TechnoHacks EduTech Internship ¶

Data Cleaning : Cleaning the dataset by removing missing values and outliers.

- Datasets : https://www.kaggle.com/c/titanic/data (https://www.kaggle.com/c/titanic/data)

In [5]:
```
import numpy as np
import pandas as pd
```

Load DataFrame

In [3]:
```
df = pd.read_csv('train.csv')
```

Getting comprehensive information about the dataset

In [4]:
```
#displaying first few rows of the DataFrame
df.head()
```

Out[4]:

	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 [5]:
```
#displaying last few rows of the DataFrame
df.tail()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	

In [6]:
```
#getting information about the DataFrame
df.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 [4]:
```
#generating descriptive statistics of a DataFrames's numerical columns
df.describe()
```

Out[4]:

	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

In [7]:
```
#obtaining a list of columns name in a DataFrame
df.columns
```

Out[7]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [8]:
```
#obtaining the data types of each column in a DataFrame
df.dtypes
```

Out[8]:
```
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 [6]:
```
#getting the dimensions of a DataFrame
df.shape
```

Out[6]:
```
(891, 12)
```

In [10]:
```
#getting the total number of elements (the count of all the cells) in the D
df.size
```

Out[10]:
```
10692
```

Checking for missing values in the dataset

In [11]:
```
#Using the 'isnull()' function to identify missing values
df.isnull()
```

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	False	False	False	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True
...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True
887	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True
889	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True

891 rows × 12 columns

In [12]:
```
#getting the number of null values in a DataFrame using the isnull().sum() _
df.isnull().sum()
```

Out[12]:
```
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
```

In [13]:
```
#Using the isnull() function to identify missing values in the specified co
df[['Age','Cabin','Embarked']].isnull()
```

Out[13]:

	Age	Cabin	Embarked
0	False	True	False
1	False	False	False
2	False	True	False
3	False	False	False
4	False	True	False
...	...	...	...
886	False	True	False
887	False	False	False
888	True	True	False
889	False	False	False
890	False	True	False

891 rows × 3 columns

In [8]:
```
#Using the isnull().sum() function to identify the number of missing values
df[['Age','Cabin','Embarked']].isnull().sum()
```

Out[8]:
```
Age         177
Cabin       687
Embarked      2
dtype: int64
```

The Age, Cabin, Embarked columns contains the mssing values. So, let's clean the missing values in specified columns. Using the parameter 'inplace = True', to modifiy the original DataFrame.

For "Age" : Filling missing age values with the mean age of the dataset.

In [9]:
```
#getting the mean of the Age column
mean_age = df['Age'].mean()
```

In [10]:
```
#using the fillna() method the fill the missing values with the mean Age
df['Age'].fillna(mean_age, inplace = True)
```

For "Cabin" : Since the "Cabin" column has many missing values, so dropping the entire column.

In [11]:
```
#dropping the "Cabin" column
df.drop('Cabin', axis = 1, inplace = True)
```

For "Embarked" : Filling missing values with the (mode) most common value in the "Embarked" column.

In [12]:
```
#finding most common value in the 'Embarked' columns
mode_embarked = df['Embarked'].mode()[0]
```

In [13]:
```
#using the fillna() method the fill the missing values with the mode of emb
df['Embarked'].fillna(mode_embarked, inplace = True)
```

Checking if there are any remaining missing values in the DataFrame.

In [14]:
```
missing_values_after_cleaning = df.isnull().sum()
print(missing_values_after_cleaning)
```

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

Part II - Cleaning Outliers in a Dataset

In [21]:
```
df.describe()
```

Out[21]:

	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

Calculating IQR for 'Age' column

In [22]:
```
Q1_age = df.Age.quantile(0.25)
Q3_age = df.Age.quantile(0.75)

IQR_age = Q3_age - Q1_age
print(IQR_age)
```

```
13.0
```

Identifying data points below Q1 - 1.5 * IQR and Q3 + 1.5 * IQR

In [23]:
```
lower_limit_age = Q1_age - 1.5 * IQR_age
upper_limit_age = Q3_age + 1.5 * IQR_age
lower_limit_age, upper_limit_age
```

Out[23]:
```
(2.5, 54.5)
```

Detecting outliers

In [24]:
```
outliers_age = df[(df.Age < lower_limit_age) | (df.Age > upper_limit_age)]
```

Removing outliers from original DataFrame

In [25]:
```
df.drop(outliers_age.index, inplace = True)
```

In [27]:
```
#saving the cleaned data to a new 'cleaned_data' file
cleaned_data = df.to_csv('cleaned_data.csv', index=False)
```