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

import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
#warnings.filterwarnings("ignore", category=DeprecationWarning)
#warnings.filterwarnings("ignore")

from subprocess import check_output
print(check_output(["ls", "../content/gender_submission.csv"]).decode("utf8"))
```

```
../content/gender_submission.csv
```

```
sns.__version__
```

```
'0.12.2'
```

```
df_train = pd.read_csv("../content/train.csv")
df_test = pd.read_csv("../content/test.csv")
```

```
df_train.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence	female	38.0	1	0	PC 17599	71.2833	C85

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
df_test.head()
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
			Myles								

```
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
```

```
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

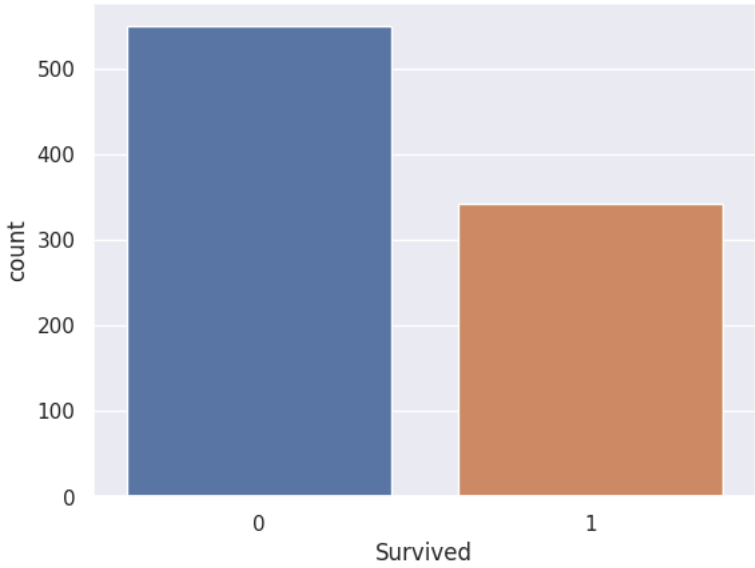
```
df_train.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
df_test.describe()
```

	PassengerId	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	3.000000	76.000000	8.000000	9.000000	512.329200

```
sns.countplot(x='Survived', data=df_train);
```




```
print(df_train.Survived.sum()/df_train.Survived.count())
```

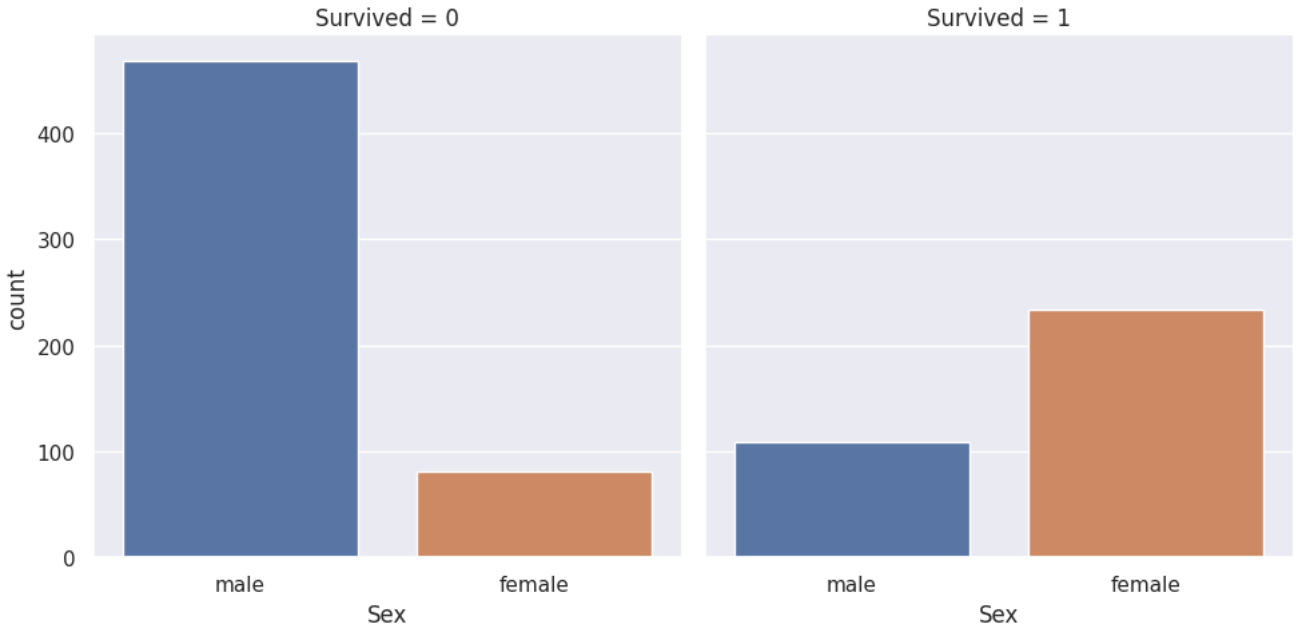
```
0.3838383838383838
```

```
df_train.groupby(['Survived','Sex'])['Survived'].count()
```

```
Survived  Sex
0         female     81
          male      468
```

```
1         female    233
          male      109
Name: Survived, dtype: int64
```

```
sns.catplot(x='Sex', col='Survived', kind='count', data=df_train);
```

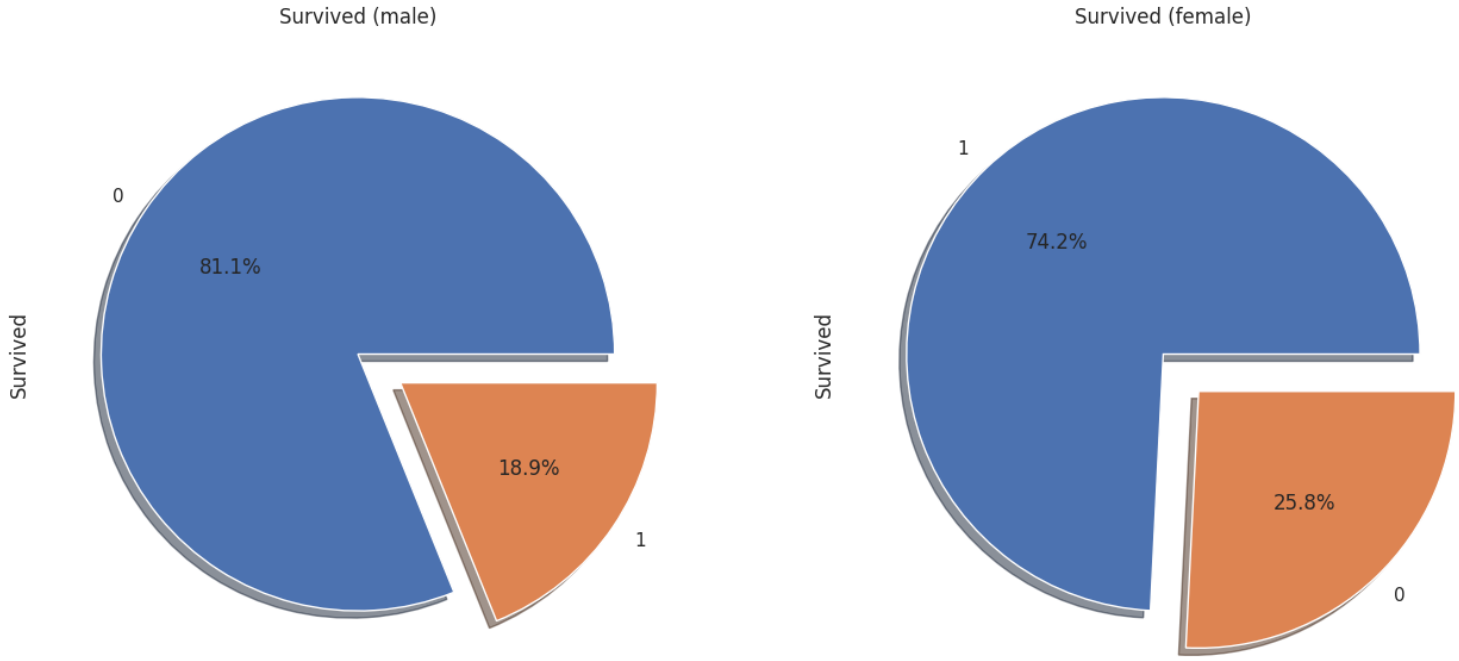



```
print("% of women survived: " , df_train[df_train.Sex == 'female'].Survived.sum()/df_train[df_train.Sex == 'female'].Survived.count())
print("% of men survived:   " , df_train[df_train.Sex == 'male'].Survived.sum()/df_train[df_train.Sex == 'male'].Survived.count())
```

```
% of women survived:  0.7420382165605095
% of men survived:    0.18890814558058924
```

```
f,ax=plt.subplots(1,2,figsize=(16,7))
df_train['Survived'][df_train['Sex']=='male'].value_counts().plot.pie(explode=[0,0.2],autopct='%1.1f%%',ax=ax[0],shadow=True)
df_train['Survived'][df_train['Sex']=='female'].value_counts().plot.pie(explode=[0,0.2],autopct='%1.1f%%',ax=ax[1],shadow=True)
ax[0].set_title('Survived (male)')
ax[1].set_title('Survived (female)')

plt.show()
```




```
pd.crosstab(df_train.Pclass, df_train.Survived, margins=True).style.background_gradient(cmap='autumn_r')
```

Survived	0	1	All
Pclass			
1	80	136	216
2	97	87	184

```
print("% of survivals in")
print("Pclass=1 : ", df_train.Survived[df_train.Pclass == 1].sum()/df_train[df_train.Pclass == 1].Survived.count())
print("Pclass=2 : ", df_train.Survived[df_train.Pclass == 2].sum()/df_train[df_train.Pclass == 2].Survived.count())
print("Pclass=3 : ", df_train.Survived[df_train.Pclass == 3].sum()/df_train[df_train.Pclass == 3].Survived.count())
```

```
% of survivals in
Pclass=1 :  0.6296296296296297
Pclass=2 :  0.47282608695652173
Pclass=3 :  0.24236252545824846
```

```
pd.crosstab([df_train.Sex, df_train.Survived], df_train.Pclass, margins=True).style.background_gradient(cmap='autumn_r')
```

	Pclass	1	2	3	All
Sex	**Survived**				
female	**0**	3	6	72	81
	1	91	70	72	233
male	**0**	77	91	300	468
	1	45	17	47	109
All		216	184	491	891

```
pd.crosstab([df_train.Survived], [df_train.Sex, df_train.Pclass, df_train.Embarked], margins=True)
```

Sex	female									male									All
Pclass	1			2			3			1			2			3			
Embarked	C	Q	S	C	Q	S	C	Q	S	C	Q	S	C	Q	S	C	Q	S	
Survived																			
0	1	0	2	0	0	6	8	9	55	25	1	51	8	1	82	33	36	231	549
1	42	1	46	7	2	61	15	24	33	17	0	28	2	0	15	10	3	34	340
All	43	1	48	7	2	67	23	33	88	42	1	79	10	1	97	43	39	265	889

```
# model 3
df_test['Survived'] = 0
# all women survived
df_test.loc[ (df_test.Sex == 'female'), 'Survived'] = 1
# except for those in Pclass 3 and embarked in S
df_test.loc[ (df_test.Sex == 'female') & (df_test.Pclass == 3) & (df_test.Embarked == 'S') , 'Survived'] = 0
#df_test[['PassengerId', 'Survived']].to_csv('embarked_pclass_sex.csv', index=False)
```

```
for df in [df_train, df_test]:
    df['Age_bin']=np.nan
    for i in range(8,0,-1):
        df.loc[ df['Age'] <= i*10, 'Age_bin'] = i
```

```
print(df_train[['Age' , 'Age_bin']].head(10))
```

```
    Age  Age_bin
0  22.0      3.0
1  38.0      4.0
2  26.0      3.0
3  35.0      4.0
4  35.0      4.0
5   NaN      NaN
6  54.0      6.0
7   2.0      1.0
8  27.0      3.0
9  14.0      2.0
```

```
pd.crosstab([df_train.Sex, df_train.Survived], [df_train.Age_bin, df_train.Pclass], margins=True).style.background_gradient(cmap='autumn_r')
```

Sex	Survived	Age_bin 1.000000			2.000000			3.000000			4.000000			5.000000			6.000000			7.000000			8.000000		All
	Pclass	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	3	
female	0	1	0	11	0	0	12	1	3	16	0	1	8	1	1	8	0	1	0	0	0	0	0	0	64

```
# in Pclass 1 and 2 all men in Age_bin = 1 survived
df_test.loc[ (df_test.Sex == 'male') & (df_test.Pclass == 1) & (df_test.Age_bin == 1), 'Survived'] = 1
df_test.loc[ (df_test.Sex == 'male') & (df_test.Pclass == 2) & (df_test.Age_bin == 1), 'Survived'] = 1
```

```
pd.crosstab([df_train.Sex, df_train.Survived], [df_train.SibSp, df_train.Pclass], margins=True).style.background_gradient(cmap='autumn_r')
```

Sex	Survived	SibSp 0			1			2			3		4	5	8	All
	Pclass	1	2	3	1	2	3	1	2	3	1	2	3	3	3	3	
female	0	1	3	33	2	3	21	0	0	3	0	0	7	4	1	3	81
	1	48	41	48	38	25	17	3	3	4	2	1	1	2	0	0	233
male	0	59	67	235	16	20	35	1	4	7	1	0	4	11	4	4	468
	1	29	9	35	15	7	10	1	1	1	0	0	0	1	0	0	109
All		137	120	351	71	55	83	5	8	15	3	1	12	18	5	7	891

```
# all females with SibSp > 7 died
df_test.loc[ (df_test.Sex == 'female') & (df_test.SibSp > 7) , 'Survived'] = 0
```

```
pd.crosstab([df_train.Sex, df_train.Survived], [df_train.Parch, df_train.Pclass], margins=True).style.background_gradient(cmap='autumn_r')
```

Sex	Survived	Parch 0			1			2			3		4	5	6	All
	Pclass	1	2	3	1	2	3	1	2	3	2	3	1	3	3	3	
female	0	1	5	35	0	1	13	2	0	17	0	1	0	2	3	1	81
	1	63	40	50	17	17	12	11	11	8	2	1	0	0	1	0	233
male	0	63	81	260	10	7	22	3	3	15	0	1	1	1	1	0	468
	1	36	8	36	4	7	8	5	2	3	0	0	0	0	0	0	109
All		163	134	381	31	32	55	21	16	43	2	3	1	3	5	1	891

```
for df in [df_train, df_test]:
    df['Fare_bin']=np.nan
    for i in range(12,0,-1):
        df.loc[ df['Fare'] <= i*50, 'Fare_bin'] = i
```

```
pd.crosstab([df_train.Sex, df_train.Survived], [df_train.Fare_bin, df_train.Pclass], margins=True).style.background_gradient(cmap='autumn_r')
```

Sex	Survived	Fare_bin 1.000000			2.000000			3.000000	4.000000	5.000000	6.000000	11.000000	All
	Pclass	1	2	3	1	2	3	1	1	1	1	1	
female	0	1	6	69	0	0	3	0	2	0	0	0	81
	1	11	68	72	48	2	0	15	5	7	4	1	233
male	0	42	86	294	23	5	6	5	1	4	2	0	468
	1	23	17	42	15	0	5	4	1	0	0	2	109
All		77	177	477	86	7	14	24	9	11	6	3	891

```
# males in Fare_bin = 11 survived
df_test.loc[ (df_test.Sex == 'male') & (df_test.Fare_bin == 11), 'Survived'] = 1
```

```
df_test.drop(['Survived'],axis=1,inplace=True)
```

```
df_train_ml = df_train.copy()
df_test_ml = df_test.copy()
```

```
df_train_ml = pd.get_dummies(df_train_ml, columns=['Sex', 'Embarked', 'Pclass'], drop_first=True)
df_train_ml.drop(['PassengerId','Name','Ticket', 'Cabin', 'Age_bin', 'Fare_bin'],axis=1,inplace=True)
```

```
df_train_ml.dropna(inplace=True)
```

```
passenger_id = df_test_ml['PassengerId']
df_test_ml = pd.get_dummies(df_test_ml, columns=['Sex', 'Embarked', 'Pclass'], drop_first=True)
df_test_ml.drop(['PassengerId','Name','Ticket', 'Cabin', 'Age_bin', 'Fare_bin'],axis=1,inplace=True)
```

```
df_train_ml.head(10)
```

	Survived	Age	SibSp	Parch	Fare	Sex_male	Embarked_Q	Embarked_S	Pclass_2	Pclass_3
0	0	22.0	1	0	7.2500	1	0	1	0	1
1	1	38.0	1	0	71.2833	0	0	0	0	0
2	1	26.0	0	0	7.9250	0	0	1	0	1
3	1	35.0	1	0	53.1000	0	0	1	0	0
4	0	35.0	0	0	8.0500	1	0	1	0	1
6	0	54.0	0	0	51.8625	1	0	1	0	0
7	0	2.0	3	1	21.0750	1	0	1	0	1
8	1	27.0	0	2	11.1333	0	0	1	0	1
9	1	14.0	1	0	30.0708	0	0	0	1	0
10	1	4.0	1	1	16.7000	0	0	1	0	1

```
df_train_ml.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 714 entries, 0 to 890
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Survived    714 non-null    int64
 1   Age         714 non-null    float64
 2   SibSp       714 non-null    int64
 3   Parch       714 non-null    int64
 4   Fare        714 non-null    float64
 5   Sex_male    714 non-null    uint8
 6   Embarked_Q  714 non-null    uint8
 7   Embarked_S  714 non-null    uint8
 8   Pclass_2    714 non-null    uint8
 9   Pclass_3    714 non-null    uint8
dtypes: float64(2), int64(3), uint8(5)
memory usage: 37.0 KB
```

```
df_test_ml.info()
```

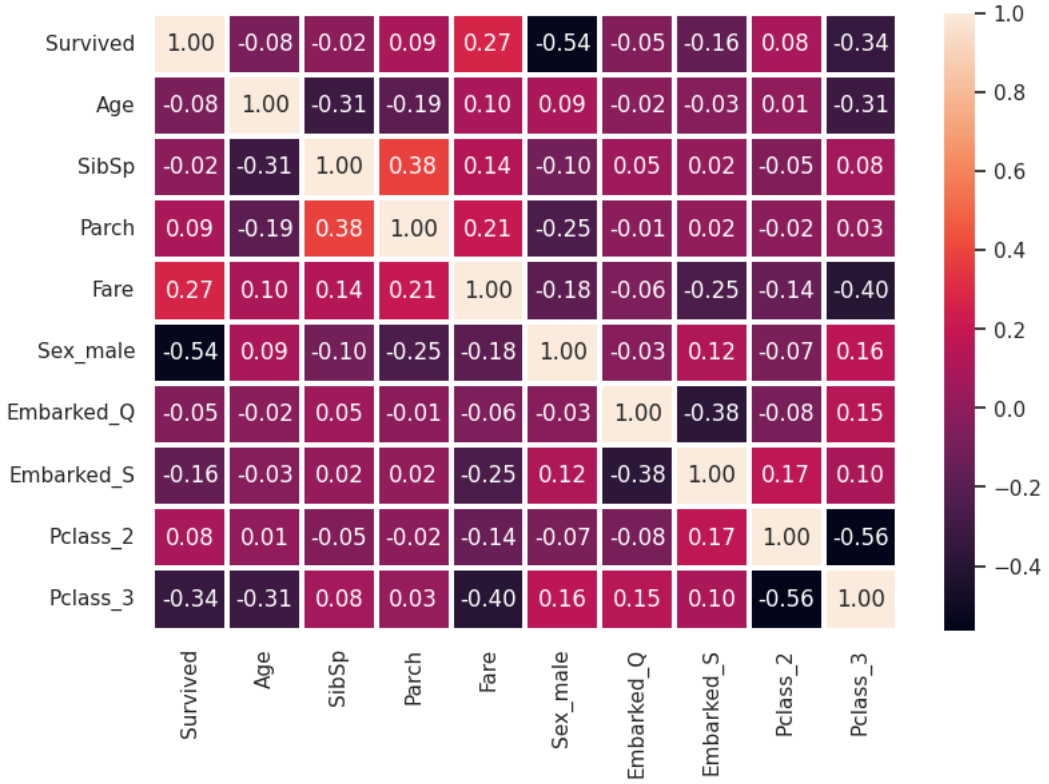
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Age         332 non-null    float64
 1   SibSp       418 non-null    int64
 2   Parch       418 non-null    int64
 3   Fare        417 non-null    float64
 4   Sex_male    418 non-null    uint8
 5   Embarked_Q  418 non-null    uint8
 6   Embarked_S  418 non-null    uint8
 7   Pclass_2    418 non-null    uint8
 8   Pclass_3    418 non-null    uint8
dtypes: float64(2), int64(2), uint8(5)
memory usage: 15.2 KB
```

```
df_test_ml.head(10)
```

Age	SibSp	Parch	Fare	Sex_male	Embarked_Q	Embarked_S	Pclass_2	Pclass_3

```
corr = df_train_ml.corr()

f,ax = plt.subplots(figsize=(9,6))
sns.heatmap(corr, annot = True, linewidths=1.5 , fmt = '.2f',ax=ax)
plt.show()
```




```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# for df_train_ml
scaler.fit(df_train_ml.drop('Survived',axis=1))
scaled_features = scaler.transform(df_train_ml.drop('Survived',axis=1))
df_train_ml_sc = pd.DataFrame(scaled_features, columns=df_train_ml.columns[:-1])

# for df_test_ml
df_test_ml.fillna(df_test_ml.mean(), inplace=True)
# scaler.fit(df_test_ml)
scaled_features = scaler.transform(df_test_ml)
df_test_ml_sc = pd.DataFrame(scaled_features, columns=df_test_ml.columns)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df_train_ml.drop('Survived',axis=1), df_train_ml['Survived'], test_size=0.30, random_state=101)
X_train_sc, X_test_sc, y_train_sc, y_test_sc = train_test_split(df_train_ml_sc, df_train_ml['Survived'], test_size=0.30, random_state=101)
```

```
# unscaled
X_train_all = df_train_ml.drop('Survived',axis=1)
y_train_all = df_train_ml['Survived']
X_test_all = df_test_ml

# scaled
X_train_all_sc = df_train_ml_sc
y_train_all_sc = df_train_ml['Survived']
X_test_all_sc = df_test_ml_sc
```

```
X_test_all.fillna(X_test_all.mean(), inplace=True)
print("*")
```

```
*
```

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

Naive Bayes

```
from sklearn.naive_bayes import GaussianNB
gnb=GaussianNB()
```

```
gnb.fit(X_train,y_train)
pred_gnb = gnb.predict(X_test)
print(confusion_matrix(y_test, pred_gnb))
print(classification_report(y_test, pred_gnb))
print(accuracy_score(y_test, pred_gnb))
```

```
[[107  16]
 [ 36  56]]
              precision    recall  f1-score   support

           0       0.75      0.87      0.80       123
           1       0.78      0.61      0.68        92

    accuracy                           0.76       215
   macro avg       0.76      0.74      0.74       215
weighted avg       0.76      0.76      0.75       215

0.7581395348837209
```

KNN - KNeighborsClassifier

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=20)
knn.fit(X_train_sc,y_train_sc)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier(n_neighbors=20)
```

```
pred_knn = knn.predict(X_test)
print(confusion_matrix(y_test, pred_knn))
print(classification_report(y_test, pred_knn))
print(accuracy_score(y_test, pred_knn))
```

```
[[102  21]
 [ 50  42]]
              precision    recall  f1-score   support

           0       0.67      0.83      0.74       123
           1       0.67      0.46      0.54        92

    accuracy                           0.67       215
   macro avg       0.67      0.64      0.64       215
weighted avg       0.67      0.67      0.66       215

0.6697674418604651
```

```
knn.fit(X_train_all, y_train_all)
pred_all_knn = knn.predict(X_test_all)
```

```
sub_knn = pd.DataFrame()
sub_knn['PassengerId'] = df_test['PassengerId']
sub_knn['Survived'] = pred_all_knn
#sub_knn.to_csv('knn.csv',index=False)
```