Titanic Dataset

Missing Values- Feature Engineering- Day 1

Missing values are a common issue in real-world datasets. Handling missing data is critical because it can significantly affect the performance of machine learning models. Feature engineering is the process of transforming raw data into a format that can be effectively used by algorithms, and dealing with missing values is an important part of that process. There are various strategies to handle missing data, and the right approach depends on the type of data (numerical or categorical), the context, and the overall dataset size and characteristics.

Lifecycle of a Data Science Projects

Data Collection Statergy---from company side,3rd party APi's,Surveys,Surveys
Feature Engineering---Handling Missing Values
Why are their Missing values?? Survey--Depression Survey

1.They hesitate to put down the information
2.Survey informations are not that valid
3.Men--salary
4.Women---age
5.People may have died----NAN

Types of Missing Data

1.MCAR (Missing Completely at Random)
2.MAR (Missing at Random)
3.MNAR (Missing Not at Random)

1.MCAR (Missing Completely at Random)

The missingness of data is independent of both observed and unobserved data. . When data is MCAR, there is absolutely no relationship between the data missing and any other values, observed or missing, within the dataset. In other words, those missing data points are a random subset of the data. There is nothing systematic going on that makes some data more likely to be missing than other.

For example, survey respondents might accidentally skip a question, and this is unrelated to their characteristics or responses.

In [42]:
```
import pandas as pd   # first import pandas to work on data
```

In [43]:
```
df=pd.read_csv('titanic.csv') # read data
```

In [44]:
```
df.head() # it will show top 5 record of data
```

Out[44]:

	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 [45]:
```
df.isnull().sum() # to find null values
```

Out[45]:
```
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 [46]:
```
df[df['Embarked'].isnull()]
```

Out[46]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
61	62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	NaN
829	830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	NaN

2.MNAR (Missing Not at Random)

There is absolutely some relationship between the data missing and any other values, observed or missing, within the dataset.The missingness is related to the value of the missing data itself. For instance, people with very high income might be less likely to report their income, leading to a missing data pattern that is dependent on the value of the missing attribute.

In [47]:
```
import numpy as np
df['cabin_null']=np.where(df['Cabin'].isnull(),1,0)

##find the percentage of null values
df['cabin_null'].mean()
```

Out[47]:
```
0.7710437710437711
```

In [48]:
```
df[df.columns]
```

Out[48]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	cabin_null
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	0
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S	0
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S	1
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C	0
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q	1

891 rows × 13 columns

In [49]:
```
df.groupby(['Survived'])['cabin_null'].mean()
```

Out[49]:
```
Survived
0    0.876138
1    0.602339
Name: cabin_null, dtype: float64
```

3.MAR (Missing at Random)

The missingness is related to observed data but not to the missing data itself. For example, older people may be less likely to report their income, but this can be explained by age, which is observed in the dataset.

```
Men---hide their salary
Women---hide their age
```

All the techniques of handling ,missing values

In [ ]:
```
### All the techniques of handling ,missing values
1. Mean/ Median/Mode replacement
2. Random Sample Imputation
3. Capturing NAN values with a new feature
4. End of Distribution imputation
5. Arbitrary imputation
6. Frequent categories imputation
```

1. Mean/ Median/Mode replacement

When should we apply?
Mean/median imputation has the assumption that the data are missing completely at random(MCAR). We solve this by replacing the NAN with the most frequent occurance of the variables

Mean Imputation: Replace missing values with the mean of the non-missing values. This works well if the data is approximately normally distributed.

Median Imputation: Replace missing values with the median of the non-missing values. This is more robust than the mean, especially for skewed distributions, as it is less sensitive to outliers

Mode Imputation (for Categorical Data):

Replace missing values with the most frequent (mode) category. This is a simple approach but may not be suitable if the mode is not representative of the missing data.

In [ ]:
```
```

In [51]:
```
df=pd.read_csv('titanic.csv',usecols=['Age','Fare','Survived']) # i have take these rows to show how to handle missing
df.head()                                                       #values by mean/mode
```

Out[51]:

	Survived	Age	Fare
0	0	22.0	7.2500
1	1	38.0	71.2833
2	1	26.0	7.9250
3	1	35.0	53.1000
4	0	35.0	8.0500

In [52]:
```
## Lets go and see the percentage of missing values
df.isnull().mean()
```

Out[52]:
```
Survived    0.000000
Age         0.198653
Fare        0.000000
dtype: float64
```

In [53]:
```
def impute_nan(df,variable,median):
    df[variable+"_median"]=df[variable].fillna(median)
```

In [54]:
```
median=df.Age.median()
median
```

Out[54]:
```
28.0
```

In [55]:
```
impute_nan(df,'Age',median)
df.head()
```

Out[55]:

	Survived	Age	Fare	Age_median
0	0	22.0	7.2500	22.0
1	1	38.0	71.2833	38.0
2	1	26.0	7.9250	26.0
3	1	35.0	53.1000	35.0
4	0	35.0	8.0500	35.0

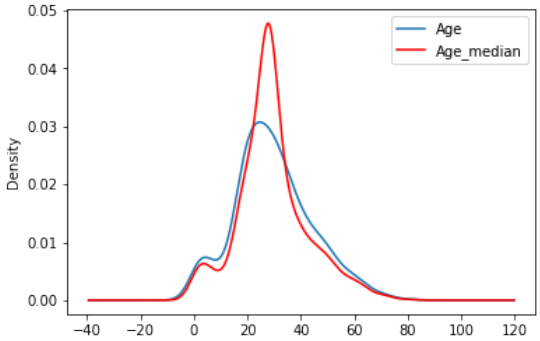
In [56]:
```
print(df['Age'].std())
print(df['Age_median'].std())
```

```
14.526497332334044
13.019696550973194
```

In [57]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [58]:
```
fig = plt.figure()
ax = fig.add_subplot(111)
df['Age'].plot(kind='kde', ax=ax)
df.Age_median.plot(kind='kde', ax=ax, color='red')
lines, labels = ax.get_legend_handles_labels()
ax.legend(lines, labels, loc='best')
```

Out[58]:
```
<matplotlib.legend.Legend at 0x126e56c8100>
```



Advantages And Disadvantages of Mean/Median Imputation

Advantages Easy to implement(Robust to outliers) Faster way to obtain the complete dataset
Disadvantages Change or Distortion in the original variance Impacts Correlation

In [ ]:

In [ ]:

In [ ]: