Loading the data

In [2]:

```
# importing required libraries
import pandas as pd
```

In [3]:

```
#loading the data
data = pd.read_csv('titanic_train.csv')
data.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Missing Values

In [4]:

```
#missing values in the data
data.isnull().sum()
```

Out[4]:

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

- Age and Cabin have a very high number of missing values
- Embarked has very low number of missing values

Imputing Missing Values Using central tendency

In [5]:

```
# finding mean value
mean_val = data['Age'].mean()
mean_val
```

Out[5]:

```
29.69911764705882
```

In [6]:

```
# making a copy
data_cleaned = data.copy()

#imputing missing values
data_cleaned['Age'] = data['Age'].fillna(value = mean_val)
data_cleaned['Age'].isnull().sum()
```

Out[6]:

```
0
```

In [7]:

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

Out[7]:

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

In [9]:

```
mode_val = data['Embarked'].mode()[0]
mode_val
```

Out[9]:

```
'S'
```

In [10]:

```
data_cleaned['Embarked'] = data['Embarked'].fillna(value = mode_val)
```

In [ ]:

```

```

Dealing with Categorical Variables

In [11]:

```
#Categorical variables in the data
data.dtypes
```

Out[11]:

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [12]:

```
categorical_cols = ['Name', 'Sex', 'Ticket', 'Cabin', 'Embarked']
```

In [13]:

```
#number of unique values
data[categorical_cols].nunique()
```

Out[13]:

```
Name        891
Sex           2
Ticket      681
Cabin       147
Embarked      3
dtype: int64
```

- Can One-hot-Encode Sex and Embarked
- Deal with them differently (extract features)
- Name, Ticket and Cabin (when encoded) will have zeros

One-hot Encoding

In [14]:

```
pd.get_dummies(data['Embarked']).head()
```

Out[14]:

	C	Q	S
0	0	0	1
1	1	0	0
2	0	0	1
3	0	0	1
4	0	0	1

In [15]:

```
data_cleaned = data_cleaned.drop(['Name','Ticket','Cabin'], axis=1)
```

In [16]:

```
data_cleaned = pd.get_dummies(data_cleaned)
data_cleaned.head()
```

Out[16]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embark
0	1	0	3	22.0	1	0	7.2500	0	1	0	0	
1	2	1	1	38.0	1	0	71.2833	1	0	1	0	
2	3	1	3	26.0	0	0	7.9250	1	0	0	0	
3	4	1	1	35.0	1	0	53.1000	1	0	0	0	
4	5	0	3	35.0	0	0	8.0500	0	1	0	0	

- SibSp and Parch hold discrete values
- We can convert them into separate columns as well

Label Encoding

In [17]:

```
data.head()
```

Out[17]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [18]:

```
## map function

data['Embarked'].map({'Q': 0, 'S': 1, 'C':2})
```

Out[18]:

```
0      1.0
1      2.0
2      1.0
3      1.0
4      1.0
      ...
886    1.0
887    1.0
888    1.0
889    2.0
890    0.0
Name: Embarked, Length: 891, dtype: float64
```

In [19]:

```
data['Embarked'] = data['Embarked'].map({'Q': 0, 'S': 1, 'C':2})
data['Embarked'].head()
```

Out[19]:

```
0    1.0
1    2.0
2    1.0
3    1.0
4    1.0
Name: Embarked, dtype: float64
```

In [ ]:

```

```