

Libraries important- Numpy, Pandas, Seaborn, Matplotlib

**Feature Engineering:**

1. Handling missing values:

* Many machine learning algorithms fail if the dataset contains missing values. However, algorithms like K-nearest and Naive Bayes support data with missing values.
* Can lead to lack of precision in the statistical analysis.
* Finding missing values- train\_df.isnull().sum().sum() -> total number in entire dataset
* Ways to handle it: deleting it or replacing them.
* Delete the row with missing values- df.dropna(axis=0)
* Delete the column with missing values- df.drop(['Dependents'],axis=1)
* Filling missing values- df['Dependents'].fillna(0).
* Filling missing value with mean- if outliers are present , it should be handled first- train\_df['LoanAmount'].fillna(train\_df['LoanAmount'].mean())
* Filling with mode value- done with categorical data train\_df['Gender'].fillna(train\_df['Gender'].mode()[0])
* Filling with median value- done in case of outliers- train\_df['Loan\_Amount\_Term'].fillna(train\_df['Loan\_Amount\_Term'].median())
* Replacing with previous value- test.fillna(method=‘ffill')
* Replacing with next value- test.fillna(method=‘bfill')
* Replacing categorical data with most common data by using sklearn-

from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='most\_frequent')

imputer.fit\_transform(X)

1. **Data exploratory methods:**

* Sns.distplot(train[‘age’)
* Sns.countplot(x=’siblings’,data=)- shows the different count of a certain column/category
* Train[‘age’].hist(bins=30,color=’red’)
* Sns.bloxplot(x= ,y= ,data=train)
* Sns.heatmap(train.isnull(), yticklabels=False)

1. Convert categorical data to numerical data-

* Using label encoder-

from sklearn import preprocessing

label\_encoder = preprocessing.LabelEncoder()

df[‘’body\_style”] = label\_encoder.fir\_transform(data[‘body\_style’)

* Convert categorical to numerical value- df[‘sex’].get\_dummies
* Using onehot encoder- if there are k values for categorical data then create k separate features having values 0 or 1.

From sklearn.preprocessing, import OneHotEncoder.

Onehotencoder = OneHotEncoder()

X = onehotencoder.fit\_transform(df)

* Find and replace method
* Label encoding-

obj\_df["body\_style\_cat"] = obj\_df["body\_style"].cat.codes

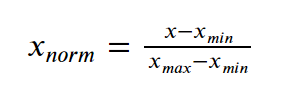
##assigns the code value for each category

1. Detecting outliers:

* Detecting outliers with z-scores: zscore=  (Xi-mean)/std.
* using the Inter Quantile Range(IQR)
* visually summarize the distribution of a variable is the box plot

1. Normalisation and standardization:

* Normalisation formula:



* from sklearn import preprocessing

preprocessing.normalize([x\_array])

* ##normalising the value between 0-2

from sklearn import preprocessing

scaler = preprocessing.MinMaxScaler(feature\_range=(0, 2))

* Standardisation:

from sklearn.preprocessing import StandardScaler

scale.fit\_transform(X\_data)

Feature selection:

1. Pearson’s correlation: summarize the strength of the linear relationship between two data samples. It is the normalization of the covariance between the two variables to give an interpretable score. Code:

from scipy.stats import pearsonr

pearsonr(data1, data2)

1. Spearman’s coefficient: scipy.stats.spearmanr(x, y)
2. In pandas- x.corr(y, method='spearman')

**Points to remember:**

* To find sum of null values in each col:

col for col in df.columns if in df[col].isnull().sum()>1

* To find numerical variables in dataframe:

[col for col in df.columns if df[col].dtypes!=0]

* To find categorical features in dataframe:

[col for col in df.columns if df[col].dtypes==0]

* Grouby use: df.groupby(feature)[‘col values needed’].sum()

df.groupby(year)[‘no. of people dies’].sum()

* Decision trees and tree ensembles can be used with structured data when features are given. Can be used for categorical or numerical data, for classification or regression both.
* Neural networks can be implemented on structured, un- structured or mixed.
* In random forests, random sampling with replacement is done and random set of features are taken to build each tree so that each tree is not identical to each other.