Titanic Classification Project

Data Loading and Importing the necessary libraries

In [1]:
```
# Linear algebra
import numpy as np

# Data manipulation and analysis
import pandas as pd

# Data visualization
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from matplotlib import style

# Algorithms
from sklearn import linear_model
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import Perceptron
from sklearn.linear_model import SGDClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import GridSearchCV
```

Loading the data files

In [2]:
```
train_df = pd.read_csv('train.csv')
test_df = pd.read_csv('test.csv')

train_df['train_test'] = 1
test_df['train_test'] = 0
# test_df['Survived'] = np.NaN
all_data = pd.concat([train_df,test_df])

%matplotlib inline
all_data.columns
```

Out[2]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked', 'train_test'],
      dtype='object')
```

In [3]:
```
train_df.head(15)
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	C
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C
12	13	0	3	Saundercock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.0500	
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.2750	
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.0	0	0	350406	7.8542	

In [4]:

```
test_df.head(15)
```

Out[4]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embark
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	
5	897	3	Svensson, Mr. Johan Cervin	male	14.0	0	0	7538	9.2250	NaN	
6	898	3	Connolly, Miss. Kate	female	30.0	0	0	330972	7.6292	NaN	
7	899	2	Caldwell, Mr. Albert Francis	male	26.0	1	1	248738	29.0000	NaN	
8	900	3	Abrahim, Mrs. Joseph (Sophie Halaut Easu)	female	18.0	0	0	2657	7.2292	NaN	
9	901	3	Davies, Mr. John Samuel	male	21.0	2	0	A/4 48871	24.1500	NaN	
10	902	3	Ilieff, Mr. Ylio	male	NaN	0	0	349220	7.8958	NaN	
11	903	1	Jones, Mr. Charles Cresson	male	46.0	0	0	694	26.0000	NaN	
12	904	1	Snyder, Mrs. John Pillsbury (Nelle Stevenson)	female	23.0	1	0	21228	82.2667	B45	
13	905	2	Howard, Mr. Benjamin	male	63.0	1	0	24065	26.0000	NaN	
14	906	1	Chaffee, Mrs. Herbert Fuller (Carrie Constance...	female	47.0	1	0	W.E.P. 5734	61.1750	E31	

Data understanding using Exploratory Data Analysis (EDA)

In [5]:
```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 13 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 
 12  train_test   891 non-null    int64  
dtypes: float64(2), int64(6), object(5)
memory usage: 90.6+ KB
```

In [6]:
```
train_df.describe()
```

Out[6]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	train_
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000	8
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	

Exploring missing data

In [7]:
```
total = train_df.isnull().sum().sort_values(ascending=False)
percent_1 = train_df.isnull().sum()/train_df.isnull().count()*100
percent_2 = (round(percent_1, 1)).sort_values(ascending=False)
missing_data = pd.concat([total, percent_2], axis=1, keys=['Total', '%'])
missing_data.head(13)
```

Out[7]:

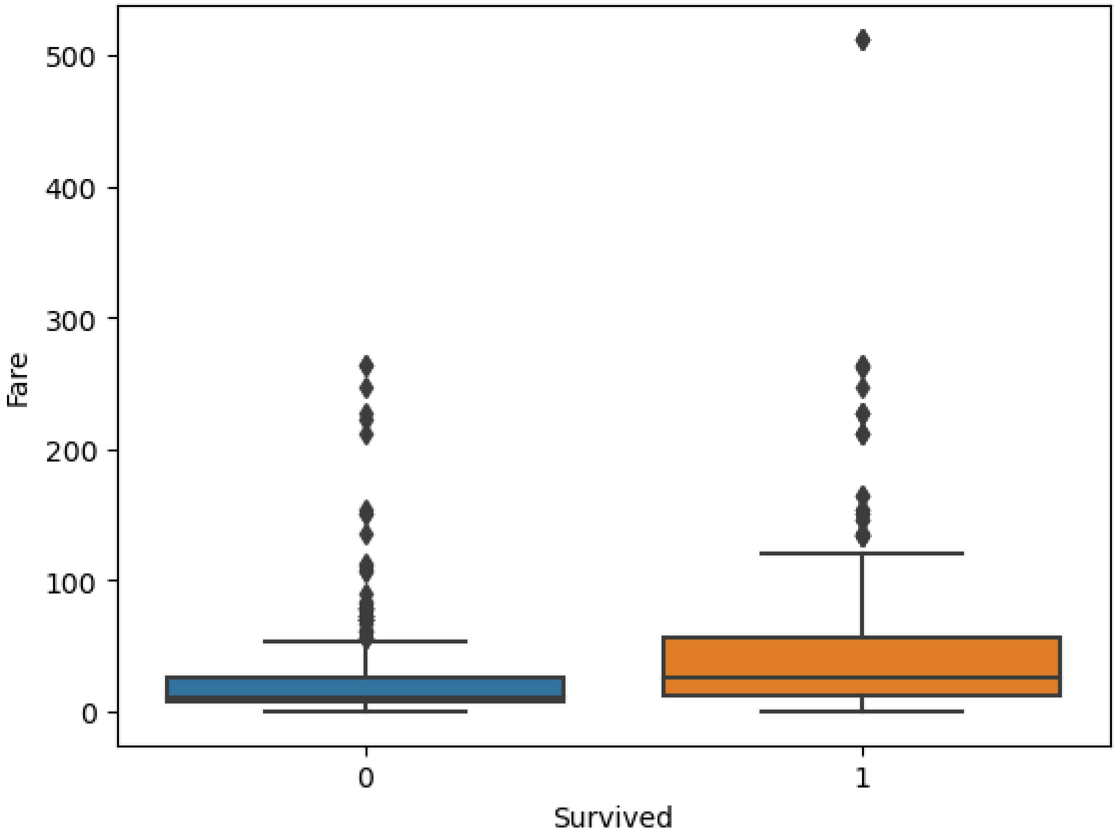
	Total	%
Cabin	687	77.1
Age	177	19.9
Embarked	2	0.2
PassengerId	0	0.0
Survived	0	0.0
Pclass	0	0.0
Name	0	0.0
Sex	0	0.0
SibSp	0	0.0
Parch	0	0.0
Ticket	0	0.0
Fare	0	0.0
train_test	0	0.0

In [8]:
```
train_df.columns.values
```

Out[8]:
```
array(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked', 'train_test'],
      dtype=object)
```

Dealing with the outlier

In [9]:
```
sns.boxplot(x='Survived',y='Fare',data=train_df);
```



In [10]:
```
train_df[train_df['Fare']>300]  #Passengers who paid over 300
```

Out[10]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
258	259	1	1	Ward, Miss. Anna	female	35.0	0	0	PC 17755	512.3292	NaN
679	680	1	1	Cardeza, Mr. Thomas Drake Martinez	male	36.0	0	1	PC 17755	512.3292	B51 B53 B55
737	738	1	1	Lesurer, Mr. Gustave J	male	35.0	0	0	PC 17755	512.3292	B101

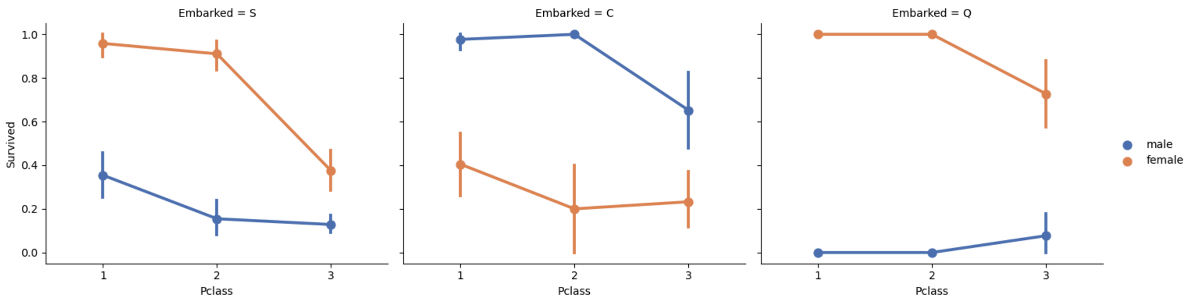
In [11]:
```
train_df[train_df['Name'].str.contains("Capt")] #The Captain went down with the shi
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embar
745	746	0	1	Crosby, Capt. Edward Gifford	male	70.0	1	1	WE/P 5735	71.0	B22	

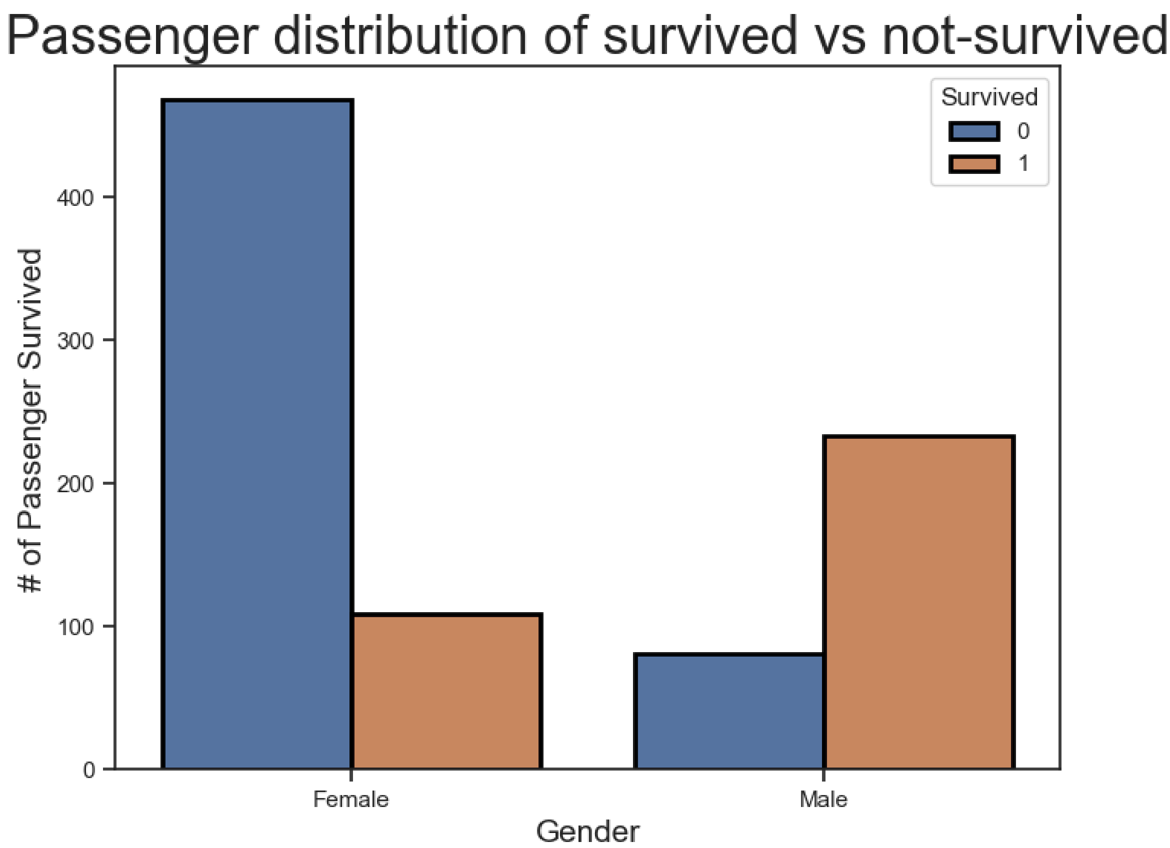
Embarked, Pclass and Sex:

In [12]:
```python
FacetGrid = sns.FacetGrid(train_df, col='Embarked', height=4, aspect=1.2)
FacetGrid.map(sns.pointplot, 'Pclass', 'Survived', 'Sex', errorbar=('ci', 95.0), pa
FacetGrid.add_legend();
```



Distribution of Pclass and Survived

In [18]:
```python
sns.set(style='ticks')
plt.subplots(figsize = (8,6))
ax=sns.countplot(x='Sex', data = train_df, hue='Survived', edgecolor=(0,0,0), linew

# Fixing title, xlabel and ylabel
plt.title('Passenger distribution of survived vs not-survived', fontsize=25)
plt.xlabel('Gender', fontsize=15)
plt.ylabel("# of Passenger Survived", fontsize = 15)
labels = ['Female', 'Male']

# Fixing xticks.
plt.xticks(sorted(train_df.Survived.unique()),labels);
```



In [20]:
```python
train_df.groupby(['Sex']).mean()
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_12664\3313102057.py:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  train_df.groupby(['Sex']).mean()
```

Out[20]:

Sex	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	train_test
female	431.028662	0.742038	2.159236	27.915709	0.694268	0.649682	44.479818	1.0
male	454.147314	0.188908	2.389948	30.726645	0.429809	0.235702	25.523893	1.0

As previously mentioned, women are much more likely to survive than men. 74% of the women survived, while only 18% of men survived.

In [23]:
```
train_df.groupby(['Sex','Pclass']).mean()
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_12664\4204354171.py:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  train_df.groupby(['Sex','Pclass']).mean()
```

Out[23]:

Sex	Pclass	PassengerId	Survived	Age	SibSp	Parch	Fare	train_test
female	**1**	469.212766	0.968085	34.611765	0.553191	0.457447	106.125798	1.0
	2	443.105263	0.921053	28.722973	0.486842	0.605263	21.970121	1.0
	3	399.729167	0.500000	21.750000	0.895833	0.798611	16.118810	1.0
male	**1**	455.729508	0.368852	41.281386	0.311475	0.278689	67.226127	1.0
	2	447.962963	0.157407	30.740707	0.342593	0.222222	19.741782	1.0
	3	455.515850	0.135447	26.507589	0.498559	0.224784	12.661633	1.0

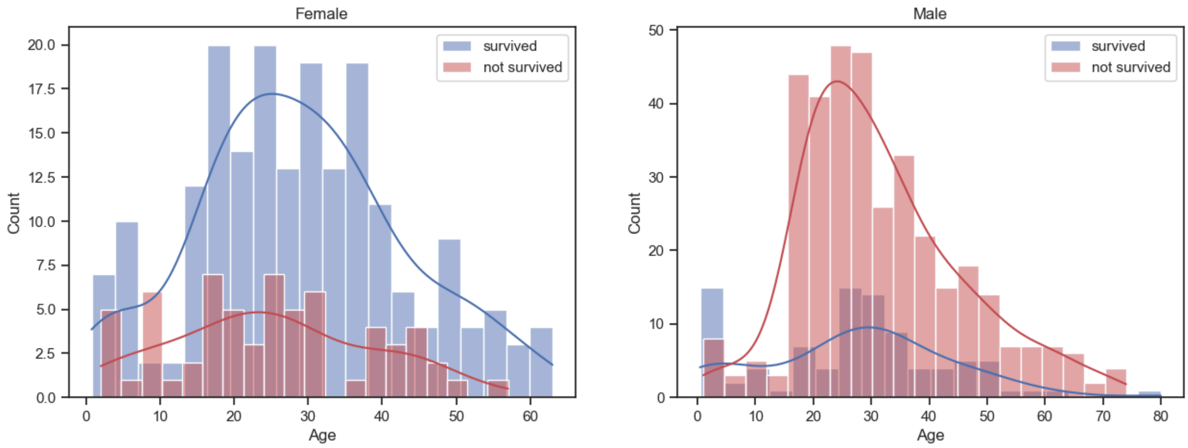
We are grouping passengers based on Sex and Ticket class (Pclass). Notice the difference between survival rates between men and women.

Women are much more likely to survive than men, specially women in the first and second class. It also shows that men in the first class are almost 3-times more likely to survive than men in the third class.

Age and Sex distributions

In [24]:
```
survived = 'survived'
not_survived = 'not survived'

fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(15, 5))

women = train_df[train_df['Sex']=='female']
men = train_df[train_df['Sex']=='male']

# Plot Female Survived vs Not-Survived distribution
ax = sns.histplot(women[women['Survived']==1].Age.dropna(), bins=20, label = surviv
ax = sns.histplot(women[women['Survived']==0].Age.dropna(), bins=20, label = not_su
```

```
ax.legend()
ax.set_title('Female')

# Plot Male Survived vs Not-Survived distribution
ax = sns.histplot(men[men['Survived']==1].Age.dropna(), bins=20, label = survived,
ax = sns.histplot(men[men['Survived']==0].Age.dropna(), bins=20, label = not_surviv
ax.legend()
ax.set_title('Male');
```



We can see that men have a higher probability of survival when they are between 18 and 35 years old. For women, the survival chances are higher between 15 and 40 years old.

For men the probability of survival is very low between the ages of 5 and 18, and after 35, but that isn't true for women. Another thing to note is that infants have a higher probability of survival.

Saving children first

In [25]:

```
train_df[train_df['Age']<18].groupby(['Sex','Pclass']).mean()
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_12664\1113519119.py:1: FutureWarning: The
default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future
version, numeric_only will default to False. Either specify numeric_only or select
only columns which should be valid for the function.
  train_df[train_df['Age']<18].groupby(['Sex','Pclass']).mean()
```

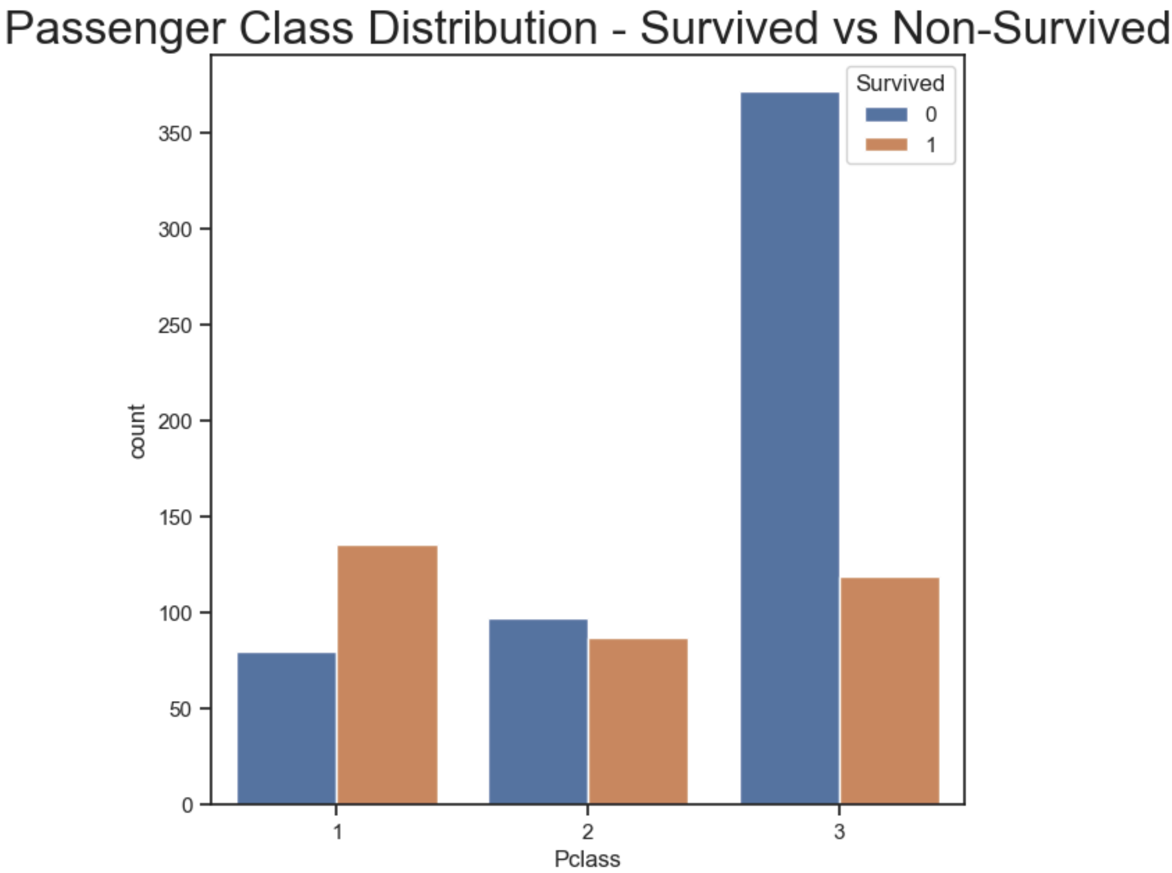
Out[25]:

Sex	Pclass	PassengerId	Survived	Age	SibSp	Parch	Fare	train_test
female	1	525.375000	0.875000	14.125000	0.500000	0.875000	104.083337	1.0
	2	369.250000	1.000000	8.333333	0.583333	1.083333	26.241667	1.0
	3	374.942857	0.542857	8.428571	1.571429	1.057143	18.727977	1.0
male	1	526.500000	1.000000	8.230000	0.500000	2.000000	116.072900	1.0
	2	527.818182	0.818182	4.757273	0.727273	1.000000	25.659473	1.0
	3	437.953488	0.232558	9.963256	2.069767	1.000000	22.752523	1.0

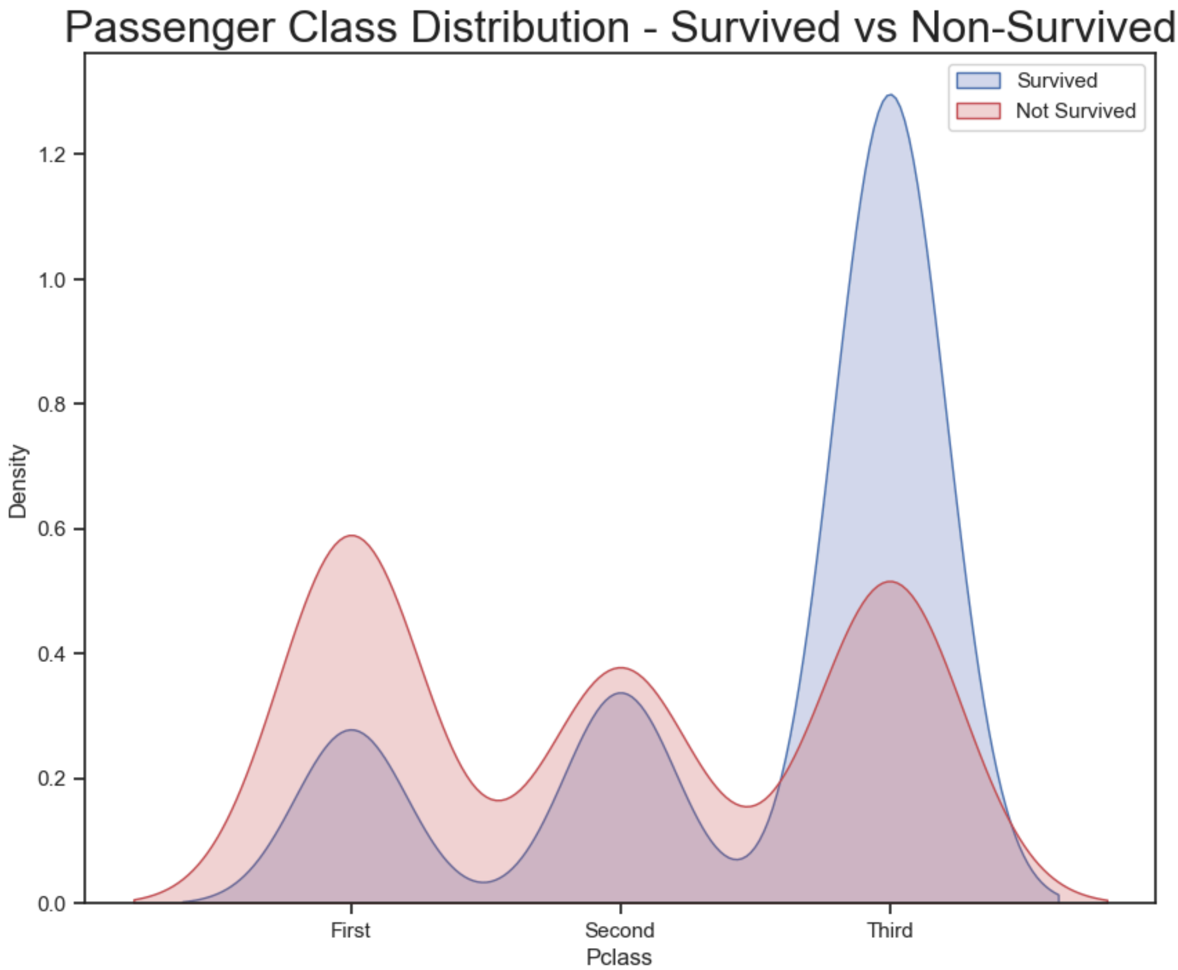
Children below 18 years of age have higher chances of surviving, proven they saved childen firstved

Passenger class distribution; Survived vs Non-Survived

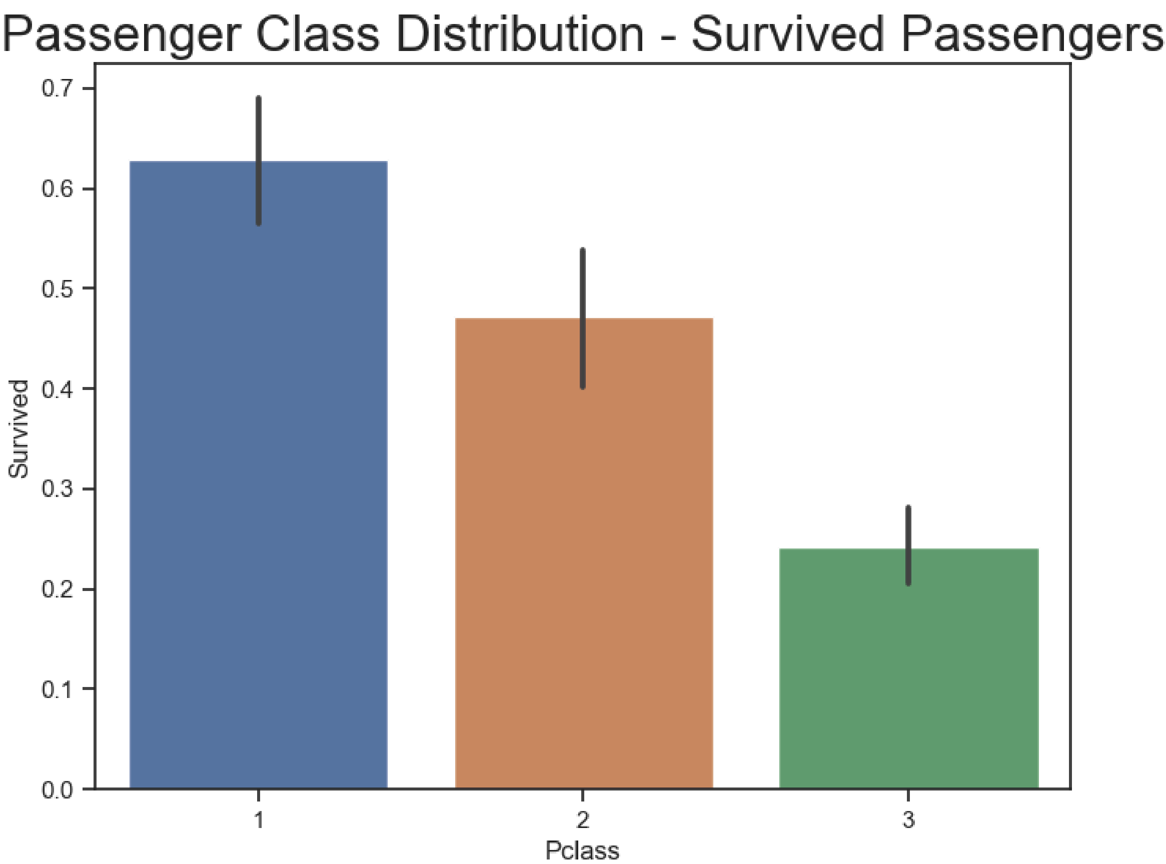
In [28]:
```
plt.subplots(figsize = (7,7))
ax=sns.countplot(x='Pclass',hue='Survived',data=train_df)
plt.title("Passenger Class Distribution - Survived vs Non-Survived", fontsize = 24)
```



In [30]:
```
plt.subplots(figsize=(10,8))
ax=sns.kdeplot(train_df.loc[(train_df['Survived'] == 0),'Pclass'],fill=True,color='
ax.legend()
ax=sns.kdeplot(train_df.loc[(train_df['Survived'] == 1),'Pclass'],fill=True,color='
ax.legend()

plt.title("Passenger Class Distribution - Survived vs Non-Survived", fontsize = 23)
labels = ['First', 'Second', 'Third']
plt.xticks(sorted(train_df.Pclass.unique()),labels);
```



In [31]:
```
plt.subplots(figsize = (8,6))
sns.barplot(x='Pclass', y='Survived', data=train_df);
plt.title("Passenger Class Distribution - Survived Passengers", fontsize = 23);
```



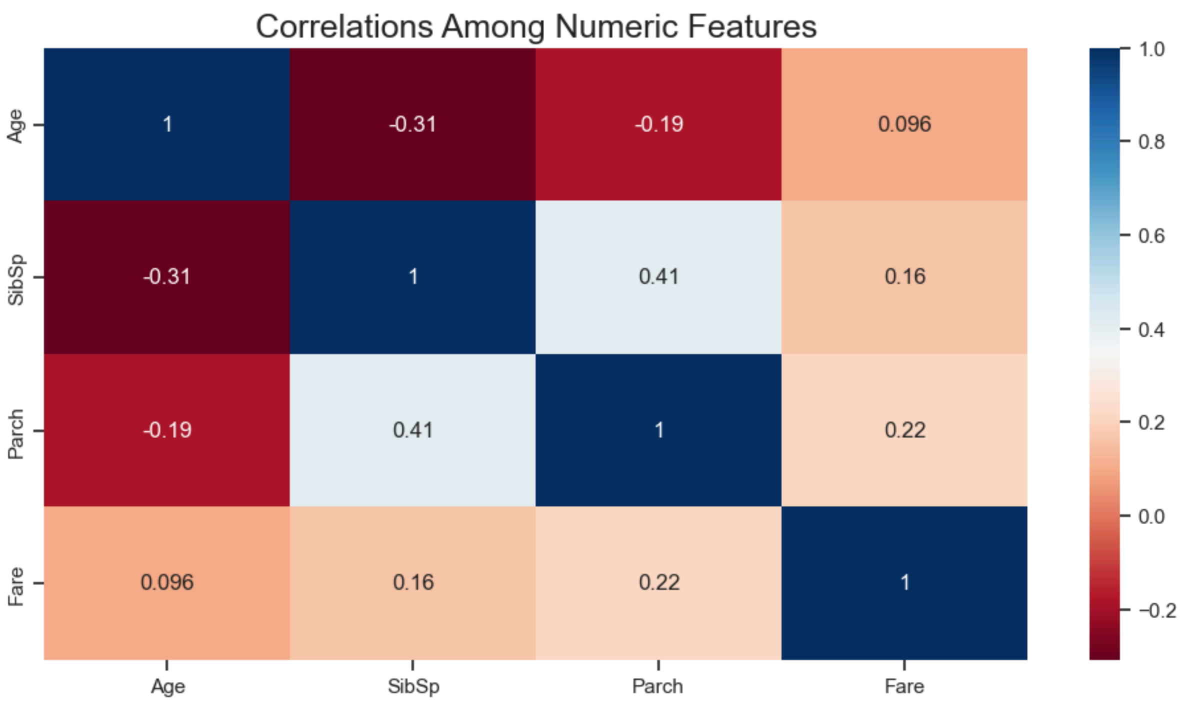
The graphs above clearly shows that economic status (Pclass) played an important role regarding the potential survival of the Titanic passengers. First class passengers had a much higher chance of survival than passengers in the 3rd class. We note that:

1. 63% of the 1st class passengers survived the Titanic wreck
2. 48% of the 2nd class passengers survived
3. Only 24% of the 3rd class passengers survived

Correlation, Matrix and Heatmap

In [34]:

```
# Look at numeric and categorical values separately
df_num = train_df[['Age','SibSp','Parch','Fare']]
df_cat = train_df[['Survived','Pclass','Sex','Ticket','Cabin','Embarked']]
plt.subplots(figsize = (12,6))
sns.heatmap(df_num.corr(), annot=True,cmap="RdBu")
plt.title("Correlations Among Numeric Features", fontsize = 18);
```



We notice from the heatmap above that:

1. Parents and sibling like to travel together (light blue squares)
2. Age has a high negative correlation with number of siblings

In [ ]: