Deepika
AIML Trainee | AI/ML bootcamp

Data Cleaning

Data Cleaning is the process of identifying and fixing or removing errors and inconsistencies in data to improve its quality. This is a critical step in the data analysis pipeline to ensure that your analysis or model is built on accurate, complete, and reliable data. Data Cleaning is also called **data cleansing** or **scrubbing**

-->Why is Data Cleaning Important?

- Improves **accuracy** of analysis and models
- Ensures **valid results** from data insights
- Eliminates **duplicate, corrupted, or incomplete data**
- Helps maintain **data consistency** across sources

--> Data Cleaning Tasks

1. Handling Missing Data:-Missing data is a common problem in datasets. Strategies to handle missing data include:

- Removing Records: Deleting rows with missing values if they are relatively few and insignificant.
- Imputing Values: Replacing missing values with estimated ones, such as the mean, median, or mode of the dataset.

2. Removing Duplicates:-Duplicates can skew analyses and lead to inaccurate results. Identifying and removing duplicate records ensures that each data point is unique and accurately represented.

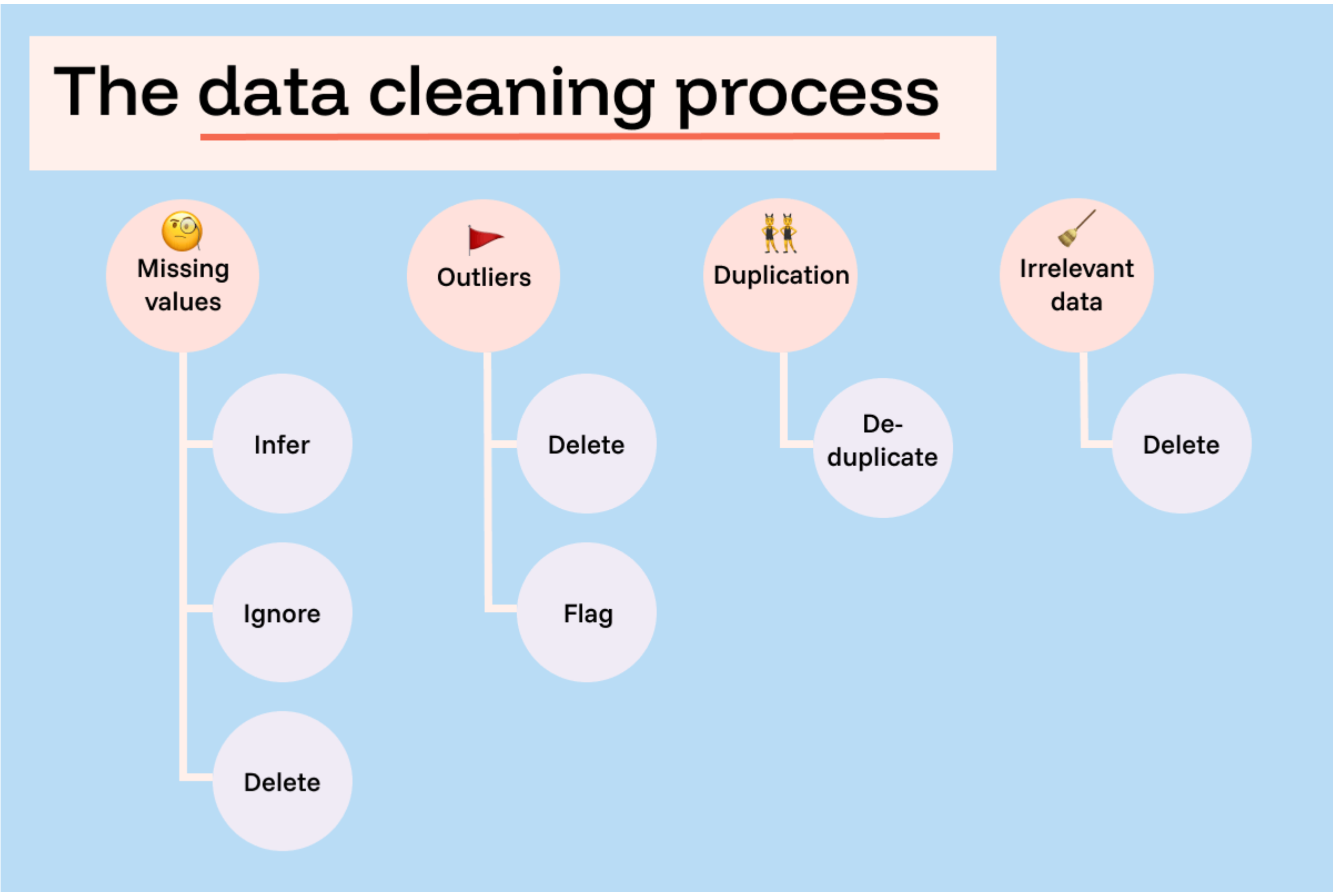
3. Correcting Inaccuracies:-Data entry errors, such as typos or incorrect values, need to be identified and corrected. This can involve cross-referencing with other data sources or using validation rules to ensure data accuracy.

4. Standardizing Data:-Standardization means bringing data to a common format, like:

- Date formats (2025-07-29 vs 29-07-2025)
- Units (kg, lbs)
- Capitalization (INDIA, India, india → India)

-->Challenges in Data Cleaning

- **Volume of Data:** Large datasets can be challenging to clean due to their sheer size. Efficient techniques and tools are necessary to handle big data cleaning tasks.
- **Complexity of Data:** Data from diverse sources may have different structures and formats, making it difficult to clean and integrate.
- **Continuous Process:** Data cleaning is not a one-time task but an ongoing process. As new data is collected, it needs to be continually cleaned and maintained.



Data Cleaning using Titanic DataSet

In [1]:
```python
#import libraries
import pandas as pd
import numpy as np
```

In [2]:
```python
#reading csv file into notebook
df = pd.read_csv('tested.csv')
```

In [3]:
```python
df
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

418 rows × 12 columns

In [4]:
```python
#print  1st to 7 columns
df.head(7)
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
5	897	0	3	Svensson, Mr. Johan Cervin	male	14.0	0	0	7538	9.2250	NaN	S
6	898	1	3	Connolly, Miss. Kate	female	30.0	0	0	330972	7.6292	NaN	Q

In [5]:
```python
#print last 4 column
df.tail(4)
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

In [6]:
```python
# get a summary of the DataFrame, statistics of the numerical columns
df.describe()
```

Out[6]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	0.363636	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.481622	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	0.000000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	0.000000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	1.000000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	1.000000	3.000000	76.000000	8.000000	9.000000	512.329200

In [7]:
```python
# get a random sample of 5 rows from the DataFrame
df.sample(5)
```

Out[7]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
20	912	0	1	Rothschild, Mr. Martin	male	55.00	1	0	PC 17603	59.4000	NaN	C
168	1060	1	1	Cassebeer, Mrs. Henry Arthur Jr (Eleanor Genev...	female	NaN	0	0	17770	27.7208	NaN	C
348	1240	0	2	Giles, Mr. Ralph	male	24.00	0	0	248726	13.5000	NaN	S
307	1199	0	3	Aks, Master. Philip Frank	male	0.83	0	1	392091	9.3500	NaN	S
313	1205	1	3	Carr, Miss. Jeannie	female	37.00	0	0	368364	7.7500	NaN	Q

In [8]:
```python
# check the data types and null values in the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Survived     418 non-null    int64
 2   Pclass       418 non-null    int64
 3   Name         418 non-null    object
 4   Sex          418 non-null    object
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64
 7   Parch        418 non-null    int64
 8   Ticket       418 non-null    object
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object
 11  Embarked     418 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [9]:
```python
#slicing
df.iloc[1:10]
```

Out[9]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
5	897	0	3	Svensson, Mr. Johan Cervin	male	14.0	0	0	7538	9.2250	NaN	S
6	898	1	3	Connolly, Miss. Kate	female	30.0	0	0	330972	7.6292	NaN	Q
7	899	0	2	Caldwell, Mr. Albert Francis	male	26.0	1	1	248738	29.0000	NaN	S
8	900	1	3	Abrahim, Mrs. Joseph (Sophie Halaut Easu)	female	18.0	0	0	2657	7.2292	NaN	C
9	901	0	3	Davies, Mr. John Samuel	male	21.0	2	0	A/4 48871	24.1500	NaN	S

In [10]:
```python
# checking for missing values in the DataFrame
df.isnull().sum()
```

Out[10]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [11]:
```python
# particaluerly checking for missing values in the 'Ticket' columns
df['Ticket'].isnull().sum()
```

Out[11]:
```
np.int64(0)
```

In [12]:
```python
# multiple columns check for missing values
df[['Ticket','Fare', 'Age']].isnull().sum()
```

Out[12]:
```
Ticket     0
Fare       1
Age       86
dtype: int64
```

In [13]:
```python
 # check for duplicate rows in the DataFrame
df.duplicated().sum()
```

Out[13]:
```
np.int64(0)
```

In [14]:
```python
#check shape
df.shape
```

Out[14]:
```
(418, 12)
```

In [15]:
```python
df['Name'].head()
```

Out[15]:
```
0                               Kelly, Mr. James
1                Wilkes, Mrs. James (Ellen Needs)
2                       Myles, Mr. Thomas Francis
3                               Wirz, Mr. Albert
4    Hirvonen, Mrs. Alexander (Helga E Lindqvist)
Name: Name, dtype: object
```

In [16]:
```python
df['Age'].unique()
```

Out[16]:
```
array([34.5 , 47.  , 62.  , 27.  , 22.  , 14.  , 30.  , 26.  , 18.  ,
       21.  ,   nan, 46.  , 23.  , 63.  , 24.  , 35.  , 45.  , 55.  ,
        9.  , 48.  , 50.  , 22.5 , 41.  , 33.  , 18.5 , 25.  , 39.  ,
       60.  , 36.  , 20.  , 28.  , 10.  , 17.  , 32.  , 13.  , 31.  ,
       29.  , 28.5 , 32.5 ,  6.  , 67.  , 49.  ,  2.  , 76.  , 43.  ,
       16.  ,  1.  , 12.  , 42.  , 53.  , 26.5 , 40.  , 61.  , 60.5 ,
        7.  , 15.  , 54.  , 64.  , 37.  , 34.  , 11.5 ,  8.  ,  0.33,
       38.  , 57.  , 40.5 ,  0.92, 19.  , 36.5 ,  0.75,  0.83, 58.  ,
        0.17, 59.  , 14.5 , 44.  ,  5.  , 51.  ,  3.  , 38.5 ])
```

Handle these errors or anomalies

- *Listwise deletion:-* removes entire rows from the dataset if any single value is missing or erroneous in that row.
- *Pairwise Deletion:-*Pairwise deletion keeps all available data for each analysis, only excluding missing values when necessary.

In [17]:
```python
 # drop rows with any missing values
df_listwise_deletion = df.dropna()
```

In [18]:
```python
# drop rows with missing values in 'Ticket' and 'Age' columns
df_pairwise_deletion = df.dropna(subset=['Fare'])
```

In [19]:
```python
#df.isnull().sum()
df_pairwise_deletion.isnull().sum()
```

Out[19]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          326
Embarked         0
dtype: int64
```

In [20]:
```python
df_listwise_deletion.isnull().sum()
```

Out[20]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
```

IMPUTATION - impute - filling the values

In [21]:
```python
df.isnull().sum()
```

Out[21]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [22]:
```python
df
```

Out[22]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

418 rows × 12 columns

In [23]:
```python
# calculate the mean of the 'Quantity' column
mean_quantity=df['Survived'].mean()
mean_quantity
```

Out[23]:
```
np.float64(0.36363636363636365)
```

In [24]:
```python
 # create a copy of the DataFrame to avoid modifying the original DataFrame
df1 = df.copy()
```

In [25]:
```python
# fill missing values in 'Quantity' with the mean value

df1['Fare'].fillna(mean_quantity)
```

Out[25]:
```
0        7.8292
1        7.0000
2        9.6875
3        8.6625
4       12.2875
         ...
413      8.0500
414    108.9000
415      7.2500
416      8.0500
417     22.3583
Name: Fare, Length: 418, dtype: float64
```

In [26]:
```python
mean_quantity//1
```

Out[26]:
```
np.float64(0.0)
```

In [27]:
```python
# round the mean value to 0 decimal places
round(mean_quantity)
```

Out[27]:
```
0
```

In [28]:
```python
 # fill missing values in 'Quantity' with the rounded mean value
df1['Cabin'].fillna(round(mean_quantity), inplace=True)
```

```
C:\Users\Deepika\AppData\Local\Temp\ipykernel_15300\1054803618.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.


  df1['Cabin'].fillna(round(mean_quantity), inplace=True)
```

In [29]:
```python
df1
```

Out[29]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	0	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	0	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	0	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	0	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	0	S
...	...	...	...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	0	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	0	S
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	0	S
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	0	C

418 rows × 12 columns

In [30]:
```python
# calculate the median of the 'PassengerId' column
median_customer_id = df['PassengerId'].median()
median_customer_id
```

Out[30]:
```
np.float64(1100.5)
```

In [31]:
```python
# calculate the mode of the 'PassengerId' column
mode_customer_id = df['PassengerId'].mode()[0]
mode_customer_id

# [] used here to get the first value of the mode series
# [1] next mode
```

Out[31]:
```
np.int64(892)
```

______________________________END______________________________

THANK YOU!