Logistic Regression on Titanic Dataset

```
## Table of Contents

1. Introduction

1.1 Maths Behind Logistic Regression

1.2 Assumptions of Logistic Regression

1.3 Pros/Cons of Logistic Regression

1.4 Applications of Logistic Regression

2. Problem Statement
3. Installing & Importing Libraries

3.1 Installing Libraries

3.2 Upgrading Libraries

3.3 Importing Libraries

4. Data Acquisition & Description

4.1 Data Information

4.2 Data Statistics

5. Data Pre-processing

5.1 Pre-Profiling Report

5.2 Identification & Handling of Missing Data

5.3 Feature Engineering

6. Exploratory Data Analysis

7. Post Data Processing & Feature Selection

7.1 Feature Selection

7.2 Post Profiling Report

7.3 Encoding the Categorical Data

7.4 Data Preparation

8. Model Development & Evaluation

8.1 Logistic Regression - Baseline Model

8.2 Using Trained Model for Prediction

8.3 Logistic Regression - Model Evaluation

```

```
9. Conclusion



```

1. Introduction¶

Logistic regression is a techinque used for solving the classification problem.

Classification is nothing but a problem of identifying to which of a set of categories a new observation belongs, on the basis of training dataset containing observations (or instances) whose categorical membership is known.

For example to predict: Whether an email is spam (1) or not (0) or, Whether the tumor is malignant (1) or not (0)

2. Problem Statement

The Titanic dataset provides observations for each passenger and the survival outcome.

The goal of this case study is to predict survival of passenger travelling in RMS Titanic using Logistic Regression given the features such as passenger class, sex, fair, age, number of siblings/spouse aboard, number of parents/children aboard, and others.

3. Installing & Importing Libraries¶

In [1]:

```
import pandas as pd                                          # Importing for pa
from pandas_profiling import ProfileReport                   # Import Pandas Pr
pd.set_option('display.max_columns', None)                   # Unfolding hidden
pd.set_option('display.max_colwidth', None)                  # Unfolding the ma
pd.set_option('display.max_rows', None)                      # Unfolding hidden
pd.set_option('mode.chained_assignment', None)               # Removing restric
pd.set_option('display.float_format', lambda x: '%.5f' % x)  # To suppress scie
#----------------------------------------------------------------------------------
import numpy as np                                           # Importing packag
#----------------------------------------------------------------------------------
import matplotlib.pyplot as plt                              # Importing pyplot
from matplotlib.pylab import rcParams                        # Backend used for
import seaborn as sns                                        # Importin seaborm
%matplotlib inline
#----------------------------------------------------------------------------------
from sklearn.metrics import accuracy_score                   # For calculating
from sklearn.metrics import precision_score                  # For calculating
from sklearn.metrics import recall_score                     # For calculating
from sklearn.metrics import precision_recall_curve           # For precision an
from sklearn.metrics import confusion_matrix                 # For verifying mo
from sklearn.metrics import f1_score                         # For Checking the
from sklearn.metrics import roc_curve                        # For Roc-Auc metr
#----------------------------------------------------------------------------------
from sklearn.model_selection import train_test_split         # To split the dat
from sklearn.linear_model import LogisticRegression          # To create the Lo
#----------------------------------------------------------------------------------
import warnings                                              # Importing warnin
warnings.filterwarnings("ignore")                            # Warnings will ap
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-1-f272af1c5d4b> in <module>
      1 import pandas as pd
# Importing for panel data analysis
----> 2 from pandas_profiling import ProfileReport
# Import Pandas Profiling (To generate Univariate Analysis)
      3 pd.set_option('display.max_columns', None)
# Unfolding hidden features if the cardinality is high
      4 pd.set_option('display.max_colwidth', None)
# Unfolding the max feature width for better clearity
      5 pd.set_option('display.max_rows', None)
# Unfolding hidden data points if the cardinality is high

ModuleNotFoundError: No module named 'pandas_profiling'
```

In [ ]:

```

```

In [44]:

```
titanic_data = pd.read_csv("https://raw.githubusercontent.com/insaid2018/Term-1/master/
titanic_data.head()
```

Out[44]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
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
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500

In [ ]:

```

```

4.1 Data Information¶

In this section we will see the information about the types of features.

In [45]:

```
titanic_data.shape
```

Out[45]:

```
(891, 12)
```

In [46]:

```
1 titanic_data.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
```

In [47]:

```
1 titanic_data['Age'].isnull().sum()
```

Out[47]:

```
177
```

In [48]:

```
1 titanic_data['Cabin'].isnull().sum()
```

Out[48]:

```
687
```

In [49]:

```
1 titanic_data['Embarked'].isnull().sum()
```

Out[49]:

```
2
```

In [50]:

```
1 titanic_data['Pclass'].isnull().sum()
```

Out[50]:

```
0
```

Observation:

There are null values present in Age, Cabin and Embarked columns.

Each feature seems to have correct data type

If the given data is catogorical then we have to replace by mode, Here embarked column is having catogorical dataso will replaced it with mode

In [51]:

```
1 titanic_data['Embarked'].value_counts()
```

Out[51]:

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

In [52]:

```
1 titanic_data['Embarked'].mode()
```

Out[52]:

```
0    S
dtype: object
```

In [53]:

```
1 titanic_data['Embarked'].mode()[0]
```

Out[53]:

```
'S'
```

In [54]:

```
1 titanic_data.Embarked = titanic.Embarked.fillna(titanic['Embarked'].mode())[0]
```

In [55]:

```
1 titanic_data.Embarked
```

Out[55]:

```
0      S
1      S
2      S
3      S
4      S
      ..
886    S
887    S
888    S
889    S
890    S
Name: Embarked, Length: 891, dtype: object
```

In [58]:

```
1 titanic_data['Embarked'].value_counts()
```

Out[58]:

```
S    891
Name: Embarked, dtype: int64
```

In [57]:

```
titanic_data['Embarked'].value_counts().sum()
```

Out[57]:

```
891
```

In [59]:

```
titanic_data['Age'].isna().sum()
```

Out[59]:

```
177
```

In [60]:

```
titanic_data.describe()
```

Out[60]:

	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

Hear in case of age column mean and median both are near to each other hence will replace by mean or by median

If outlier are there then will replace by median, If there is no ourliers means then go for mean

In [62]:

```
titanic_data.Age = titanic.Age.fillna(titanic.Age.mean())
titanic_data['Age'].isna().sum()
```

Out[62]:

```
0
```

In [63]:

```
titanic_data.describe()
```

Out[63]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	891.000000	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	13.002015	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	22.000000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	29.699118	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	35.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

now all coiumns having same counts 891.00000

In [64]:

```
titanic_data.isna().sum()
```

Out[64]:

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         0
dtype: int64
```

Dealing with missing values

Replacing missing entries of Embarked.with mode values Replacing missing values of Age and Fare with median values. Dropping the column 'Cabin' as it has too many null values and due to the presence of high cardinality it is impossible to impute with anything. The missing value in the data should always be checked before moving on with any experimentation.

In [65]:

```
titanic_data.drop('Cabin',axis= 1,inplace = True)
```

In [66]:

```
titanic_data.isna().sum()
```

Out[66]:

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [67]:

```
titanic_data['SibSp'].value_counts()
```

Out[67]:

```
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: SibSp, dtype: int64
```

In [68]:

```
titanic_data['Parch'].value_counts()
```

Out[68]:

```
0    678
1    118
2     80
5      5
3      5
4      4
6      1
Name: Parch, dtype: int64
```

multicollinearity is observed:

```
1. Fare and pclass
2. Parch and SibSp
```

Feature Engineering.

We are going to create a new column that takes in the values of sibling/Spouse and Parents/Child aboard the RMS Titanic.

We are also adding 1 because we are including the passenger too.

In [69]:

```
titanic_data['FamilySize'] = titanic['SibSp'] + titanic['Parch']+1
```

In [70]:

```
titanic_data.head()
```

Out[70]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	I
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	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [71]:

```
titanic_data['Age'].describe()
```

Out[71]:

```
count    891.000000
mean      29.699118
std       13.002015
min        0.420000
25%       22.000000
50%       29.699118
75%       35.000000
max       80.000000
Name: Age, dtype: float64
```

In [30]:

```
# if Age < 15 child
#  else   keep male as male and female as female
```

In [72]:

```
titanic_data['Genderclass'] = titanic.apply(lambda x: ['child'] if x ['Age'] <15 else 
```

In [73]:

```
1  titanic_data.sample(5)
```

Out[73]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
849	850	1	1	Goldenberg, Mrs. Samuel L (Edwiga Grabowska)	female	29.699118	1	0	17453	89.
836	837	0	3	Pasic, Mr. Jakob	male	21.000000	0	0	315097	8.
523	524	1	1	Hippach, Mrs. Louis Albert (Ida Sophia Fischer)	female	44.000000	0	1	111361	57.
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.000000	0	0	350406	7.
340	341	1	2	Navratil, Master. Edmond Roger	male	2.000000	1	1	230080	26.

if we run sample (5) again and again then will get child in recordes

In [74]:

```
1  titanic_data[titanic . Age < 15].head(2)
```

Out[74]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Emb
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	

In [75]:

```
titanic_data[titanic.Age > 15 ].head(2)
```

Out[75]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Emba
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	

In [77]:

```
titanic_data['Genderclass']
```

Out[77]:

```
0        male
1      female
2      female
3      female
4        male
        ...
886      male
887    female
888    female
889      male
890      male
Name: Genderclass, Length: 891, dtype: object
```

In [82]:

```
titanic_data['Embarked'].value_counts()
```

Out[82]:

```
S    891
Name: Embarked, dtype: int64
```

In [84]:

```
titanic_data.head()
```

Out[84]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	I
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	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [85]:

```
1 titanic_data.head()
```

Out[85]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	I
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	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [86]:

```
1 titanic_data.describe()
```

Out[86]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	F
count	891.000000	891.000000	891.000000	891.000000	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	13.002015	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	22.000000	0.000000	0.000000	7.910400	
50%	446.000000	0.000000	3.000000	29.699118	0.000000	0.000000	14.454200	
75%	668.500000	1.000000	3.000000	35.000000	1.000000	0.000000	31.000000	
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200	

we want all categorical columns from data and for that we have to (-) all numerical columns with non numerical columns

In [87]:

```
titanic_data.columns
```

Out[87]:

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Embarked', 'FamilySize', 'Genderclass'],
      dtype='object')
```

In [88]:

```
numerical_col = titanic_data.describe().columns
numerical_col

```

Out[88]:

```
Index(['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare',
       'FamilySize'],
      dtype='object')
```

now total columns - numerical_col = categorical columns

but we can not substract directly so that we have to convert as set

In [89]:

```
set(titanic_data.columns) - set(numerical_col)


```

Out[89]:

```
{'Embarked', 'Genderclass', 'Name', 'Sex', 'Ticket'}
```

In [90]:

```
categorical_col = set(titanic_data.columns) - set(numerical_col)
categorical_col
```

Out[90]:

```
{'Embarked', 'Genderclass', 'Name', 'Sex', 'Ticket'}
```

In [91]:

```
titanic_data['Embarked'].value_counts()
```

Out[91]:

```
S    891
Name: Embarked, dtype: int64
```

In [92]:

```
titanic_data['Genderclass'].value_counts
```

Out[92]:

```
<bound method IndexOpsMixin.value_counts of 0        male
1      female
2      female
3      female
4        male
        ...
886      male
887    female
888    female
889      male
890      male
Name: Genderclass, Length: 891, dtype: object>
```

Here we donot want sex columns becouse we are allready converted it into Genderclass so will going to drop this column

we need not Name column also, Ticket colume is also not needed

In [116]:

```
titanic_data.head()
```

Out[116]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	I
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	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [119]:

```
titanic_data.Survived.value_counts()
```

Out[119]:

```
0    549
1    342
Name: Survived, dtype: int64
```

Question: What is the frequency and proportion of Survival?

In [132]:

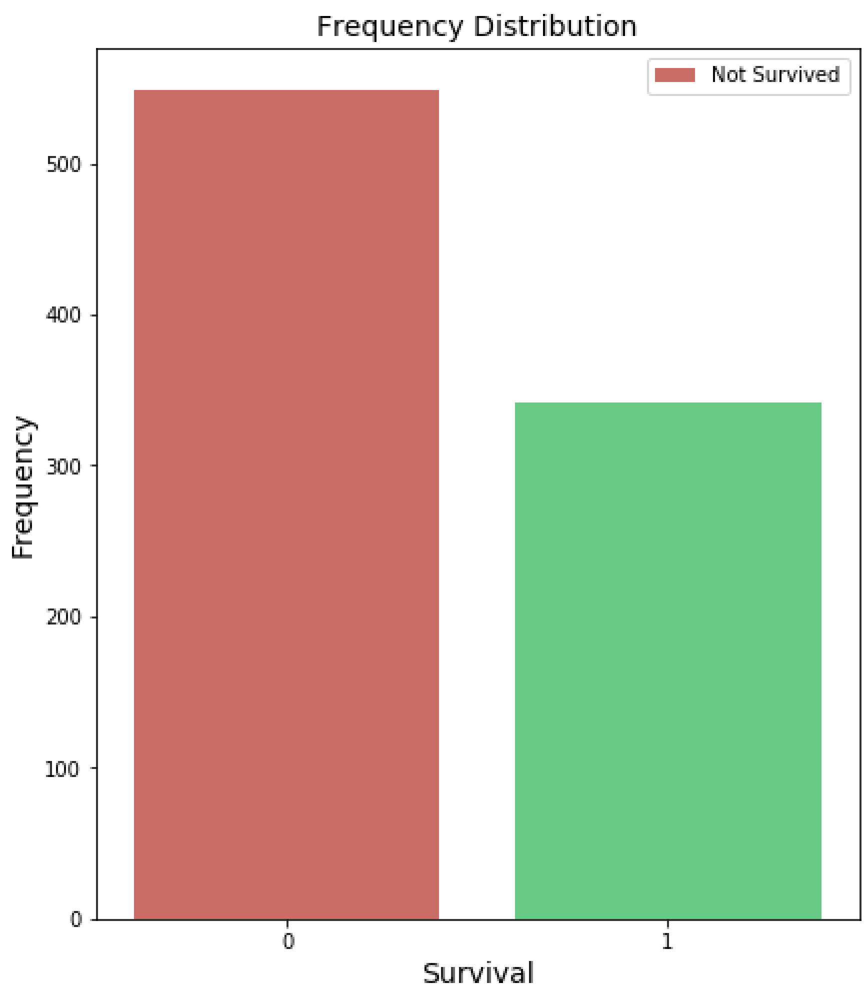
```
import matplotlib . pyplot as plt
import seaborn as sns


fig = plt.figure(figsize = [15, 8])
plt.subplot(1, 2, 1)
sns.countplot(x = 'Survived', data = titanic_data, palette = ['#DB5E56','#56DB7F'])
plt.xlabel(xlabel = 'Survival', size = 14)
plt.ylabel(ylabel = 'Frequency', size = 14)
plt.title(label = 'Frequency Distribution', size = 14)
plt.legend(['Not Survived', 'Survived'])

```

Out[132]:

```
<matplotlib.legend.Legend at 0x1cfb813d108>
```



Observation:

We can observe that there are lesser number of people that Survived.

Additionally, we can observe class imbalance in our data set.

This might pose a big problem while performing model building, but we will see what we can do.

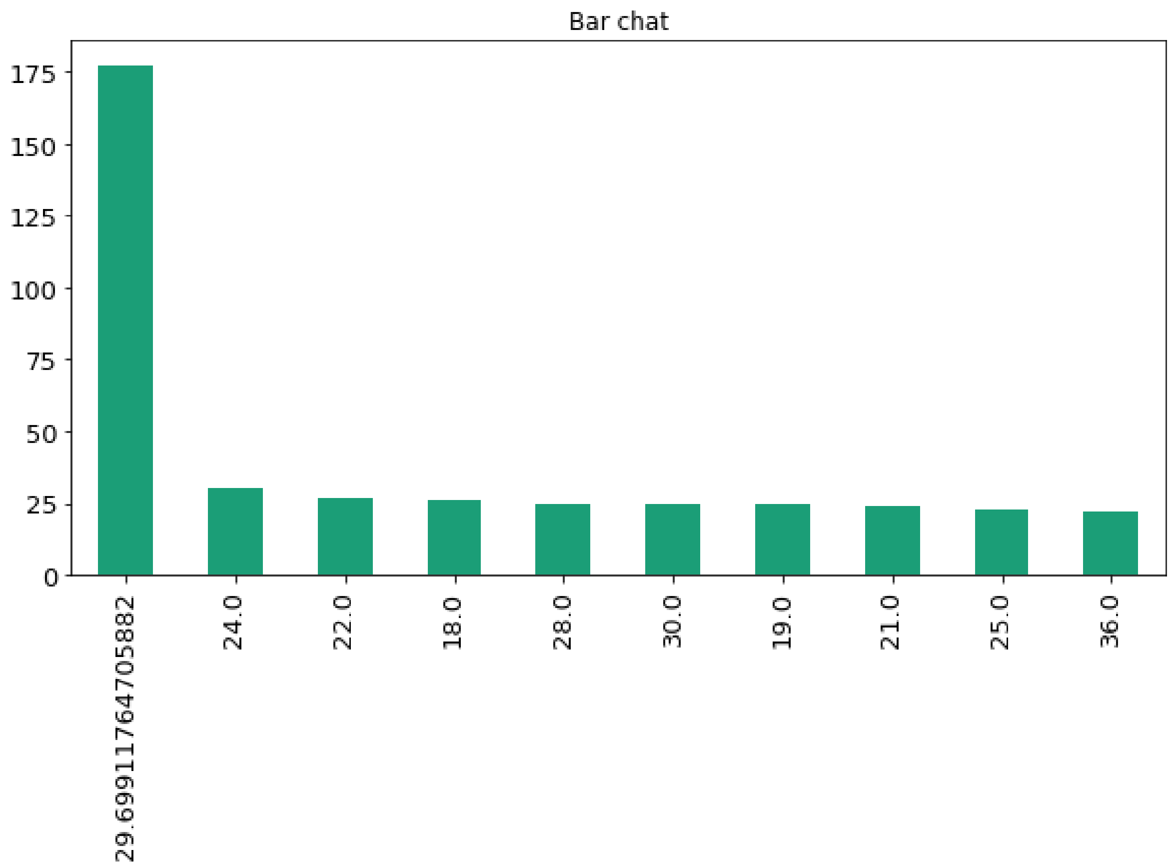
Question: Is there any association between Age and Survival?

In [138]:

```
titanic_data['Age'].value_counts().head(10).plot(kind='bar',figsize=(10,5),y = 'Survive
plt.title('Bar chat')
```

Out[138]:

```
Text(0.5, 1.0, 'Bar chat')
```

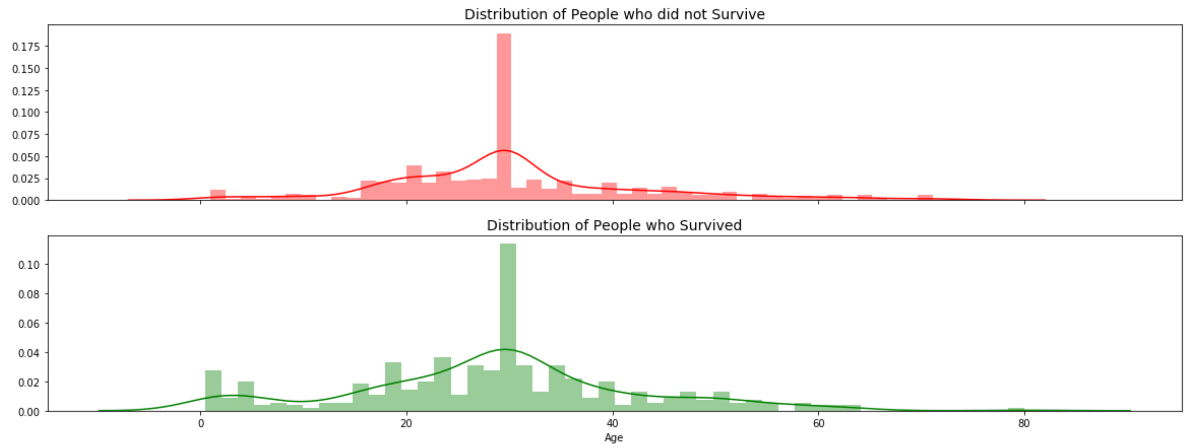


In [139]:

```
# Slicing data with non-survial
Not_Survived = titanic_data['Age'][titanic_data['Survived'] == 0]

# Slicing data with survival
Survived = titanic_data['Age'][titanic_data['Survived'] == 1]

# Plotting the distribution of the sliced data
fig, (ax1, ax2) = plt.subplots(nrows = 2, ncols = 1, sharex = True, figsize = (20, 7))
sns.distplot(a = Not_Survived, bins = 50, ax = ax1, color = 'red')
ax1.set_title(label = 'Distribution of People who did not Survive', size = 14)
ax1.set_xlabel(xlabel = '')
sns.distplot(a = Survived, bins = 50, ax = ax2, color = 'green')
ax2.set_title(label = 'Distribution of People who Survived', size = 14)
plt.show()
```



In [ ]:

```
# directly will logistic regression start
```

In [140]:

```
1 titanic_data.head()
```

Out[140]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	I
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	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [142]:

```
1 titanic_data.drop(['Parch','SibSp','Name','Sex','Ticket'],axis=1 ,inplace = True)
```

In [143]:

```
1 titanic_data.head()
```

Out[143]:

	PassengerId	Survived	Pclass	Age	Fare	FamilySize	Genderclass
0	1	0	3	22.0	7.2500	2	male
1	2	1	1	38.0	71.2833	2	female
2	3	1	3	26.0	7.9250	1	female
3	4	1	1	35.0	53.1000	2	female
4	5	0	3	35.0	8.0500	1	male

In [ ]:

```
1 t
```