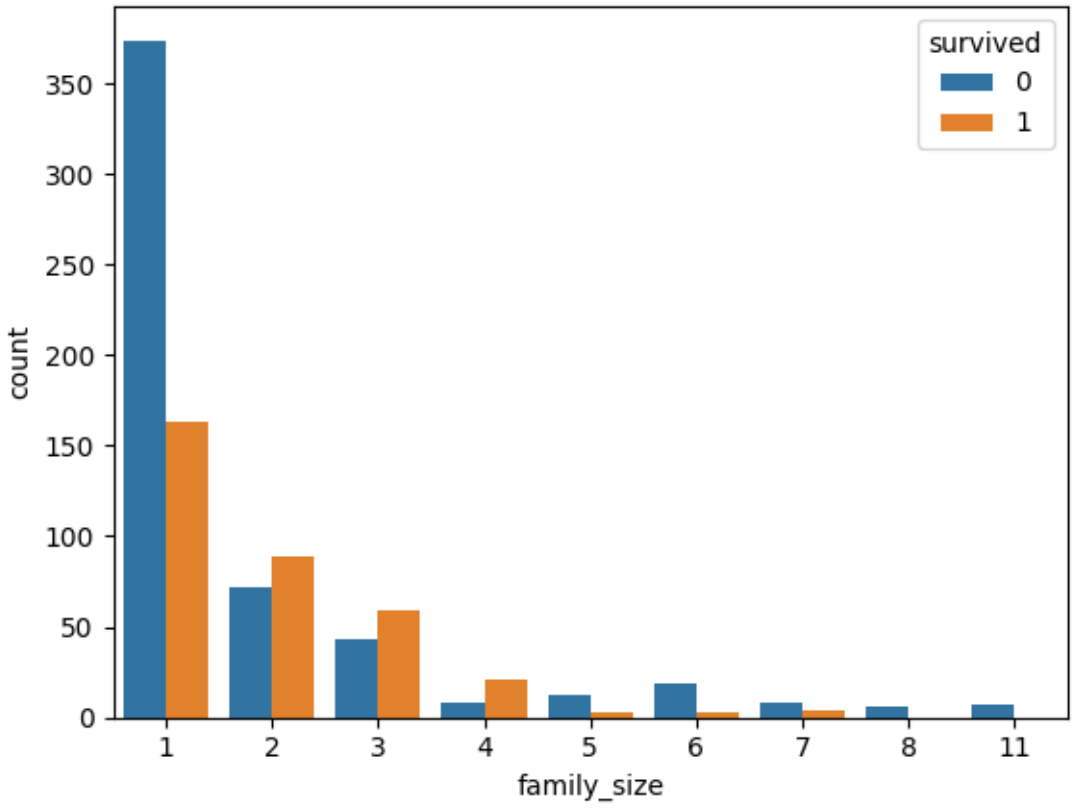
```
[891 rows x 12 columns]
```

[9]:
```
tt['survived'].value_counts().plot(kind='pie')
```

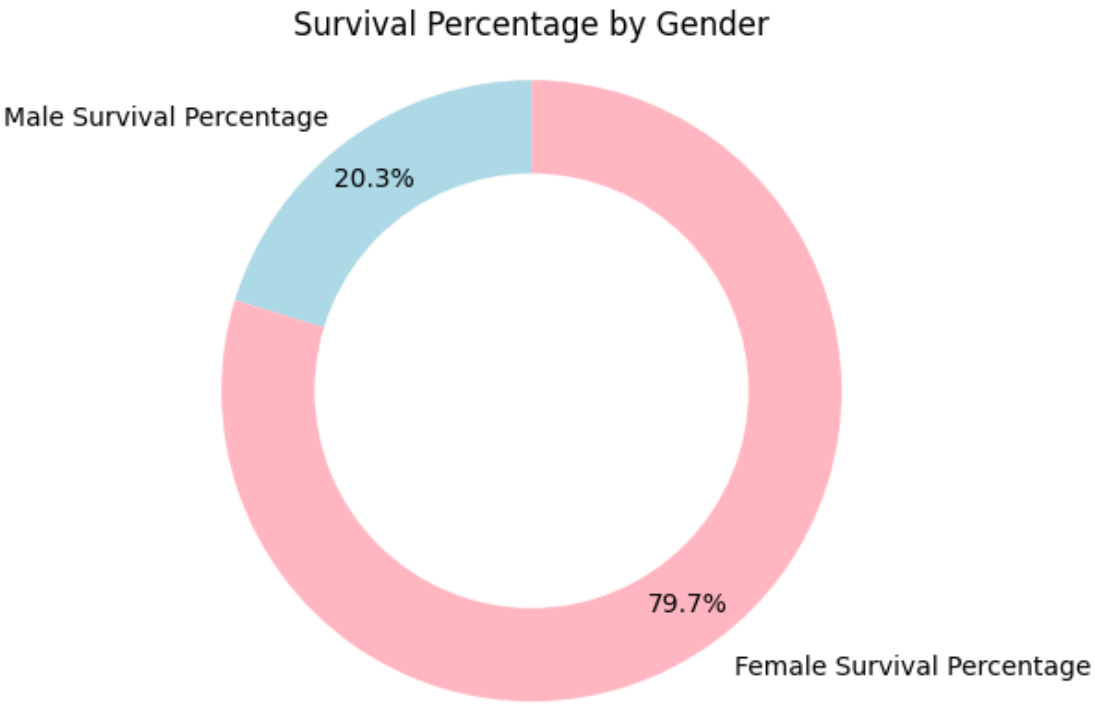
[9]:
```
<Axes: ylabel='count'>
```



[10]:
```
sns.countplot(x ='family_size', hue='survived', data= tt)
plt.show()
```



[11]:
```
male_survival_percentage = (tt[(tt['sex'] == 'male') & (tt['survived'] == 1)].
 ↪shape[0] / len(tt[tt['sex'] == 'male'])) * 100
female_survival_percentage = (tt[(tt['sex'] == 'female') & (tt['survived'] ==␣
 ↪1)].shape[0] / len(tt[tt['sex'] == 'female'])) * 100

labels = ['Male Survival Percentage', 'Female Survival Percentage']
sizes = [male_survival_percentage, female_survival_percentage]
colors = ['lightblue', 'lightpink']

plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90,␣
 ↪pctdistance=0.85)

centre_circle = plt.Circle((0, 0), 0.70, fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

plt.axis('equal')
plt.title('Survival Percentage by Gender')
plt.show()
```



[12]:
```
enco_sex = pd.get_dummies(tt['sex'], prefix= 'sex')
tt = pd.concat([tt, enco_sex],axis=1)
tt.drop('sex', axis=1, inplace=True)

tt
```

[12]:

	survived	pclass	age	fare	embarked	who	adult_male	deck	\
0	0	3	22.0	7.2500	S	man	True	C	
1	1	1	38.0	71.2833	C	woman	False	C	
2	1	3	26.0	7.9250	S	woman	False	C	
3	1	1	35.0	53.1000	S	woman	False	C	
4	0	3	35.0	8.0500	S	man	True	C	
..	…	…	…	…	…	…	…	…	
886	0	2	27.0	13.0000	S	man	True	C	
887	1	1	19.0	30.0000	S	woman	False	B	
888	0	3	24.0	23.4500	S	woman	False	C	
889	1	1	26.0	30.0000	C	man	True	C	
890	0	3	32.0	7.7500	Q	man	True	C	

	embark_town	alone	family_size	sex_female	sex_male
0	Southampton	False	2	False	True
1	Cherbourg	False	2	True	False
2	Southampton	True	1	True	False

```
3    Southampton  False            2        True     False
4    Southampton   True            1       False      True
..           …    …          …         …       …
886  Southampton   True            1       False      True
887  Southampton   True            1        True     False
888  Southampton  False            4        True     False
889    Cherbourg   True            1       False      True
890   Queenstown   True            1       False      True

[891 rows x 13 columns]
```

[13]:
```
enco_embark = pd.get_dummies(tt['embarked'], prefix= 'embarked')
tt = pd.concat([tt, enco_embark],axis=1)
tt.drop('embarked', axis=1, inplace=True)

tt
```

[13]:
```
     survived  pclass   age     fare    who  adult_male deck  embark_town  \
0           0       3  22.0   7.2500    man        True    C  Southampton
1           1       1  38.0  71.2833  woman       False    C    Cherbourg
2           1       3  26.0   7.9250  woman       False    C  Southampton
3           1       1  35.0  53.1000  woman       False    C  Southampton
4           0       3  35.0   8.0500    man        True    C  Southampton
..        …     …   …      …    …          …  …          …
886         0       2  27.0  13.0000    man        True    C  Southampton
887         1       1  19.0  30.0000  woman       False    B  Southampton
888         0       3  24.0  23.4500  woman       False    C  Southampton
889         1       1  26.0  30.0000    man        True    C    Cherbourg
890         0       3  32.0   7.7500    man        True    C   Queenstown

     alone  family_size  sex_female  sex_male  embarked_C  embarked_Q  \
0    False            2       False      True       False       False
1    False            2        True     False        True       False
2     True            1        True     False       False       False
3    False            2        True     False       False       False
4     True            1       False      True       False       False
..     …           …         …         …          …          …
886   True            1       False      True       False       False
887   True            1        True     False       False       False
888  False            4        True     False       False       False
889   True            1       False      True        True       False
890   True            1       False      True       False        True

     embarked_S
0          True
1         False
2          True
```

```
3          True
4          True
..          …
886        True
887        True
888        True
889       False
890       False

[891 rows x 15 columns]
```

[14]:
```
enco_who = pd.get_dummies(tt['who'], prefix= 'who')
tt = pd.concat([tt, enco_who],axis=1)
tt.drop('who', axis=1, inplace=True)

tt
```

[14]:
```
     survived  pclass   age     fare  adult_male deck  embark_town  alone  \
0           0       3  22.0   7.2500        True    C  Southampton  False
1           1       1  38.0  71.2833       False    C    Cherbourg  False
2           1       3  26.0   7.9250       False    C  Southampton   True
3           1       1  35.0  53.1000       False    C  Southampton  False
4           0       3  35.0   8.0500        True    C  Southampton   True
..        …     …   …      …          …  …           …    …
886         0       2  27.0  13.0000        True    C  Southampton   True
887         1       1  19.0  30.0000       False    B  Southampton   True
888         0       3  24.0  23.4500       False    C  Southampton  False
889         1       1  26.0  30.0000        True    C    Cherbourg   True
890         0       3  32.0   7.7500        True    C   Queenstown   True

     family_size  sex_female  sex_male  embarked_C  embarked_Q  embarked_S  \
0              2       False      True       False       False        True
1              2        True     False        True       False       False
2              1        True     False       False       False        True
3              2        True     False       False       False        True
4              1       False      True       False       False        True
..           …         …       …          …          …         …
886            1       False      True       False       False        True
887            1        True     False       False       False        True
888            4        True     False       False       False        True
889            1       False      True        True       False       False
890            1       False      True       False        True       False

     who_child  who_man  who_woman
0        False     True      False
1        False    False       True
2        False    False       True
```

```
3        False    False       True
4        False     True      False
..         …       …         …
886      False     True      False
887      False    False       True
888      False    False       True
889      False     True      False
890      False     True      False

[891 rows x 17 columns]
```

[15]:
```
enco_deck = pd.get_dummies(tt['deck'], prefix= 'deck')
tt = pd.concat([tt, enco_deck],axis=1)
tt.drop('deck', axis=1, inplace=True)

tt
```

[15]:
```
     survived  pclass   age     fare  adult_male  embark_town  alone  \
0           0       3  22.0   7.2500        True  Southampton  False
1           1       1  38.0  71.2833       False    Cherbourg  False
2           1       3  26.0   7.9250       False  Southampton   True
3           1       1  35.0  53.1000       False  Southampton  False
4           0       3  35.0   8.0500        True  Southampton   True
..        …     …   …      …          …           …    …
886         0       2  27.0  13.0000        True  Southampton   True
887         1       1  19.0  30.0000       False  Southampton   True
888         0       3  24.0  23.4500       False  Southampton  False
889         1       1  26.0  30.0000        True    Cherbourg   True
890         0       3  32.0   7.7500        True   Queenstown   True

     family_size  sex_female  sex_male  …  who_child  who_man  who_woman  \
0              2       False      True  …      False     True      False
1              2        True     False  …      False    False       True
2              1        True     False  …      False    False       True
3              2        True     False  …      False    False       True
4              1       False      True  …      False     True      False
..           …          …       …  …         …       …        …
886            1       False      True  …      False     True      False
887            1        True     False  …      False    False       True
888            4        True     False  …      False    False       True
889            1       False      True  …      False     True      False
890            1       False      True  …      False     True      False

     deck_A  deck_B  deck_C  deck_D  deck_E  deck_F  deck_G
0     False   False    True   False   False   False   False
1     False   False    True   False   False   False   False
2     False   False    True   False   False   False   False
```

```
3     False   False    True   False   False   False   False
4     False   False    True   False   False   False   False
..      …       …      …      …       …       …       …
886   False   False    True   False   False   False   False
887   False    True   False   False   False   False   False
888   False   False    True   False   False   False   False
889   False   False    True   False   False   False   False
890   False   False    True   False   False   False   False

[891 rows x 23 columns]
```

[16]:
```
enco_embark_town = pd.get_dummies(tt['embark_town'], prefix= 'embark_town')
tt = pd.concat([tt, enco_embark_town],axis=1)
tt.drop('embark_town', axis=1, inplace=True)

tt
```

[16]:
```
     survived  pclass   age     fare  adult_male  alone  family_size  \
0           0       3  22.0   7.2500        True  False            2
1           1       1  38.0  71.2833       False  False            2
2           1       3  26.0   7.9250       False   True            1
3           1       1  35.0  53.1000       False  False            2
4           0       3  35.0   8.0500        True   True            1
..        …     …    …      …          …     …           …
886         0       2  27.0  13.0000        True   True            1
887         1       1  19.0  30.0000       False   True            1
888         0       3  24.0  23.4500       False  False            4
889         1       1  26.0  30.0000        True   True            1
890         0       3  32.0   7.7500        True   True            1

     sex_female  sex_male  embarked_C  …  deck_A  deck_B  deck_C  deck_D  \
0         False      True       False  …   False   False    True   False
1          True     False        True  …   False   False    True   False
2          True     False       False  …   False   False    True   False
3          True     False       False  …   False   False    True   False
4         False      True       False  …   False   False    True   False
..          …       …          …  …     …       …      …
886       False      True       False  …   False   False    True   False
887        True     False       False  …   False    True   False   False
888        True     False       False  …   False   False    True   False
889       False      True        True  …   False   False    True   False
890       False      True       False  …   False   False    True   False

     deck_E  deck_F  deck_G  embark_town_Cherbourg  embark_town_Queenstown  \
0     False   False   False                  False                   False
1     False   False   False                   True                   False
2     False   False   False                  False                   False
```

```
3     False   False   False                  False                   False
4     False   False   False                  False                   False
..      …       …     …                        …                       …
886   False   False   False                  False                   False
887   False   False   False                  False                   False
888   False   False   False                  False                   False
889   False   False   False                   True                   False
890   False   False   False                  False                    True

     embark_town_Southampton
0                       True
1                      False
2                       True
3                       True
4                       True
..                       …
886                     True
887                     True
888                     True
889                    False
890                    False

[891 rows x 25 columns]
```

[17]:
```
tt[tt.columns] = tt[tt.columns].astype(float)

tt
```

[17]:
```
     survived  pclass   age     fare  adult_male  alone  family_size  \
0         0.0     3.0  22.0   7.2500         1.0    0.0          2.0
1         1.0     1.0  38.0  71.2833         0.0    0.0          2.0
2         1.0     3.0  26.0   7.9250         0.0    1.0          1.0
3         1.0     1.0  35.0  53.1000         0.0    0.0          2.0
4         0.0     3.0  35.0   8.0500         1.0    1.0          1.0
..        …     …   …      …          …     …           …
886       0.0     2.0  27.0  13.0000         1.0    1.0          1.0
887       1.0     1.0  19.0  30.0000         0.0    1.0          1.0
888       0.0     3.0  24.0  23.4500         0.0    0.0          4.0
889       1.0     1.0  26.0  30.0000         1.0    1.0          1.0
890       0.0     3.0  32.0   7.7500         1.0    1.0          1.0

     sex_female  sex_male  embarked_C  …  deck_A  deck_B  deck_C  deck_D  \
0           0.0       1.0         0.0  …     0.0     0.0     1.0     0.0
1           1.0       0.0         1.0  …     0.0     0.0     1.0     0.0
2           1.0       0.0         0.0  …     0.0     0.0     1.0     0.0
3           1.0       0.0         0.0  …     0.0     0.0     1.0     0.0
4           0.0       1.0         0.0  …     0.0     0.0     1.0     0.0
```

```
..          …       …          …  …     …     …     …     …
886         0.0       1.0          0.0  …     0.0     0.0     1.0     0.0
887         1.0       0.0          0.0  …     0.0     1.0     0.0     0.0
888         1.0       0.0          0.0  …     0.0     0.0     1.0     0.0
889         0.0       1.0          1.0  …     0.0     0.0     1.0     0.0
890         0.0       1.0          0.0  …     0.0     0.0     1.0     0.0

     deck_E  deck_F  deck_G  embark_town_Cherbourg  embark_town_Queenstown  \
0       0.0     0.0     0.0                    0.0                     0.0
1       0.0     0.0     0.0                    1.0                     0.0
2       0.0     0.0     0.0                    0.0                     0.0
3       0.0     0.0     0.0                    0.0                     0.0
4       0.0     0.0     0.0                    0.0                     0.0
..      …     …     …                      …                       …
886     0.0     0.0     0.0                    0.0                     0.0
887     0.0     0.0     0.0                    0.0                     0.0
888     0.0     0.0     0.0                    0.0                     0.0
889     0.0     0.0     0.0                    1.0                     0.0
890     0.0     0.0     0.0                    0.0                     1.0

     embark_town_Southampton
0                        1.0
1                        0.0
2                        1.0
3                        1.0
4                        1.0
..                       …
886                      1.0
887                      1.0
888                      1.0
889                      0.0
890                      0.0

[891 rows x 25 columns]
```

[18]:
```
tt.corr()
```

[18]:
```
                          survived    pclass       age      fare  adult_male  \
survived                  1.000000 -0.338481 -0.052872  0.257307   -0.557080
pclass                   -0.338481  1.000000 -0.356187 -0.549500    0.094035
age                      -0.052872 -0.356187  1.000000  0.107554    0.232281
fare                      0.257307 -0.549500  0.107554  1.000000   -0.182024
adult_male               -0.557080  0.094035  0.232281 -0.182024    1.000000
alone                    -0.203367  0.135207  0.151002 -0.271832    0.404744
family_size               0.016639  0.065997 -0.236339  0.217138   -0.348143
sex_female                0.543351 -0.131900 -0.073377  0.182333   -0.908578
sex_male                 -0.543351  0.131900  0.073377 -0.182333    0.908578
```

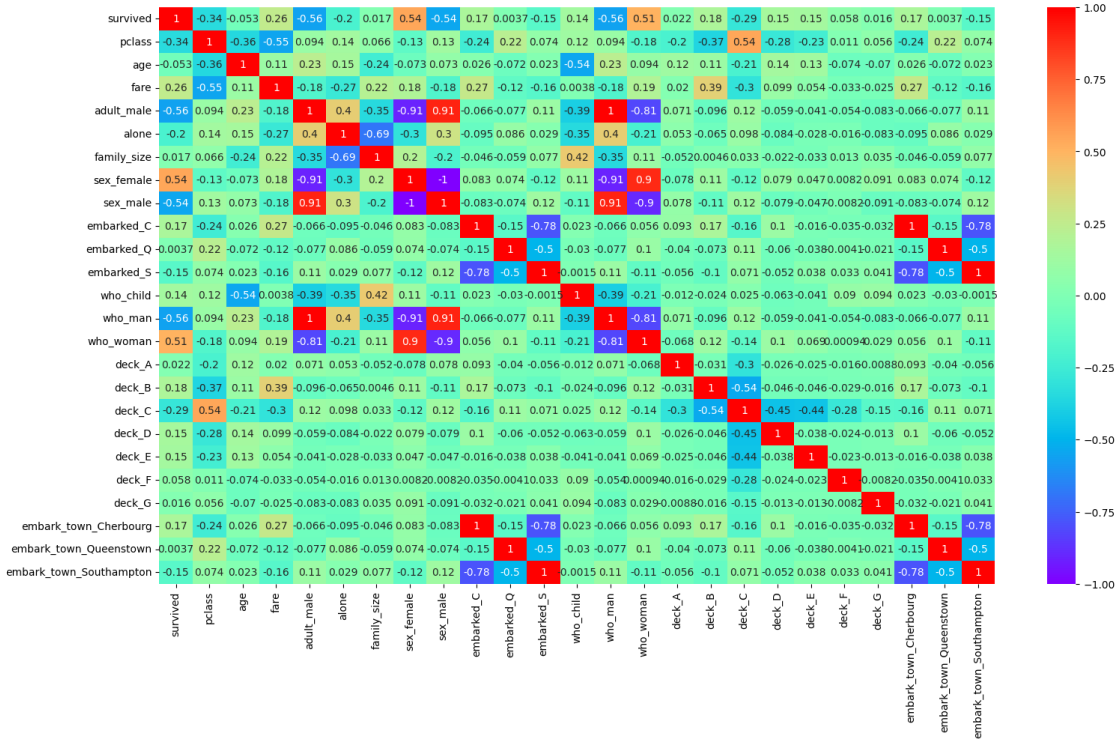
```
embarked_C                0.168240 -0.243292  0.025811  0.269335   -0.065980
embarked_Q                0.003650  0.221009 -0.071806 -0.117216   -0.076789
embarked_S               -0.149683  0.074053  0.022577 -0.162184    0.106125
who_child                 0.136107  0.121920 -0.539288  0.003753   -0.394747
who_man                  -0.557080  0.094035  0.232281 -0.182024    1.000000
who_woman                 0.506562 -0.177049  0.093639  0.191243   -0.814281
deck_A                    0.022287 -0.204934  0.120020  0.019549    0.070590
deck_B                    0.175095 -0.369572  0.105940  0.386297   -0.095696
deck_C                   -0.286690  0.541655 -0.206475 -0.303948    0.117064
deck_D                    0.150716 -0.278690  0.142034  0.098878   -0.059374
deck_E                    0.145321 -0.230091  0.126809  0.053717   -0.040505
deck_F                    0.057935  0.011063 -0.073698 -0.033093   -0.054228
deck_G                    0.016040  0.055561 -0.070334 -0.025180   -0.082709
embark_town_Cherbourg     0.168240 -0.243292  0.025811  0.269335   -0.065980
embark_town_Queenstown    0.003650  0.221009 -0.071806 -0.117216   -0.076789
embark_town_Southampton  -0.149683  0.074053  0.022577 -0.162184    0.106125

                            alone  family_size  sex_female  sex_male  \
survived                 -0.203367     0.016639    0.543351 -0.543351
pclass                    0.135207     0.065997   -0.131900  0.131900
age                       0.151002    -0.236339   -0.073377  0.073377
fare                     -0.271832     0.217138    0.182333 -0.182333
adult_male                0.404744    -0.348143   -0.908578  0.908578
alone                     1.000000    -0.690922   -0.303646  0.303646
family_size              -0.690922     1.000000    0.200988 -0.200988
sex_female               -0.303646     0.200988    1.000000 -1.000000
sex_male                  0.303646    -0.200988   -1.000000  1.000000
embarked_C               -0.095298    -0.046215    0.082853 -0.082853
embarked_Q                0.086464    -0.058592    0.074115 -0.074115
embarked_S                0.029074     0.077359   -0.119224  0.119224
who_child                -0.347400     0.416472    0.111141 -0.111141
who_man                   0.404744    -0.348143   -0.908578  0.908578
who_woman                -0.211036     0.107192    0.896214 -0.896214
deck_A                    0.052762    -0.051767   -0.078271  0.078271
deck_B                   -0.064914     0.004620    0.109689 -0.109689
deck_C                    0.098372     0.032640   -0.122877  0.122877
deck_D                   -0.083664    -0.021566    0.079248 -0.079248
deck_E                   -0.028179    -0.033466    0.047003 -0.047003
deck_F                   -0.015972     0.013003    0.008202 -0.008202
deck_G                   -0.082709     0.035206    0.091031 -0.091031
embark_town_Cherbourg    -0.095298    -0.046215    0.082853 -0.082853
embark_town_Queenstown    0.086464    -0.058592    0.074115 -0.074115
embark_town_Southampton   0.029074     0.077359   -0.119224  0.119224

                         embarked_C  …    deck_A    deck_B    deck_C  \
survived                   0.168240  …  0.022287  0.175095 -0.286690
pclass                    -0.243292  … -0.204934 -0.369572  0.541655
```

```
age                          0.025811  …  0.120020  0.105940 -0.206475
fare                         0.269335  …  0.019549  0.386297 -0.303948
adult_male                  -0.065980  …  0.070590 -0.095696  0.117064
alone                       -0.095298  …  0.052762 -0.064914  0.098372
family_size                 -0.046215  … -0.051767  0.004620  0.032640
sex_female                   0.082853  … -0.078271  0.109689 -0.122877
sex_male                    -0.082853  …  0.078271 -0.109689  0.122877
embarked_C                   1.000000  …  0.093040  0.168642 -0.162513
embarked_Q                  -0.148258  … -0.040246 -0.072579  0.113335
embarked_S                  -0.782742  … -0.056180 -0.102063  0.071046
who_child                    0.023201  … -0.011925 -0.023809  0.025325
who_man                     -0.065980  …  0.070590 -0.095696  0.117064
who_woman                    0.055524  … -0.067551  0.116831 -0.140518
deck_A                       0.093040  …  1.000000 -0.030880 -0.298039
deck_B                       0.168642  … -0.030880  1.000000 -0.537473
deck_C                      -0.162513  … -0.298039 -0.537473  1.000000
deck_D                       0.102977  … -0.025663 -0.046280 -0.446676
deck_E                      -0.015939  … -0.025256 -0.045547 -0.439600
deck_F                      -0.034726  … -0.015923 -0.028715 -0.277143
deck_G                      -0.032371  … -0.008787 -0.015847 -0.152949
embark_town_Cherbourg        1.000000  …  0.093040  0.168642 -0.162513
embark_town_Queenstown      -0.148258  … -0.040246 -0.072579  0.113335
embark_town_Southampton     -0.782742  … -0.056180 -0.102063  0.071046

                             deck_D    deck_E    deck_F    deck_G  \
survived                   0.150716  0.145321  0.057935  0.016040
pclass                    -0.278690 -0.230091  0.011063  0.055561
age                        0.142034  0.126809 -0.073698 -0.070334
fare                       0.098878  0.053717 -0.033093 -0.025180
adult_male                -0.059374 -0.040505 -0.054228 -0.082709
alone                     -0.083664 -0.028179 -0.015972 -0.082709
family_size               -0.021566 -0.033466  0.013003  0.035206
sex_female                 0.079248  0.047003  0.008202  0.091031
sex_male                  -0.079248 -0.047003 -0.008202 -0.091031
embarked_C                 0.102977 -0.015939 -0.034726 -0.032371
embarked_Q                -0.060318 -0.037897 -0.004113 -0.020654
embarked_S                -0.052254  0.037812  0.033010  0.041356
who_child                 -0.062856 -0.041108  0.089818  0.094001
who_man                   -0.059374 -0.040505 -0.054228 -0.082709
who_woman                  0.102866  0.069056  0.000936  0.028588
deck_A                    -0.025663 -0.025256 -0.015923 -0.008787
deck_B                    -0.046280 -0.045547 -0.028715 -0.015847
deck_C                    -0.446676 -0.439600 -0.277143 -0.152949
deck_D                     1.000000 -0.037852 -0.023864 -0.013170
deck_E                    -0.037852  1.000000 -0.023486 -0.012961
deck_F                    -0.023864 -0.023486  1.000000 -0.008171
deck_G                    -0.013170 -0.012961 -0.008171  1.000000
```

```
embark_town_Cherbourg    0.102977 -0.015939 -0.034726 -0.032371
embark_town_Queenstown  -0.060318 -0.037897 -0.004113 -0.020654
embark_town_Southampton -0.052254  0.037812  0.033010  0.041356

                         embark_town_Cherbourg  embark_town_Queenstown  \
survived                              0.168240                0.003650
pclass                               -0.243292                0.221009
age                                   0.025811               -0.071806
fare                                  0.269335               -0.117216
adult_male                           -0.065980               -0.076789
alone                                -0.095298                0.086464
family_size                          -0.046215               -0.058592
sex_female                            0.082853                0.074115
sex_male                             -0.082853               -0.074115
embarked_C                            1.000000               -0.148258
embarked_Q                           -0.148258                1.000000
embarked_S                           -0.782742               -0.499421
who_child                             0.023201               -0.029861
who_man                              -0.065980               -0.076789
who_woman                             0.055524                0.100544
deck_A                                0.093040               -0.040246
deck_B                                0.168642               -0.072579
deck_C                               -0.162513                0.113335
deck_D                                0.102977               -0.060318
deck_E                               -0.015939               -0.037897
deck_F                               -0.034726               -0.004113
deck_G                               -0.032371               -0.020654
embark_town_Cherbourg                 1.000000               -0.148258
embark_town_Queenstown               -0.148258                1.000000
embark_town_Southampton              -0.782742               -0.499421

                         embark_town_Southampton
survived                               -0.149683
pclass                                  0.074053
age                                     0.022577
fare                                   -0.162184
adult_male                              0.106125
alone                                   0.029074
family_size                             0.077359
sex_female                             -0.119224
sex_male                                0.119224
embarked_C                             -0.782742
embarked_Q                             -0.499421
embarked_S                              1.000000
who_child                              -0.001534
who_man                                 0.106125
who_woman                              -0.111914
```

```
deck_A                          -0.056180
deck_B                          -0.102063
deck_C                           0.071046
deck_D                          -0.052254
deck_E                           0.037812
deck_F                           0.033010
deck_G                           0.041356
embark_town_Cherbourg           -0.782742
embark_town_Queenstown          -0.499421
embark_town_Southampton          1.000000

[25 rows x 25 columns]
```

[19]:
```
plt.figure(figsize=(18, 10))

sns.heatmap(tt.corr(), annot=True, cmap='rainbow')
plt.show()
```



[20]:
```
corr_matrix = tt.corr().abs()

threshold = 0.8

# Get column and index names where correlation is strong
```

```
col_names = corr_matrix.columns.tolist()
index_names = corr_matrix.index.tolist()

# Find strongly correlated pairs (excluding self-correlations)
strong_pairs = []
for i in range(len(col_names)):
    for j in range(i + 1, len(col_names)):  # Avoid redundant comparisons
        if corr_matrix.iloc[i, j] > threshold:
            strong_pairs.append((col_names[i], col_names[j]))

print("Strongly correlated pairs (above threshold", threshold, "):")

for pair in strong_pairs:
    print(pair)
```

```
Strongly correlated pairs (above threshold 0.8 ):
('adult_male', 'sex_female')
('adult_male', 'sex_male')
('adult_male', 'who_man')
('adult_male', 'who_woman')
('sex_female', 'sex_male')
('sex_female', 'who_man')
('sex_female', 'who_woman')
('sex_male', 'who_man')
('sex_male', 'who_woman')
('embarked_C', 'embark_town_Cherbourg')
('embarked_Q', 'embark_town_Queenstown')
('embarked_S', 'embark_town_Southampton')
('who_man', 'who_woman')
```

[21]:
```
tt = tt.drop(columns= list(set([pair[1] for pair in strong_pairs])))
```

[22]:
```
tt
```

[22]:

	survived	pclass	age	fare	adult_male	alone	family_size	\
0	0.0	3.0	22.0	7.2500	1.0	0.0	2.0	
1	1.0	1.0	38.0	71.2833	0.0	0.0	2.0	
2	1.0	3.0	26.0	7.9250	0.0	1.0	1.0	
3	1.0	1.0	35.0	53.1000	0.0	0.0	2.0	
4	0.0	3.0	35.0	8.0500	1.0	1.0	1.0	
..	…	…	…	…	…	…	…	
886	0.0	2.0	27.0	13.0000	1.0	1.0	1.0	
887	1.0	1.0	19.0	30.0000	0.0	1.0	1.0	
888	0.0	3.0	24.0	23.4500	0.0	0.0	4.0	
889	1.0	1.0	26.0	30.0000	1.0	1.0	1.0	
890	0.0	3.0	32.0	7.7500	1.0	1.0	1.0	

```
     embarked_C  embarked_Q  embarked_S  who_child  deck_A  deck_B  deck_C  \
0           0.0         0.0         1.0        0.0     0.0     0.0     1.0
1           1.0         0.0         0.0        0.0     0.0     0.0     1.0
2           0.0         0.0         1.0        0.0     0.0     0.0     1.0
3           0.0         0.0         1.0        0.0     0.0     0.0     1.0
4           0.0         0.0         1.0        0.0     0.0     0.0     1.0
..          …           …           …          …       …       …       …
886         0.0         0.0         1.0        0.0     0.0     0.0     1.0
887         0.0         0.0         1.0        0.0     0.0     1.0     0.0
888         0.0         0.0         1.0        0.0     0.0     0.0     1.0
889         1.0         0.0         0.0        0.0     0.0     0.0     1.0
890         0.0         1.0         0.0        0.0     0.0     0.0     1.0

     deck_D  deck_E  deck_F  deck_G
0       0.0     0.0     0.0     0.0
1       0.0     0.0     0.0     0.0
2       0.0     0.0     0.0     0.0
3       0.0     0.0     0.0     0.0
4       0.0     0.0     0.0     0.0
..      …       …       …       …
886     0.0     0.0     0.0     0.0
887     0.0     0.0     0.0     0.0
888     0.0     0.0     0.0     0.0
889     0.0     0.0     0.0     0.0
890     0.0     0.0     0.0     0.0

[891 rows x 18 columns]
```

[23]:
```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
tt[tt.columns[1:]] = sc.fit_transform(tt[tt.columns[1:]])
```

[24]:
```
tt
```

[24]:
```
     survived  pclass       age      fare  adult_male  alone  family_size  \
0         0.0     1.0  0.271174  0.014151         1.0    0.0          0.1
1         1.0     0.0  0.472229  0.139136         0.0    0.0          0.1
2         1.0     1.0  0.321438  0.015469         0.0    1.0          0.0
3         1.0     0.0  0.434531  0.103644         0.0    0.0          0.1
4         0.0     1.0  0.434531  0.015713         1.0    1.0          0.0
..        …       …    …         …          …       …      …
886       0.0     0.5  0.334004  0.025374         1.0    1.0          0.0
887       1.0     0.0  0.233476  0.058556         0.0    1.0          0.0
888       0.0     1.0  0.296306  0.045771         0.0    0.0          0.3
889       1.0     0.0  0.321438  0.058556         1.0    1.0          0.0
890       0.0     1.0  0.396833  0.015127         1.0    1.0          0.0
```

```
     embarked_C  embarked_Q  embarked_S  who_child  deck_A  deck_B  deck_C  \
0           0.0         0.0         1.0        0.0     0.0     0.0     1.0
1           1.0         0.0         0.0        0.0     0.0     0.0     1.0
2           0.0         0.0         1.0        0.0     0.0     0.0     1.0
3           0.0         0.0         1.0        0.0     0.0     0.0     1.0
4           0.0         0.0         1.0        0.0     0.0     0.0     1.0
..          …           …           …          …       …       …       …
886         0.0         0.0         1.0        0.0     0.0     0.0     1.0
887         0.0         0.0         1.0        0.0     0.0     1.0     0.0
888         0.0         0.0         1.0        0.0     0.0     0.0     1.0
889         1.0         0.0         0.0        0.0     0.0     0.0     1.0
890         0.0         1.0         0.0        0.0     0.0     0.0     1.0

     deck_D  deck_E  deck_F  deck_G
0       0.0     0.0     0.0     0.0
1       0.0     0.0     0.0     0.0
2       0.0     0.0     0.0     0.0
3       0.0     0.0     0.0     0.0
4       0.0     0.0     0.0     0.0
..      …       …       …       …
886     0.0     0.0     0.0     0.0
887     0.0     0.0     0.0     0.0
888     0.0     0.0     0.0     0.0
889     0.0     0.0     0.0     0.0
890     0.0     0.0     0.0     0.0

[891 rows x 18 columns]
```

[25]:
```
from sklearn.model_selection import train_test_split

X = tt.drop('survived', axis=1)
y = tt['survived']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

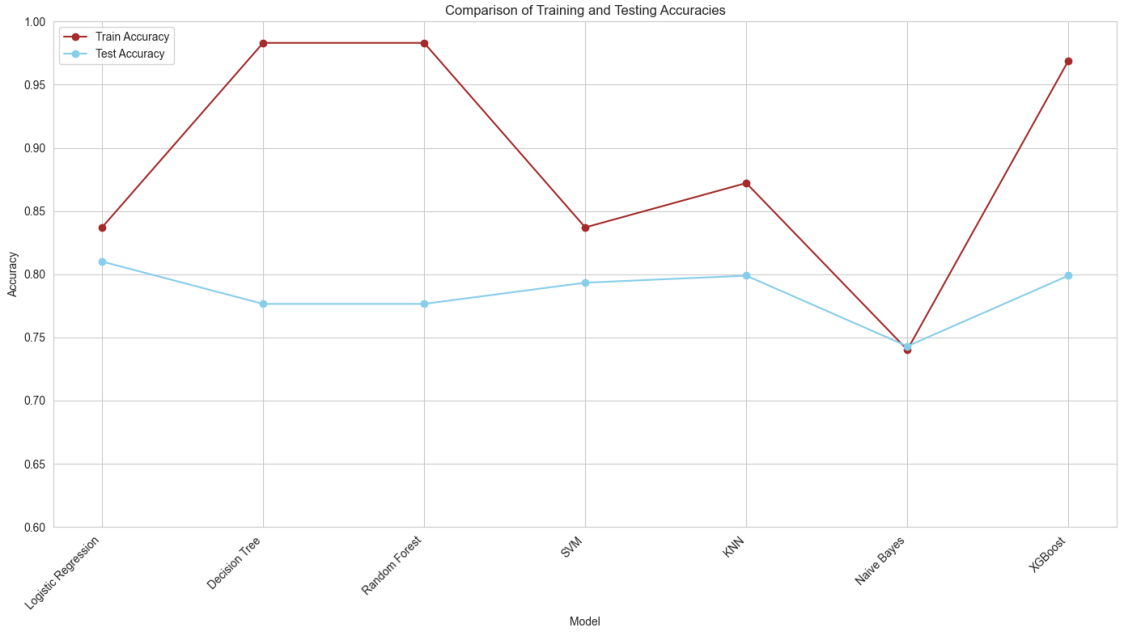
[26]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score

models = {
    'Logistic Regression': LogisticRegression(),
```

```
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'SVM': SVC(),
    'KNN': KNeighborsClassifier(),
    'Naive Bayes': GaussianNB(),
    'XGBoost': XGBClassifier()
}

model_accuracies = {}

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred_train = model.predict(X_train)
    train_accuracy = accuracy_score(y_train, y_pred_train)

    y_pred_test = model.predict(X_test)
    test_accuracy = accuracy_score(y_test, y_pred_test)

    model_accuracies[name] = {'Train Accuracy': train_accuracy, 'Test Accuracy':
 ↪ test_accuracy}

    print(f'{name}: Train Accuracy = {train_accuracy:.4f}, Test Accuracy =␣
 ↪{test_accuracy:.4f}')
```

```
Logistic Regression: Train Accuracy = 0.8371, Test Accuracy = 0.8101
Decision Tree: Train Accuracy = 0.9831, Test Accuracy = 0.7765
Random Forest: Train Accuracy = 0.9831, Test Accuracy = 0.7765
SVM: Train Accuracy = 0.8371, Test Accuracy = 0.7933
KNN: Train Accuracy = 0.8722, Test Accuracy = 0.7989
Naive Bayes: Train Accuracy = 0.7402, Test Accuracy = 0.7430
XGBoost: Train Accuracy = 0.9691, Test Accuracy = 0.7989
```

[27]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

accuracy_df = pd.DataFrame.from_dict(model_accuracies, orient='index')

sns.set_style("whitegrid")

plt.figure(figsize=(14, 8))
plt.plot(accuracy_df.index, accuracy_df["Train Accuracy"], marker='o',␣
 ↪linestyle='-', color="brown", label="Train Accuracy")
plt.plot(accuracy_df.index, accuracy_df["Test Accuracy"], marker='o',␣
 ↪linestyle='-', color="skyblue", label="Test Accuracy")

plt.xlabel("Model")
```

```
plt.ylabel("Accuracy")
plt.title("Comparison of Training and Testing Accuracies")
plt.xticks(rotation=45, ha="right")
plt.ylim(0.6, 1.0)

plt.legend()

plt.tight_layout()
plt.show()
```



[28]:
```
from tpot import TPOTClassifier

tpot = TPOTClassifier(generations=5, population_size=50, verbosity=2,␣
↪random_state=42)

tpot.fit(X_train, y_train)

print(tpot.score(X_test, y_test))
```

```
c:\Users\shiva\AppData\Local\Programs\Python\Python312\Lib\site-
packages\tpot\builtins\__init__.py:36: UserWarning: Warning: optional dependency
`torch` is not available. - skipping import of NN models.
  warnings.warn("Warning: optional dependency `torch` is not available. -
skipping import of NN models.")

is_classifier
is_regressor
```

```
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_classifier
is_classifier
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_classifier
is_classifier
is_classifier
```

```
c:\Users\shiva\AppData\Local\Programs\Python\Python312\Lib\site-
packages\sklearn\base.py:1230: FutureWarning: passing a class to None is
deprecated and will be removed in 1.8. Use an instance of the class instead.
  warnings.warn(
c:\Users\shiva\AppData\Local\Programs\Python\Python312\Lib\site-
packages\sklearn\base.py:1270: FutureWarning: passing a class to None is
deprecated and will be removed in 1.8. Use an instance of the class instead.
  warnings.warn(
```

```
is_classifier
is_classifier
is_classifier
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
```

```
is_regressor
is_classifier
is_regressor
is_classifier
is_classifier
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
```

```
Version 0.12.2 of tpot is outdated. Version 1.0.0 was released Wednesday
February 26, 2025.
```

```
Optimization Progress:   0%|          | 0/300 [00:00<?, ?pipeline/s]


Generation 1 - Current best internal CV score: 0.8328769821727569

Generation 2 - Current best internal CV score: 0.8398995370826355

Generation 3 - Current best internal CV score: 0.8399290849995076

Generation 4 - Current best internal CV score: 0.8413375357037328

Generation 5 - Current best internal CV score: 0.8413375357037328

Best pipeline: XGBClassifier(XGBClassifier(input_matrix, learning_rate=1.0,
max_depth=9, min_child_weight=7, n_estimators=100, n_jobs=1,
subsample=0.15000000000000002, verbosity=0), learning_rate=0.1, max_depth=4,
min_child_weight=10, n_estimators=100, n_jobs=1, subsample=0.9000000000000001,
verbosity=0)
0.8212290502793296
```

[31]:
```
best_pipeline = tpot.fitted_pipeline_

# Predict on test data using the best pipeline
y_pred_test = best_pipeline.predict(X_test)

# Evaluate accuracy and other metrics
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score, roc_auc_score, classification_report

accuracy = accuracy_score(y_test, y_pred_test)
precision = precision_score(y_test, y_pred_test)
recall = recall_score(y_test, y_pred_test)
f1 = f1_score(y_test, y_pred_test)
roc_auc = roc_auc_score(y_test, y_pred_test)
report = classification_report(y_test, y_pred_test)
```

```
print("Final Model Evaluation on Test Data:")
print(f"  Accuracy: {accuracy:.4f}")
print(f"  Precision: {precision:.4f}")
print(f"  Recall: {recall:.4f}")
print(f"  F1-Score: {f1:.4f}")
print(f"  ROC-AUC Score: {roc_auc:.4f}")

# Print classification report
print("\nClassification Report:")
print(report)
```

```
Final Model Evaluation on Test Data:
  Accuracy: 0.8212
  Precision: 0.8088
  Recall: 0.7432
  F1-Score: 0.7746
  ROC-AUC Score: 0.8097

Classification Report:
              precision    recall  f1-score   support

         0.0       0.83      0.88      0.85       105
         1.0       0.81      0.74      0.77        74

    accuracy                           0.82       179
   macro avg       0.82      0.81      0.81       179
weighted avg       0.82      0.82      0.82       179
```

[ ]: