EDA and Feature Engineering : Various different methods of performing EDA and Feature engineering Treatment

Pre-Processing or Feature Engineering various methods :

Step 1 : Missing Value Treatment

Step 2 : Outlier Handle

Step 3 : Scale the data

Step 4 : Transformation

Step 5 : Encoding

Step 6 : Handle imbalanced data

Step 7 : Feature Selection

Step 8 : Dimensionality reduction (PCA, LOA)

Step 9 : Duplicate value

Step 10 : Split/Merge/Drop/Add

Missing Value various Treatment:

Step 1 : Fill with random numbers

Step 2 : Forward /backward filling

Step 3 : Statistical approach (Mean/median and Mode)

Step 4 : With the help of end of distribution , fill the missing values

Step 5 : Drop the row

Step 6 : Impute with KNN (KNN-Imputer)

Step 7 : ML Algorithm for missing value

Step 8 : Build own ML Model to predict missing values

Outlier Treatments:

Step 1 : Detect the outlier using Z-score, IQR range, Box Plot, Scatter Plot, Violin Plot

Step 2 : After detection of Outlier, we can **Drop/Fill with median/Replace/trimming**

Transformation of the Data various methods:

Step 1 : Box cox transformation

Step 2 : Power Transformation

Step 3 : Log

Step 4 : Square

Step 5 : Cube

Scaling of the Data various methods:

Step 1 : Standardization

Step 2 : Min Max Scaler

Step 3 : Unit Scaling

Encoding various methods:

Step 1 : One hot encoding

Step 2 : Label Encoding

Step 3 : Binary Coding

Step 4 : Target guided encoding

Step 5 : Hash Encoding

Imbalanced dataset Treatment various methods :

Inside the column if the class ration is mismatching , it is called as "Imbalanced Data"

Step 1 : Under Sampling

Step 2 : Over Sampling

Step 3 : Cluster based over sampling

How to find the best model accuracy various methods:

Step 1 : To increase the Accuracy , we need to change the preprocessing technique and use different method or steps from the above

Step 2 : We need to use each and every preprocessing steps and find the best accuracy

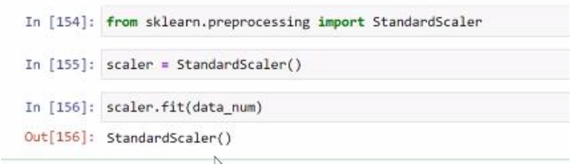
FAQ's

How do we transform the data?

Step 1 : Import numpy
Step 2 : np.log(df)
Step 3 : sns.distplot(df)

How to do scaling of the data simple code

SK Learn library , will learn about it next week class



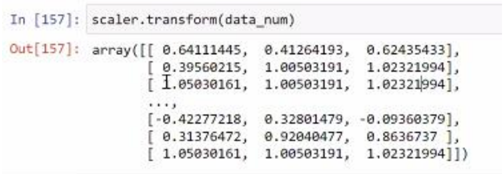
```
In [154]: from sklearn.preprocessing import StandardScaler

In [155]: scaler = StandardScaler()

In [156]: scaler.fit(data_num)

Out[156]: StandardScaler()
```

2.



```
In [157]: scaler.transform(data_num)

Out[157]: array([[ 0.64111445,  0.41264193,  0.62435433],
                 [ 0.39560215,  1.00503191,  1.02321994],
                 [ 1.05030161,  1.00503191,  1.02321994],
                 ...,
                 [-0.42277218,  0.32801479, -0.09360379],
                 [ 0.31376472,  0.92040477,  0.8636737 ],
                 [ 1.05030161,  1.00503191,  1.02321994]])
```

How do we perform imbalance dataset using code

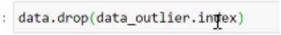
How do we perform encoding with code

Can you explain the outlier function code step by step once again

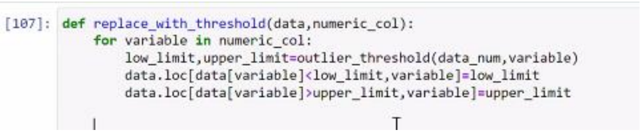
Step 1 : Any values beyond upper limit and any values beyond lower limit are outlier
Step 2 : We are replacing the outliers using lower limit value and higher limit value in the below code

(Or)

Step 3 : We can also drop the outliers from the dataset completely as per the below code



```
: data.drop(data_outlier.index)
```



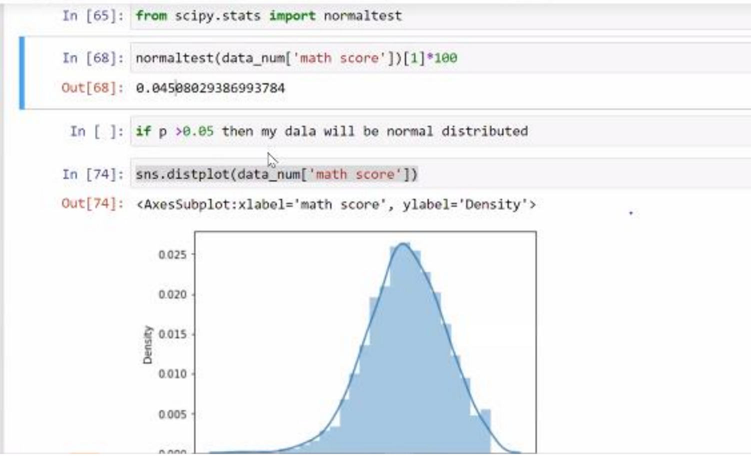
```
[107]: def replace_with_threshold(data,numeric_col):
           for variable in numeric_col:
               low_limit,upper_limit=outlier_threshold(data_num,variable)
               data.loc[data[variable]<low_limit,variable]=low_limit
               data.loc[data[variable]>upper_limit,variable]=upper_limit
```

Please explain that list transformation code where we derived the numerical column and categorical column separately. Can we use the same code whenever we want to derive the numerical and categorical columns for any type of dataset complexities

How do we calculate the P-Value for different features?

Step 1 : Import the scipy.stats import normaltest library
Step 2 : Apply normaltest(df['math score'])[0]*100100
Step 3 : If the P-value is >0.05 then the data will be normally distributed

Step 4 : If the P-value is <0.05 then the data will be not normally distributed as per the below code



```
In [65]: from scipy.stats import normaltest

In [68]: normaltest(data_num['math score'])[1]*100
Out[68]: 0.04508029386993784

In [ ]: if p >0.05 then my dala will be normal distributed

In [74]: sns.distplot(data_num['math score'])
Out[74]: <AxesSubplot:xlabel='math score', ylabel='Density'>
```

EDA and Preprocessing Task :

Step 1 : Pick any one dataset

Step 2 : Read the dataset

Step 3 : Perform Complete EDA

Step 4 : Perform missing value (All Steps or Methods for missing values)

Step 5 : Perform outlier (Every method to handle outlier minimum 4 to 5 (separate ipynb file)

Step 6 : Perform encoding (All methods)

Step 7 : Perform scaling (All methods)

Step 8 : Perform feature selection

Step 9 : GRAPH Analysis : Univariate/Bivariate/multivariate Analysis

Step 10 : Observations

Step 11 : Single folder

Step 12 : Share in GitHub

Step 13 : All the 10 dataset , perform EDA and preprocessing and save it on Github compulsory

Step 14 : We learn to build our own automation package just like pandas profiling, autoviz