PROJECT REPORT

ON

ONLINE CARBON RATE OPTIMISATION

SUBMITTED BY:-

NAME: SHARFUL HAQUE

REFERENCE NUMBER: VT20193943

NAME OF INSTITUTE: SRM IST-NCR

NAME & LOCATION OF COMPANY: TATA STEEL INDIA PVT LTD, JAMSHEDPUR

PROJECT GUIDE DETAILS:

NAME:RITESH KARN

DEPARTMENT:1IT

DURATION OF TRAINING :2ND DEC 2019 TO 2ND JAN 2020

CONTENTS

ACKNOWLEDGEMENT

The aim of this training is to provide the basic guidelines and precautions to the trainees for their better understanding of the various technology and their application in life and be aware of the hazardous area in the industrial environment. Training and such other methods have been used to develop a strategy to invoke correct principles and work it out.

I wish to express my deepest gratitude towards my guide **Mr.RITESH KARN** for his valuable guidance and suggestion, persistent encouragement, creative criticism and scientific approach throughout the duration of my project work. His vast knowledge and depth of perception were of invaluable assistance in framing a suitable outline for the "**ONLINE CARBON RATE OPTIMIZATION**"

I extend my sincere thanks to everyone in A&I for their constant support and cooperation in letting me understand various operations in and around A&I and other departments of this plant.

It has been a pleasure to work with such a group of experienced professionals and most importantly in such an ethical environment.

DATE:2 JAN 2020

SHARFUL HAQUE

SRM IST-NCR

TRAINING CERTIFICATE

This to certify that **SHARFUL HAQUE OF SRM IST-NCR,**REGNo.-**VT20193943**

who is presently pursuing his **B.Tech** degree in **COMPUTER SCIENCE ENGINEERING**discipline has successfully under taken his training from 2DEC 2019 to 2JAN 2020 (4 weeks) on the project entitled**"ONLINE CARBON RATE OPTIMIZATION "** under our guidance at **TATA STEEL, Jamshedpur**.

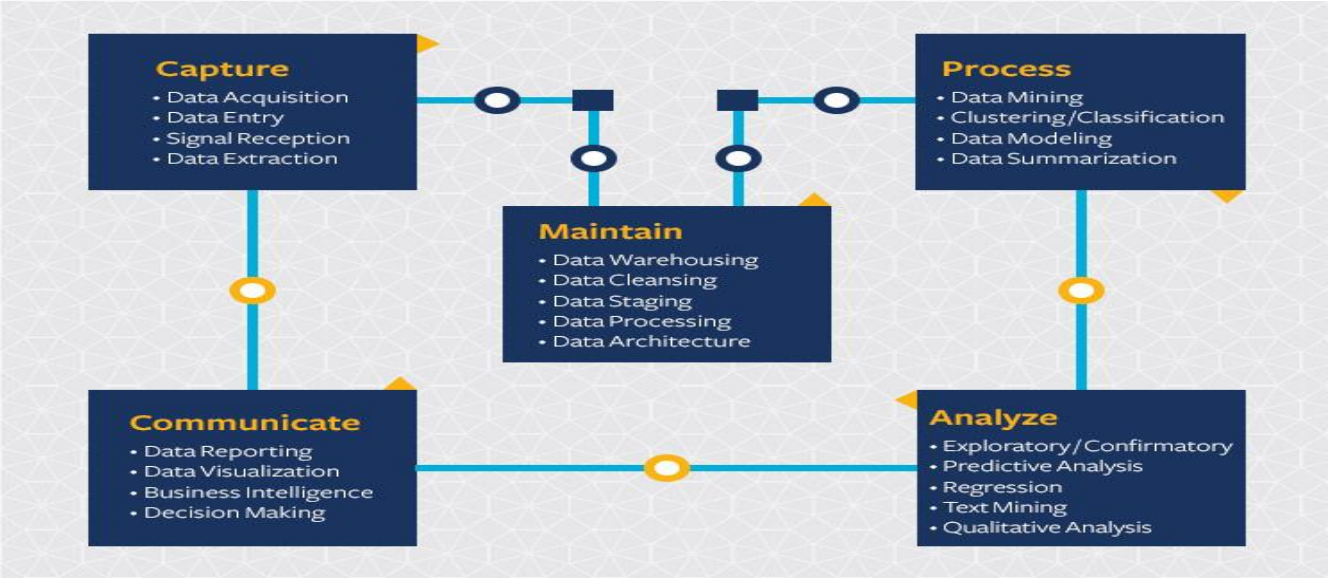
He has completed his project and training to our complete satisfaction within the allotted time.

MR .RITESH KARN
IT MANAGER
1IT
TATA STEEL INDIA LTD.
JAMSHEDPUR

INTRODUCTION

Data science is a multi-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from structured and unstructured data. Data science is related to data mining and big data.Data science is a "concept to unify statistics, data analysis, machine learning and their related methods" in order to "understand and analyze actual phenomena" with data. It employs techniques and theories drawn from many fields within the context of mathematics, statistics, computer science, and information science. Turing award winner Jim Gray imagined data science as a "fourth paradigm" of science (empirical, theoretical, computational and now data-driven) and asserted that "everything about science is changing because of the impact of information technology" and the data deluge.[4][5] In 2015, the American Statistical Association identified database management, statistics and machine learning, and distributed and parallel systems as the three emerging foundational professional communities.

Data science continues to evolve as one of the most promising and in-demand career paths for skilled professionals. Today, successful data professionals understand that they must advance past the traditional skills of analyzing large amounts of data, data mining, and programming skills. In order to uncover useful intelligence for their organizations, data scientists must master the full spectrum of the data science life cycle and possess a level of flexibility and understanding to maximize returns at each phase of the process.

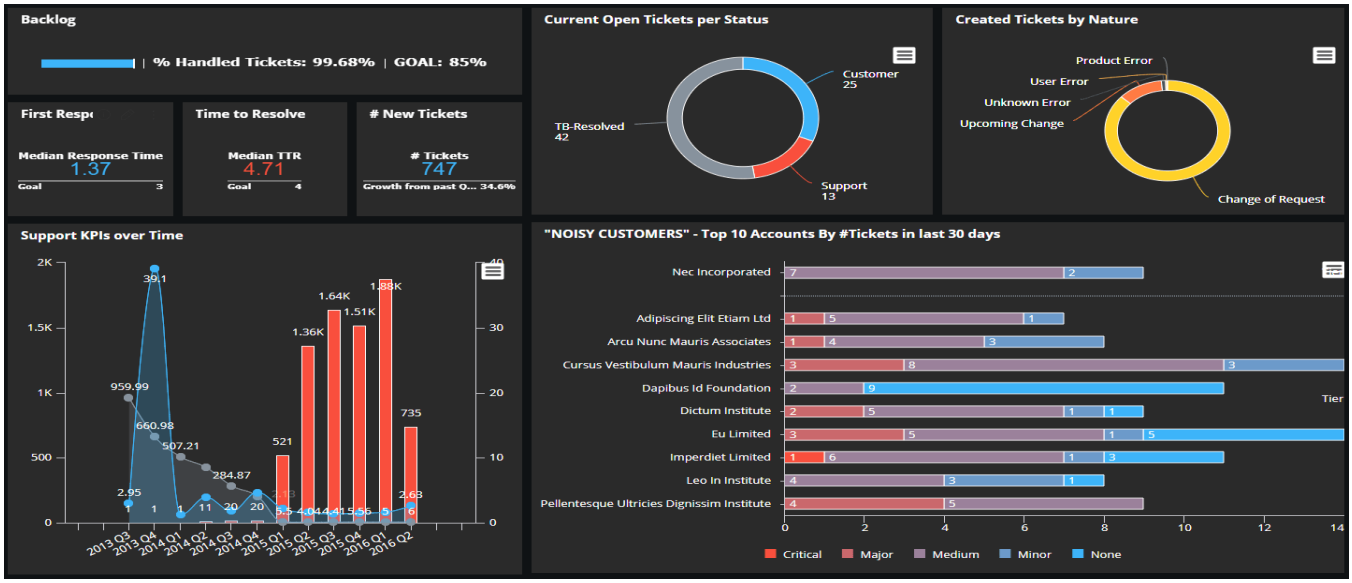


DATA EXPLORATION

Data exploration is the initial step in data analysis, where users explore a large data set in an unstructured way to uncover initial patterns, characteristics, and points of interest. This process isn't meant to reveal every bit of information a dataset holds, but rather to help create a broad picture of important trends and major points to study in greater detail.Data exploration can use a combination of manual methods and automated tools such as data visualizations, charts, and initial reports.

This process makes deeper analysis easier because it can help target future searches and begin the process of excluding irrelevant data points and search paths that may turn up no results. More importantly, it helps build a familiarity with the existing information that makes finding better answers much simpler.Many times, data exploration uses visualization because it creates a more straightforward view of data sets than simply examining thousands of individual numbers or names.In any data exploration, the manual and automated aspects also look at different sides of the same coin. Manual analysis helps users familiarize themselves with information and can point to broad trends.These methods are also by definition unstructured so that users can examine a whole set without any preconceptions. Automated tools, on the other hand, are excellent at pruning out less applicable data points, reorganizing data into sets that are easier to analyze, and scrubbing data sets to make their findings relevant.

Most data analytics software includes visualization tools and charting features that make exploration at the outset significantly easier, helping reduce data by rooting out information that isn't required, or which can distort results in the long run.



1. **Steps of Data Exploration and Preparation**
2. **Missing Value Treatment**
    - Why missing value treatment is required ?
    - Why data has missing values?
    - Which are the methods to treat missing value ?
3. **Techniques of Outlier Detection and Treatment**
    - What is an outlier?
    - What are the types of outliers ?
    - What are the causes of outliers ?
    - What is the impact of outliers on dataset ?
    - How to detect outlier ?
    - How to remove outlier ?
4. **The Art of Feature Engineering**
    - What is Feature Engineering ?
    - What is the process of Feature Engineering ?
    - What is Variable Transformation ?
    - When should we use variable transformation ?
    - What are the common methods of variable transformation ?
    - What is feature variable creation and its benefits ?

1. Steps of Data Exploration and Preparation

Remember the quality of your inputs decide the quality of your output. So, once you have got your business hypothesis ready, it makes sense to spend lot of time and efforts here. With my personal estimate, data exploration, cleaning and preparation can take up to 70% of your total project time.

Below are the steps involved to understand, clean and prepare your data for building your predictive model:

1. Variable Identification
2. Univariate Analysis
3. Bi-variate Analysis
4. Missing values treatment
5. Outlier treatment
6. Variable transformation
7. Variable creation

Finally, we will need to iterate over steps 4 – 7 multiple times before we come up with our refined model.

Let’s now study each stage in detail:-

Variable Identification

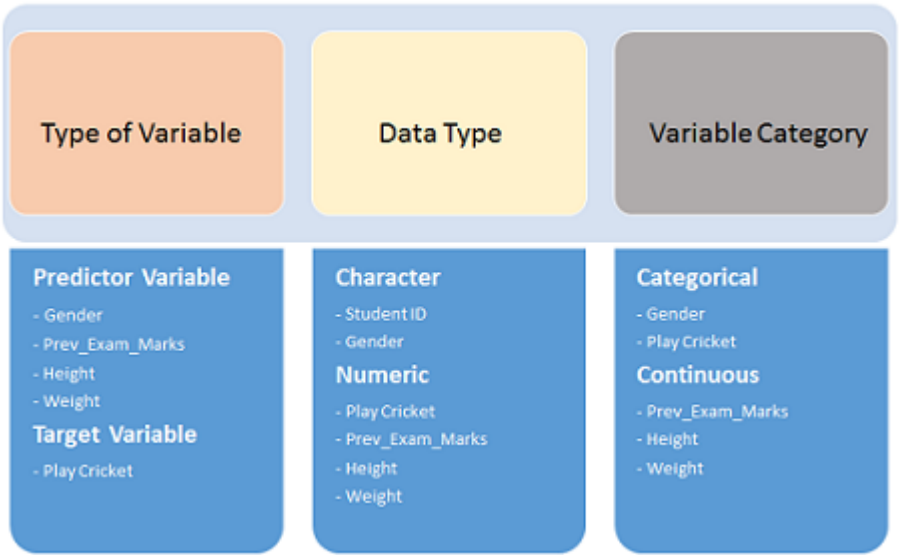
First, identify **Predictor** (Input) and **Target** (output) variables. Next, identify the data type and category of the variables.

Let's understand this step more clearly by taking an example.

Example:- Suppose, we want to predict, whether the students will play cricket or not (refer below data set). Here you need to identify predictor variables, target variable, data type of variables and category of variables.

Student_ID	Gender	Prev_Exam_Marks	Height (cm)	Weight Caregory (kgs)	Play Cricket
S001	M	65	178	61	1
S002	F	75	174	56	0
S003	M	45	163	62	1
S004	M	57	175	70	0
S005	F	59	162	67	0

Below, the variables have been defined in different category:

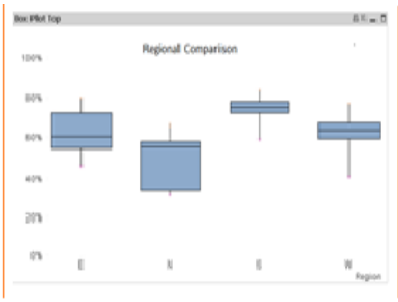


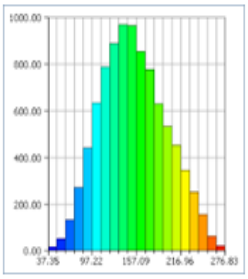
Univariate Analysis

At this stage, we explore variables one by one. Method to perform uni-variate analysis will depend on whether the variable type is categorical or continuous. Let's look at these methods and statistical measures for categorical and continuous variables individually:

Continuous Variables:- In case of continuous variables, we need to understand the central tendency and spread of the variable. These are measured using various statistical metrics visualization methods as shown below:

Central Tendency	Measure of Dispersion	Visualization Methods
Mean	Range	Histogram
Median	Quartile	Box Plot
Mode	IQR	
Min	Variance	
Max	Standard Deviation	
	Skewness and Kurtosis	





Note: Univariate analysis is also used to highlight missing and outlier values. In the upcoming part of this series, we will look at methods to handle missing and outlier values. To know more about these methods, you can refer course descriptive statistics from Udacity.

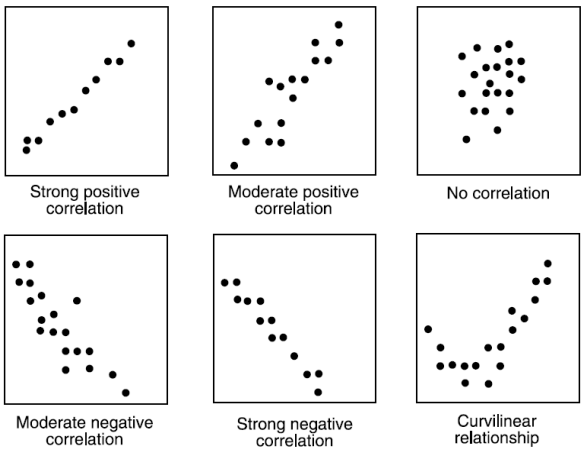
Categorical Variables:- For categorical variables, we'll use frequency table to understand distribution of each category. We can also read as percentage of values under each category. It can be be measured using two metrics, **Count** and **Count%** against each category. Bar chart can be used as visualization.

Bi-variate Analysis

Bi-variate Analysis finds out the relationship between two variables. Here, we look for association and disassociation between variables at a pre-defined significance level. We can perform bi-variate analysis for any combination of categorical and continuous variables. The combination can be: Categorical & Categorical, Categorical & Continuous and Continuous & Continuous. Different methods are used to tackle these combinations during analysis process.

Let’s understand the possible combinations in detail:

Continuous & Continuous: While doing bi-variate analysis between two continuous variables, we should look at scatter plot. It is a nifty way to find out the relationship between two variables. The pattern of scatter plot indicates the relationship between variables. The relationship can be linear or non-linear.



Scatter plot shows the relationship between two variable but does not indicates the strength of relationship amongst them. To find the strength of the relationship, we use Correlation. Correlation varies between -1 and +1.

- -1: perfect negative linear correlation
- +1:perfect positive linear correlation and
- 0: No correlation

Correlation can be derived using following formula:

Correlation = Covariance(X,Y) / SQRT(Var(X)* Var(Y))

Various tools have function or functionality to identify correlation between variables. In Excel, function CORREL() is used to return the correlation between two variables and SAS uses procedure PROC CORR to identify the correlation. These function returns Pearson Correlation value to identify the relationship between two variables:

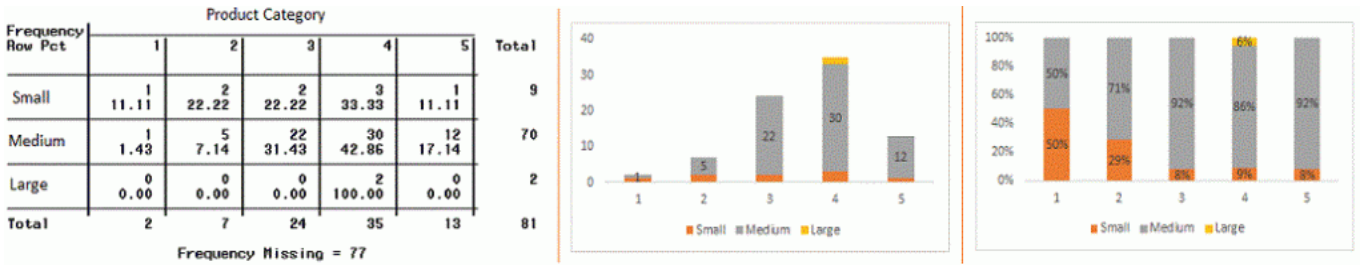
X	65	72	78	65	72	70	65	68
Y	72	69	79	69	84	75	60	73

Metrics	Formula	Value
Co-Variance (X,Y)	=COVAR(E6:L6,E7:L7)	18.77
Variance (X)	=VAR.P(E6:L6)	18.48
Variance (Y)	=VAR.P(E7:L7)	45.23
Correlation	=G10/SQRT(G11*G12)	0.65

In above example, we have good positive relationship(0.65) between two variables X and Y.

Categorical & Categorical: To find the relationship between two categorical variables, we can use following methods:

- **Two-way table:** We can start analyzing the relationship by creating a two-way table of count and count%. The rows represents the category of one variable and the columns represent the categories of the other variable. We show count or count% of observations available in each combination of row and column categories.
- **Stacked Column Chart:** This method is more of a visual form of Two-way table.



Product Category

Frequency Row Pct	1	2	3	4	5	Total
Small	1 11.11	2 22.22	2 22.22	3 33.33	1 11.11	9
Medium	1 1.43	5 7.14	22 31.43	30 42.86	12 17.14	70
Large	0 0.00	0 0.00	0 0.00	2 100.00	0 0.00	2
Total	2	7	24	35	13	81

Frequency Missing = 77

- **Chi-Square Test:** This test is used to derive the statistical significance of relationship between the variables. Also, it tests whether the evidence in the sample is strong enough to generalize that the relationship for a larger population as well. Chi-square is based on the difference between the expected and observed frequencies in one or more categories in the two-way table. It returns probability for the computed chi-square distribution with the degree of freedom.

Probability of 0: It indicates that both categorical variable are dependent

Probability of 1: It shows that both variables are independent.

Probability less than 0.05: It indicates that the relationship between the variables is significant at 95% confidence. The chi-square test statistic for a test of independence of two categorical variables is found by:

$X^2 = \sum (O - E)^2 / E$ where *O* represents the observed frequency. *E* is the expected frequency under the null hypothesis and computed by: $E = \frac{row\ total \times column\ total}{sample\ size}$

From previous two-way table, the expected count for product category 1 to be of small size is 0.22. It is derived by taking the row total for Size (9) times the column total for Product category (2) then dividing by the sample size (81). This is procedure is conducted for each cell. Statistical Measures used to analyze the power of relationship are:

- Cramer's V for Nominal Categorical Variable
- Mantel-Haenszed Chi-Square for ordinal categorical variable.

Different data science language and tools have specific methods to perform chi-square test. In SAS, we can use **Chisq** as an option with **Proc freq** to perform this test.

Categorical & Continuous: While exploring relation between categorical and continuous variables, we can draw box plots for each level of categorical variables. If levels are small in number, it will not show the statistical significance. To look at the statistical significance we can perform Z-test, T-test or ANOVA.

- **Z-Test/ T-Test:-** Either test assess whether mean of two groups are statistically different from each other or not. $z = \frac{\left|\bar{x}_1 - \bar{x}_2\right|}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$ If the probability of Z is small then the difference of two averages is more significant. The T-test is very similar to Z-test but it is used when number of

observation for both categories is less than 30.

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{S^2\left(\frac{1}{N_1} + \frac{1}{N_2}\right)}}$$

$$S^2 = \frac{(N_1 - 1)S_1^2 + (N_2 - 1)S_2^2}{N_1 + N_2 - 2}$$

where:

- $\overline{X}_1, \overline{X}_2$: *Averages*
- S_1^2, S_2^2 : *Variances*
- N_1, N_2 : *Counts*
- t : has t distribution with $N_1 + N_2 - 2$ degree of freedom

- **ANOVA:-** It assesses whether the average of more than two groups is statistically different.

Example: Suppose, we want to test the effect of five different exercises. For this, we recruit 20 men and assign one type of exercise to 4 men (5 groups). Their weights are recorded after a few weeks. We need to find out whether the effect of these exercises on them is significantly different or not. This can be done by comparing the weights of the 5 groups of 4 men each.

Till here, we have understood the first three stages of Data Exploration, Variable Identification, Uni-Variate and Bi-Variate analysis. We also looked at various statistical and visual methods to identify the relationship between variables.

Now, we will look at the methods of Missing values Treatment. More importantly, we will also look at why missing values occur in our data and why treating them is necessary.

2. Missing Value Treatment

Why missing values treatment is required?

Missing data in the training data set can reduce the power / fit of a model or can lead to a biased model because we have not analysed the behavior and relationship with other variables correctly. It can lead to wrong prediction or classification.

Name	Weight	Gender	Play Cricket/ Not
Mr. Amit	58	M	Y
Mr. Anil	61	M	Y
Miss Swati	58	F	N
Miss Richa	55		Y
Mr. Steve	55	M	N
Miss Reena	64	F	Y
Miss Rashmi	57		Y
Mr. Kunal	57	M	N

Gender	#Students	#Play Cricket	%Play Cricket
F	2	1	50%
M	4	2	50%
Missing	2	2	100%

Name	Weight	Gender	Play Cricket/ Not
Mr. Amit	58	M	Y
Mr. Anil	61	M	Y
Miss Swati	58	F	N
Miss Richa	55	F	Y
Mr. Steve	55	M	N
Miss Reena	64	F	Y
Miss Rashmi	57	F	Y
Mr. Kunal	57	M	N

Gender	#Students	#Play Cricket	%Play Cricket
F	4	3	75%
M	4	2	50%

Notice the missing values in the image shown above: In the left scenario, we have not treated missing values. The inference from this data set is that the chances of playing cricket by males is higher than females. On the other hand, if you look at the second table, which shows data after treatment of missing values (based on gender), we can see that females have higher chances of playing cricket compared to males.

Why my data has missing values?

We looked at the importance of treatment of missing values in a dataset. Now, let's identify the reasons for occurrence of these missing values. They may occur at two stages:

1. **Data Extraction**: It is possible that there are problems with extraction process. In such cases, we should double-check for correct data with data guardians. Some hashing procedures can also be used to make sure data extraction is correct. Errors at data extraction stage are typically easy to find and can be corrected easily as well.
2. **Data collection**: These errors occur at time of data collection and are harder to correct. They can be categorized in four types:
    - **Missing completely at random:** This is a case when the probability of missing variable is same for all observations. For example: respondents of data collection process decide that they will declare their earning after tossing a fair coin. If an head occurs, respondent declares his / her earnings & vice versa. Here each observation has equal chance of missing value.

- **Missing at random:** This is a case when variable is missing at random and missing ratio varies for different values / level of other input variables. For example: We are collecting data for age and female has higher missing value compare to male.
- **Missing that depends on unobserved predictors:** This is a case when the missing values are not random and are related to the unobserved input variable. For example: In a medical study, if a particular diagnostic causes discomfort, then there is higher chance of drop out from the study. This missing value is not at random unless we have included "discomfort" as an input variable for all patients.
- **Missing that depends on the missing value itself:** This is a case when the probability of missing value is directly correlated with missing value itself. For example: People with higher or lower income are likely to provide non-response to their earning.

Which are the methods to treat missing values ?

1. **Deletion:** It is of two types: List Wise Deletion and Pair Wise Deletion.
   - In list wise deletion, we delete observations where any of the variable is missing. Simplicity is one of the major advantage of this method, but this method reduces the power of model because it reduces the sample size.
   - In pair wise deletion, we perform analysis with all cases in which the variables of interest are present. Advantage of this method is, it keeps as many cases available for analysis. One of the disadvantage of this method, it uses different sample size for different variables.

List wise deletion

Gender	Manpower	Sales
M	25	343
F	.	280
M	33	332
M	.	272
F	25	.
M	29	326
	26	259
M	32	297

Pair wise deletion

Gender	Manpower	Sales
M	25	343
F	.	280
M	33	332
M	.	272
F	25	.
M	29	326
	26	259
M	32	297

- Deletion methods are used when the nature of missing data is "**Missing completely at random**" else non random missing values can bias the model output.

2. **Mean/ Mode/ Median Imputation:** Imputation is a method to fill in the missing values with estimated ones. The objective is to employ known relationships that can be identified in the valid values of the data set to assist in estimating the missing values. Mean / Mode / Median imputation is one of the most frequently used methods. It consists of replacing the missing data for a given attribute by the mean or median (quantitative attribute) or mode (qualitative attribute) of all known values of that variable. It can be of two types:-
    - **Generalized Imputation:** In this case, we calculate the mean or median for all non missing values of that variable then replace missing value with mean or median. Like in above table, variable "**Manpower"** is missing so we take average of all non missing values of "**Manpower"** (**28.33**) and then replace missing value with it.
    - **Similar case Imputation:** In this case, we calculate average for gender "**Male"** (29.75) and "**Female**" (25) individually of non missing values then replace the missing value based on gender. For "**Male**", we will replace missing values of manpower with 29.75 and for "**Female**" with 25.
3. **Prediction Model**: Prediction model is one of the sophisticated method for handling missing data. Here, we create a predictive model to estimate values that will substitute the missing data. In this case, we divide our data set into two sets: One set with no missing values for the variable and another one with missing values. First data set become training data set of the model while second data set with missing values is test data set and variable with missing values is treated as target variable. Next, we create a model to predict target variable based on other attributes of the training data set and populate missing values of test data set.We can use regression, ANOVA, Logistic regression and various modeling technique to perform this. There are 2 drawbacks for this approach:
    1. The model estimated values are usually more well-behaved than the true values
    2. If there are no relationships with attributes in the data set and the attribute with missing values, then the model will not be precise for estimating missing values.
4. **KNN Imputation:** In this method of imputation, the missing values of an attribute are imputed using the given number of attributes that are most similar to the attribute whose values are missing. The similarity of two attributes is determined using a distance function. It is also known to have certain advantage & disadvantages.

- **Advantages:**
  - k-nearest neighbour can predict both qualitative & quantitative attributes
  - Creation of predictive model for each attribute with missing data is not required
  - Attributes with multiple missing values can be easily treated
  - Correlation structure of the data is taken into consideration
- **Disadvantage:**
  - KNN algorithm is very time-consuming in analyzing large database. It searches through all the dataset looking for the most similar instances.
  - Choice of k-value is very critical. Higher value of k would include attributes which are significantly different from what we need whereas lower value of k implies missing out of significant attributes.

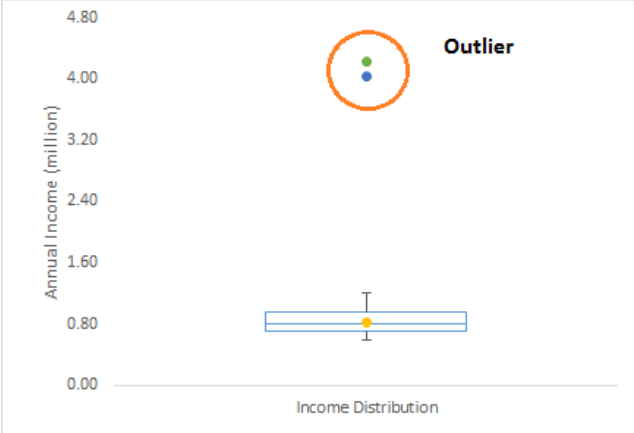
After dealing with missing values, the next task is to deal with outliers. Often, we tend to neglect outliers while building models. This is a discouraging practice. Outliers tend to make your data skewed and reduces accuracy. Let's learn more about outlier treatment.

3. Techniques of Outlier Detection and Treatment

What is an Outlier?

Outlier is a commonly used terminology by analysts and data scientists as it needs close attention else it can result in wildly wrong estimations. Simply speaking, Outlier is an observation that appears far away and diverges from an overall pattern in a sample.

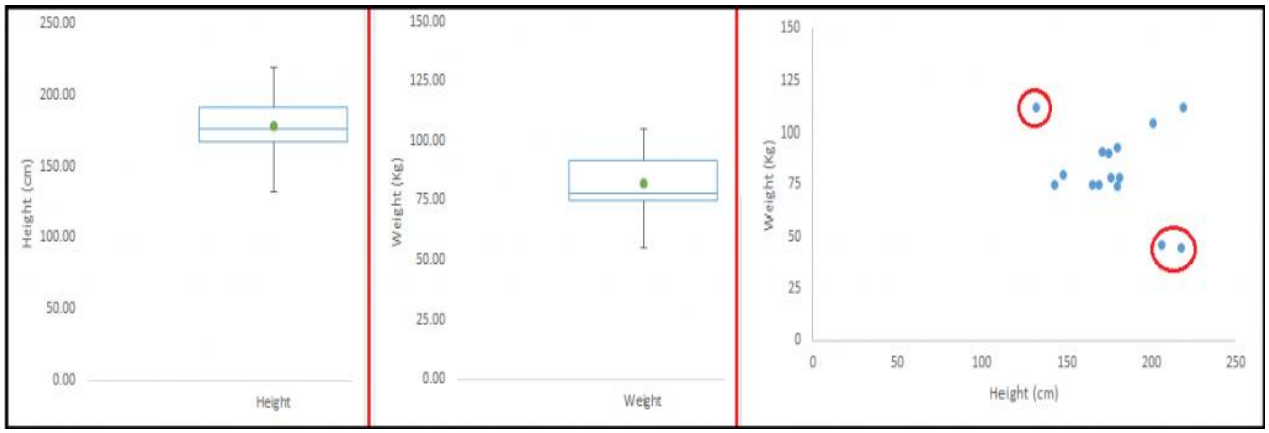
Let's take an example, we do customer profiling and find out that the average annual income of customers is $0.8 million. But, there are two customers having annual income of $4 and $4.2 million. These two customers annual income is much higher than rest of the population. These two observations will be seen as Outliers.



What are the types of Outliers?

Outlier can be of two types: **Univariate** and **Multivariate**. Above, we have discussed the example of univariate outlier. These outliers can be found when we look at distribution of a single variable. Multi-variate outliers are outliers in an n-dimensional space. In order to find them, you have to look at distributions in multi-dimensions.

Let us understand this with an example. Let us say we are understanding the relationship between height and weight. Below, we have univariate and bivariate distribution for Height, Weight. Take a look at the box plot. We do not have any outlier (above and below 1.5*IQR, most common method). Now look at the scatter plot. Here, we have two values below and one above the average in a specific segment of weight and height.



What causes Outliers?

Whenever we come across outliers, the ideal way to tackle them is to find out the reason of having these outliers. The method to deal with them would then depend on the reason of their occurrence. Causes of outliers can be classified in two broad categories:

1. **Artificial (Error) / Non-natural**
2. **Natural**.

Let's understand various types of outliers in more detail:

- **Data Entry Errors:-** Human errors such as errors caused during data collection, recording, or entry can cause outliers in data. For example: Annual income of a customer is $100,000. Accidentally, the data entry operator puts an additional zero in the figure. Now the income becomes $1,000,000 which is 10 times higher. Evidently, this will be the outlier value when compared with rest of the population.
- **Measurement Error:** It is the most common source of outliers. This is caused when the measurement instrument used turns out to be faulty. For example: There are 10 weighing machines. 9 of them are correct, 1 is faulty. Weight measured by people on the faulty machine will be higher / lower than the rest of people in the group. The weights measured on faulty machine can lead to outliers.
- **Experimental Error:** Another cause of outliers is experimental error. For example: In a 100m sprint of 7 runners, one runner missed out on concentrating on the 'Go' call which caused him to start late. Hence, this caused the runner's run time to be more than other runners. His total run time can be an outlier.
- **Intentional Outlier*:*** This is commonly found in self-reported measures that involves sensitive data. For example: Teens would typically under report the amount of alcohol that they consume. Only a fraction of them would report actual value. Here actual values might look like outliers because rest of the teens are under reporting the consumption.
- **Data Processing Error:** Whenever we perform data mining, we extract data from multiple sources. It is possible that some manipulation or extraction errors may lead to outliers in the dataset.
- **Sampling error:** For instance, we have to measure the height of athletes. By mistake, we include a few basketball players in the sample. This inclusion is likely to cause outliers in the dataset.
- **Natural Outlier:** When an outlier is not artificial (due to error), it is a natural outlier. For instance: In my last assignment with one of the

renowned insurance company, I noticed that the performance of top 50 financial advisors was far higher than rest of the population. Surprisingly, it was not due to any error. Hence, whenever we perform any data mining activity with advisors, we used to treat this segment separately.

What is the impact of Outliers on a dataset?

Outliers can drastically change the results of the data analysis and statistical modeling. There are numerous unfavourable impacts of outliers in the data set:

- It increases the error variance and reduces the power of statistical tests
- If the outliers are non-randomly distributed, they can decrease normality
- They can bias or influence estimates that may be of substantive interest
- They can also impact the basic assumption of Regression, ANOVA and other statistical model assumptions.

To understand the impact deeply, let's take an example to check what happens to a data set with and without outliers in the data set.

Example:

Without Outlier	With Outlier
4, 4, 5, 5, 5, 5, 6, 6, 6, 7, 7	4, 4, 5, 5, 5, 5, 6, 6, 6, 7, 7,300
Mean = 5.45	Mean = 30.00
Median = 5.00	Median = 5.50
Mode = 5.00	Mode = 5.00
Standard Deviation = 1.04	Standard Deviation = 85.03

As you can see, data set with outliers has significantly different mean and standard deviation. In the first scenario, we will say that average is 5.45. But with the outlier, average soars to 30. This would change the estimate completely.

How to detect Outliers?

Most commonly used method to detect outliers is visualization. We use various visualization methods, like **Box-plot**, **Histogram**, **Scatter Plot** (above, we have used box plot and scatter plot for visualization). Some analysts also various thumb rules to detect outliers. Some of them are:

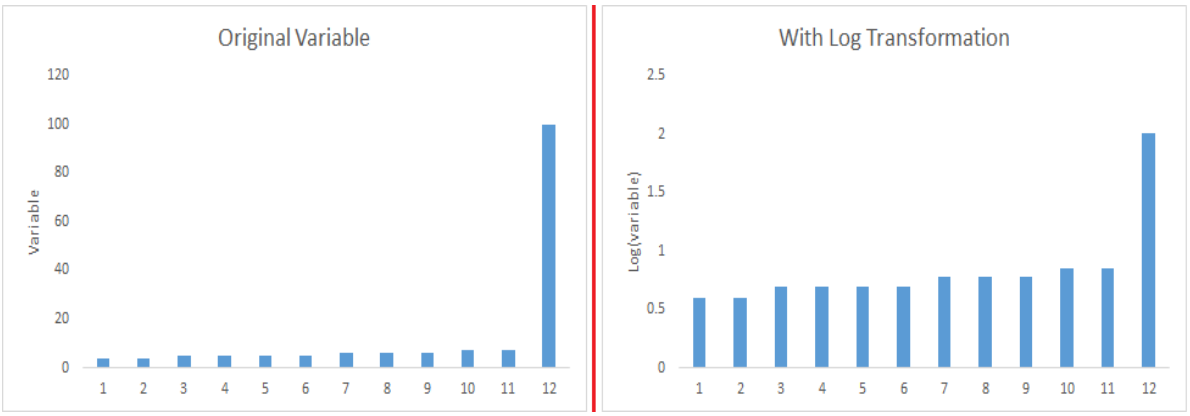
- Any value, which is beyond the range of -1.5 x IQR to 1.5 x IQR
- Use capping methods. Any value which out of range of 5th and 95th percentile can be considered as outlier
- Data points, three or more standard deviation away from mean are considered outlier
- Outlier detection is merely a special case of the examination of data for influential data points and it also depends on the business understanding
- Bivariate and multivariate outliers are typically measured using either an index of influence or leverage, or distance. Popular indices such as Mahalanobis' distance and Cook's *D* are frequently used to detect outliers.
- In SAS, we can use PROC Univariate, PROC SGPLOT. To identify outliers and influential observation, we also look at statistical measure like STUDENT, COOKD, RSTUDENT and others.

How to remove Outliers?

Most of the ways to deal with outliers are similar to the methods of missing values like deleting observations, transforming them, binning them, treat them as a separate group, imputing values and other statistical methods. Here, we will discuss the common techniques used to deal with outliers:

Deleting observations: We delete outlier values if it is due to data entry error, data processing error or outlier observations are very small in numbers. We can also use trimming at both ends to remove outliers.

Transforming and binning values: Transforming variables can also eliminate outliers. Natural log of a value reduces the variation caused by extreme values. Binning is also a form of variable transformation. Decision Tree algorithm allows to deal with outliers well due to binning of variable. We can also use the process of assigning weights to different observations.



Imputing: Like imputation of missing values, we can also impute outliers. We can use mean, median, mode imputation methods. Before imputing values, we should analyse if it is natural outlier or artificial. If it is artificial, we can go with imputing values. We can also use statistical model to predict values of outlier observation and after that we can impute it with predicted values.

Treat separately: If there are significant number of outliers, we should treat them separately in the statistical model. One of the approach is to treat both groups as two different groups and build individual model for both groups and then combine the output.

Till here, we have learnt about steps of data exploration, missing value treatment and techniques of outlier detection and treatment. These 3 stages will make your raw data better in terms of information availability and accuracy. Let's now proceed to the final stage of data exploration. It is Feature Engineering.

4. The Art of Feature Engineering

What is Feature Engineering?

Feature engineering is the science (and art) of extracting more information from existing data. You are not adding any new data here, but you are actually making the data you already have more useful.

For example, let's say you are trying to predict foot fall in a shopping mall based on dates. If you try and use the dates directly, you may not be able to extract meaningful insights from the data. This is because the foot fall is less affected by the day of the month than it is by the day of the week. Now this information about day of week is implicit in your data. You need to bring it out to make your model better.

This exercising of bringing out information from data in known as feature engineering.

What is the process of Feature Engineering ?

You perform feature engineering once you have completed the first 5 steps in data exploration – Variable Identification, Univariate, Bivariate Analysis, Missing Values Imputation and Outliers Treatment. Feature engineering itself can be divided in 2 steps:

- Variable transformation.
- Variable / Feature creation.

These two techniques are vital in data exploration and have a remarkable impact on the power of prediction. Let's understand each of this step in more details.

What is Variable Transformation?

In data modelling, transformation refers to the replacement of a variable by a function. For instance, replacing a variable x by the square / cube root or logarithm x is a transformation. In other words, transformation is a process that changes the distribution or relationship of a variable with others.

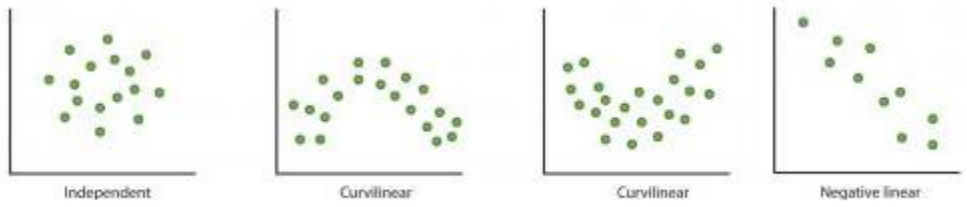
Let's look at the situations when variable transformation is useful.

When should we use Variable Transformation?

Below are the situations where variable transformation is a requisite:

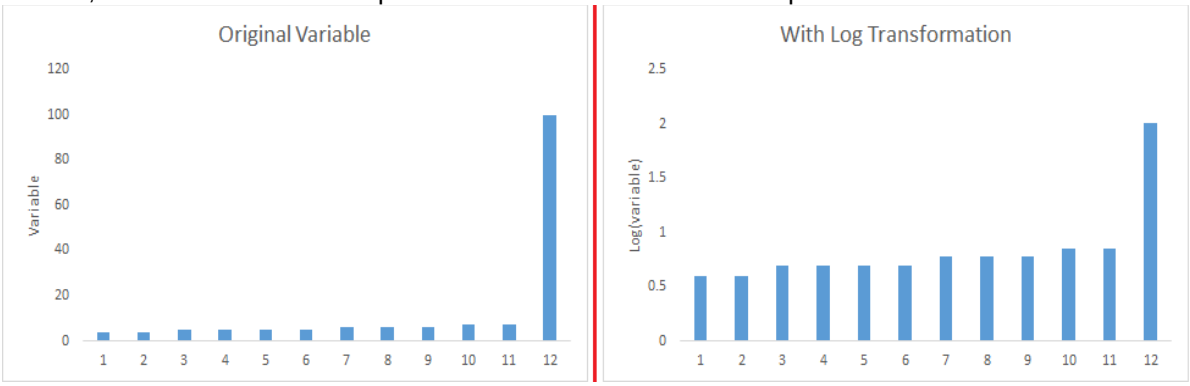
- When we want to **change the scale** of a variable or standardize the values of a variable for better understanding. While this transformation is a must if you have data in different scales, this transformation does not change the shape of the variable distribution

- When we can **transform complex non-linear relationships into linear relationships**. Existence of a linear relationship between variables is easier to comprehend compared to a non-linear or curved relation. Transformation helps us to convert a non-linear relation into linear relation. Scatter plot can be used to find the relationship between two continuous variables. These transformations also improve the prediction. Log transformation is one of the commonly used transformation technique used in these situations.



- **Symmetric distribution is preferred over skewed distribution** as it is easier to interpret and generate inferences. Some modeling techniques requires normal distribution of variables. So,

whenever we have a skewed distribution, we can use transformations which reduce skewness. For right skewed distribution, we take square / cube root or logarithm of variable and for left skewed, we take square / cube or exponential of variables.



- Variable Transformation is also done from an **implementation point of view** (Human involvement). Let's understand it more clearly. In one of my project on employee performance, I found that age has direct correlation with performance of the employee i.e. higher the age, better the performance. From an implementation stand point, launching age based progamme might present implementation challenge. However, categorizing the sales agents in three age group buckets of <30 years, 30-45 years and >45 and then formulating three different strategies for each group is a judicious approach. This categorization technique is known as Binning of Variables.

What are the common methods of Variable Transformation?

There are various methods used to transform variables. As discussed, some of them include square root, cube root, logarithmic, binning, reciprocal and many others. Let's look at these methods in detail by highlighting the pros and cons of these transformation methods.

- **Logarithm:** Log of a variable is a common transformation method used to change the shape of distribution of the variable on a distribution plot. It is generally used for reducing right skewness of variables. Though, It can't be applied to zero or negative values as well.

- **Square / Cube root:** The square and cube root of a variable has a sound effect on variable distribution. However, it is not as significant as logarithmic transformation. Cube root has its own advantage. It can be applied to negative values including zero. Square root can be applied to positive values including zero.

- **Binning:** It is used to categorize variables. It is performed on original values, percentile or frequency. Decision of categorization technique is based on business understanding. For example, we can categorize income in three categories, namely: High, Average and Low. We can also perform co-variate binning which depends on the value of more than one variables.

What is Feature / Variable Creation & its Benefits?

Feature / Variable creation is a process to generate a new variables / features based on existing variable(s). For example, say, we have date(dd-mm-yy) as an input variable in a data set. We can generate new variables like day, month, year, week, weekday that may have better relationship with target variable. This step is used to highlight the hidden relationship in a variable:

Emp_Code	Gender	Date	New_Day	New_Month	New_Year
A001	Male	21-Sep-11	21	9	2011
A002	Female	27-Feb-13	27	2	2013
A003	Female	14-Nov-12	14	11	2012
A004	Male	07-Apr-13	7	4	2013
A005	Female	21-Jan-11	21	1	2011
A006	Male	26-Apr-13	26	4	2013
A007	Male	15-Mar-12	15	3	2012

There are various techniques to create new features. Let's look at the some of the commonly used methods:

- **Creating derived variables:** This refers to creating new variables from existing variable(s) using set of functions or different methods. Let's look at it through "**Titanic – Kaggle competition**". In this data set, variable age has missing values. To predict missing values, we used the salutation (Master, Mr, Miss, Mrs) of name as a new variable. How do we decide which variable to create? Honestly, this depends on business understanding of the analyst, his curiosity and the set of hypothesis he might have about the problem. Methods such as taking log of variables, binning variables and other methods of variable transformation can also be used to create new variables.

- **Creating dummy variables:** One of the most common application of dummy variable is to convert categorical variable into numerical variables. Dummy variables are also called Indicator Variables. It is useful to take categorical variable as a predictor in statistical models. Categorical variable can take values 0 and 1. Let's take a variable 'gender'. We can produce two variables, namely, "**Var_Male**" with values 1 (Male) and 0 (No male) and "**Var_Female**" with values 1 (Female) and 0 (No Female). We can also create dummy variables for more than two classes of a categorical variables with n or n-1 dummy variables.

Emp_Code	Gender	Var_Male	Var_Female
A001	Male	1	0
A002	Female	0	1
A003	Female	0	1
A004	Male	1	0
A005	Female	0	1
A006	Male	1	0
A007	Male	1	0

MACHINE LEARNING

Machine Learning is the science of teaching machines how to learn by themselves. Now, you might be thinking – why on earth would we want machines to learn by themselves? Well – it has a lot of benefits. Machine learning creates a useful model or program by autonomously testing many solutions against the available data and finding the best fit for the problem. This means machine learning is great at solving problems that are extremely labor intensive for humans. It can inform decisions and make predictions about complex topics in an efficient and reliable way.

Machine learning is one of the many tools in the belt of a data scientist. In order to make machine learning work, you need a skilled data scientist who can organize data and apply the proper tools to fully make use of the numbers.

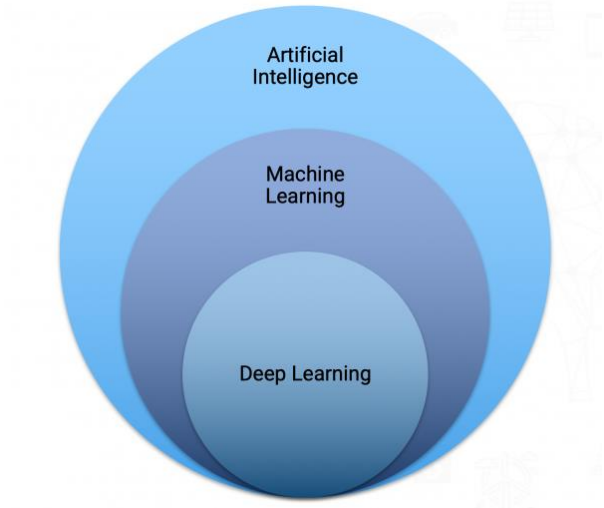
What tools are used in Machine Learning?

There are several tools and languages being used in machine learning. The exact choice of the tool depends on your need and scale of operations. But, here are the most commonly used tools in machine learning:

- Languages:
  - R
  - Python
  - SAS
  - Julia
  - Java
  - Javascript
  - Scala
- Databases:
  - SQL
  - Oracle
  - Hadoop
- Visualization tools:
  - D3.js
  - Tableau
  - QlikView
- Other tools commonly used:
  - Excel
  - Powerpoint

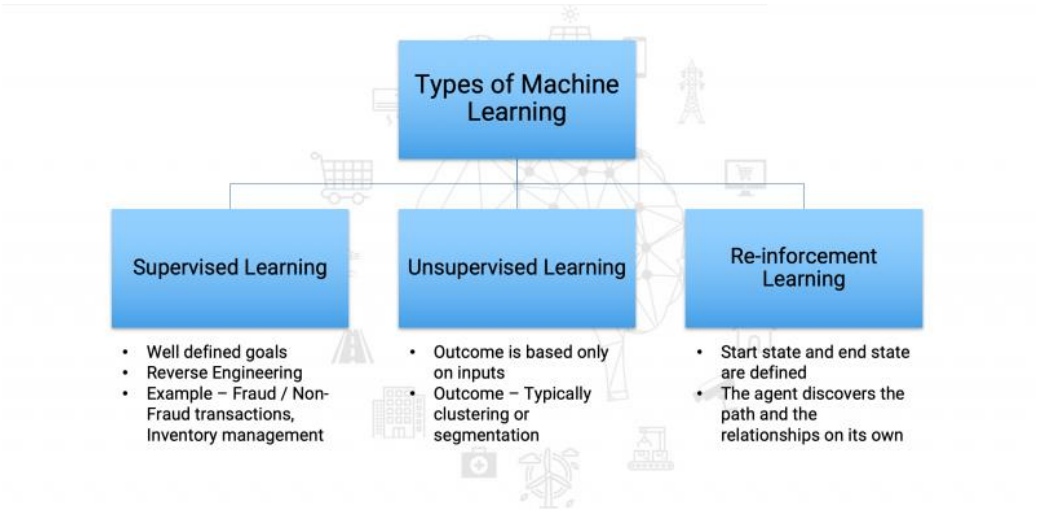
How is Machine Learning Different from Deep Learning?

Deep learning is actually a sub-field of Machine Learning. So, if you were to represent Machine Learning and Deep Learning by a simple Venn-diagram – it will look like this:



Machine Learning problems can be divided into 3 broad classes:

- **Supervised Machine Learning:** When you have past data with outcomes (labels in machine learning terminology) and you want to predict the outcomes for the future – you would use Supervised Machine Learning algorithms. Supervised Machine Learning problems can again be divided into 2 kinds of problems:
    - **Classification Problems:** When you want to classify outcomes into different classes. For example – whether the floor needs cleaning/mopping is a classification problem. The outcome can fall into one of the classes – Yes or No. Similarly, whether a customer would default on their loan or not is a classification problem which is of high interest to any Bank
    - **Regression Problem:** When you are interested in answering how much – these problems would fall under the Regression umbrella. For example – how much cleaning needs to be done is a Regression problem. Or what is the expected amount of default from a customer is a Regression problem
- **Unsupervised Machine Learning:** There are times when you don't want to exactly predict an Outcome. You just want to perform a segmentation or clustering. For example – a bank would want to have a segmentation of its customers to understand their behavior. This is an Unsupervised Machine Learning problem as we are not predicting any outcomes here
- **Reinforcement Learning:** Reinforcement Learning is said to be the hope of true artificial intelligence. And it is rightly said so because the potential that Reinforcement Learning possesses is immense. It is a slightly complex topic as compared to traditional machine learning but an equally crucial one for the future. This article is as good an introduction to reinforcement learning as any you will find



What kind of data is required to train a machine learning model?

Data is omnipresent these days. From logs on websites and smartphones to health devices – we are in a constant process of creating data. In fact, 90% of the data in this Universe has been created in the last 18 months.

Data can broadly be classified into two types:

1. **Structured Data:** Structured data typically refers to data stored in a tabular format in databases in organizations. This includes data about customers, interactions with them and several other attributes, which flow through the IT infrastructure of Enterprises
2. **Unstructured Data:** Unstructured Data includes all the data which gets captured, but is not stored in the form of tables in enterprises. For example – letters of communication from customers or tweets and pictures from customers. It also includes images and voice records.

What are the Different algorithms used in Machine Learning?

- **Supervised Learning**
    - Linear Regression
    - Logistic Regression
    - Decision Trees
    - Support Vector Machines (SVM)
- **Unsupervised Learning**
    - k means clustering
    - Hierarchical clustering

What are the steps involved in building machine learning models?

Any machine learning model development can broadly be divided into six steps:

- **Problem definition** involves converting a Business Problem to a machine learning problem
- **Hypothesis generation** is the process of creating a possible business hypothesis and potential features for the model
- **Data Collection** requires you to collect the data for testing your hypothesis and building the model
- **Data Exploration** and cleaning helps you remove outliers, missing values and then **transform** the data into the required format
- Modeling is where you actually build the **machine learning models**
- Once built, you will **deploy the models**

What are some of the Challenges in the ad0ption of Machine Learning?

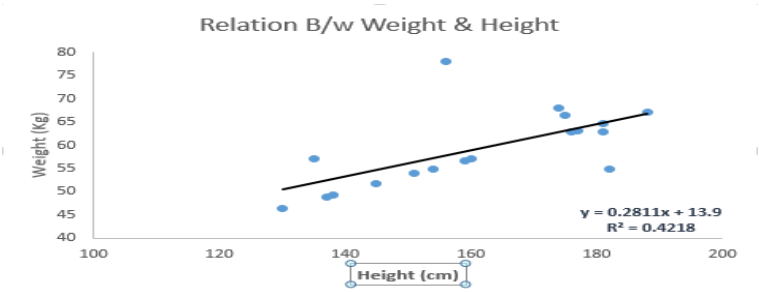
- **Huge data required:** It takes a huge amount of data to train a model today. For example – if you want to classify Cats vs. Dogs based on images (and you don't use an existing model) –
- **High compute required:** As of now, machine learning and deep learning models require huge computations to achieve simple tasks (simple according to humans). This is why the use of special hardware including GPUs and TPUs is required. The cost of computations needs to come down for machine learning to make a next-level impact
- **Interpretation of models is difficult at times:** Some modeling techniques can give us high accuracy but are difficult to explain. This can leave the business owners frustrated. Imagine being a bank, but you cannot tell why you declined a loan for a customer!
- **New and better algorithms required:** Researchers are consistently looking out for new and better algorithms to address some of the problems mentioned above
- **More Data Scientists needed:** Further, since the domain has grown so quickly – there aren't many people with the skill sets required to solve the vast variety of problems. This is expected to remain so for the next few years. So, if you are thinking about building a career in machine learning – you are in good stead!

LINEAR REGRESSION

It is one of the most widely known modeling technique. Linear regression is usually among the first few topics which people pick while learning predictive modeling. In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear.

Linear Regression establishes a relationship between **dependent variable (Y)** and one or more **independent variables (X)** using a **best fit straight line** (also known as regression line).

It is represented by an equation **Y=a+b*X + e**, where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable(s).

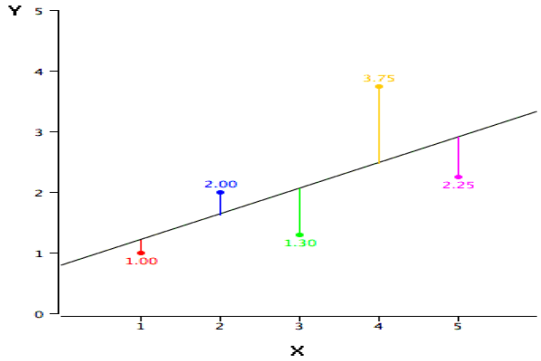


The difference between simple linear regression and multiple linear regression is that, multiple linear regression has (>1) independent variables, whereas simple linear regression has only 1 independent variable. Now, the question is “How do we obtain best fit line?”.

How to obtain best fit line (Value of a and b)?

This task can be easily accomplished by Least Square Method. It is the most common method used for fitting a regression line. It calculates the best-fit line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line. Because the deviations are first squared, when added, there is no cancelling out between positive and negative values.

$$\min_{w} ||Xw - y||_2^2$$



PROBLEM STATEMENT: TO PREDICT THE OPTIMUM TEMPERATION OF PLANT

CODE:

```
"""  Created on Thu DEC  5 17:49:52 2019

@author: SH   """

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression

df =pd.read_excel('D:\MODEL DATA\CLEANED_DATA.xlsx')

df.columns


print(df.shape)


train =df[0:4999]

test =df[5000:]

x_train=train.drop(['FUTURE_IH_TEMP','TIMESTAMP'],axis=1)

y_train=train['FUTURE_IH_TEMP']

x_test=test.drop(['FUTURE_IH_TEMP','TIMESTAMP'],axis=1)

y_test=test['FUTURE_IH_TEMP']

lreg=LinearRegression()

lreg.fit(x_train,y_train)

p=lreg.predict(x_test)

s1=lreg.score(x_train,y_train)

s2=lreg.score(x_test,y_test)

print('current prediction')

print('train score=',s1 )

print('test score=',s2)

percentage = (s2*100)
```

```
percent= float("{0:.2f}".format(percentage))

print(percent,'%')


#Line graph

from  pylab import plot

plot(df['TIMESTAMP'], df['FUTURE_IH_TEMP'])

xlabel('TIMESTAMP')

ylabel('FUTURE_IH_TEMP')

title('Python Line Chart: Plotting numbers')

grid(True)

show()
```

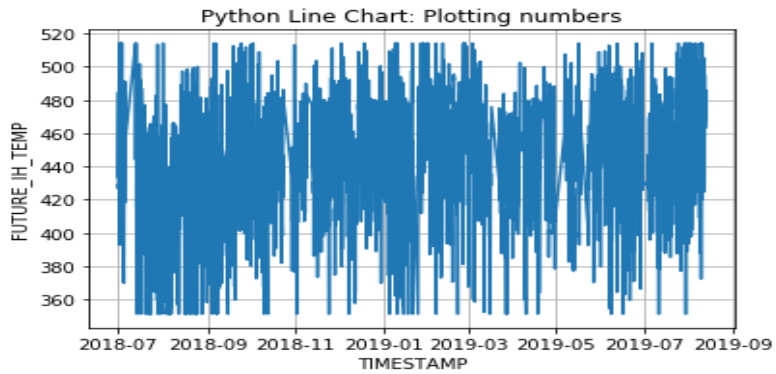
RESULT:

(7668, 87)

train score= 0.9943461886275639

test score= 0.9932157427648388

99.24 %



DECISION TREE

Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) that is mostly used in classification problems. It works for both categorical and continuous input and output variables. In this technique, we split the population or sample into two or more homogeneous sets (or sub-populations) based on most significant splitter / differentiator in input variables.

Types of Decision Tree

Types of decision tree is based on the type of target variable we have. It can be of two types:

1. **Binary Variable Decision Tree:** Decision Tree which has binary target variable then it called as Binary Variable Decision Tree. Example:- In above scenario of student problem, where the target variable was "Student will play cricket or not" i.e. YES or NO.
2. **Continuous Variable Decision Tree:** Decision Tree has continuous target variable then it is called as Continuous Variable Decision Tree.

Terminology related to Decision Trees:

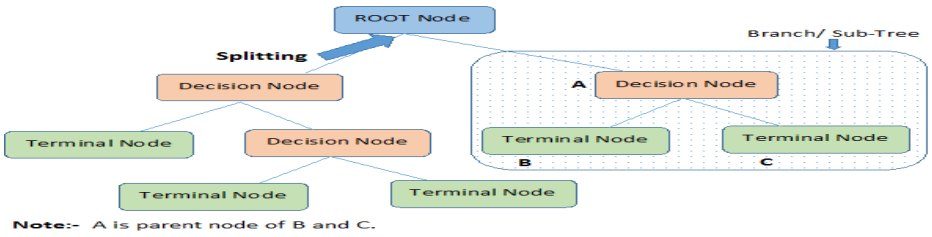
Let's look at the basic terminology used with Decision trees:

ROOT Node: It represents entire population or sample and this further gets divided into two or more homogeneous sets.

SPLITTING: It is a process of dividing a node into two or more sub-nodes.

Decision Node: When a sub-node splits into further sub-nodes, then it is called decision node.

Leaf/ Terminal Node: Nodes do not split is called Leaf or Terminal node.



Pruning: When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.

Branch / Sub-Tree: A sub section of entire tree is called branch or sub-tree

Parent and Child Node: A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node.

PROBLEM STATEMENT: TO PREDICT THE OPTIMUM TEMPERATION OF PLANT

CODE:

```
"""

Created on Tue Dec 10 14:14:47 2019

@author: SH

"""

import pandas as pd

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score

df =pd.read_excel('D:\MODEL DATA\CLEANED_DATA.xlsx')

print(df.shape)

train =df[0:4999]

test =df[5000:]

train_x=train.drop(['FUTURE_IH_TEMP','TIMESTAMP'],axis=1)

train_y=train['FUTURE_IH_TEMP']

test_x=test.drop(['FUTURE_IH_TEMP','TIMESTAMP'],axis=1)

test_y=test['FUTURE_IH_TEMP']

test_y=test_y.astype('int')

train_y=train_y.astype('int')

model = DecisionTreeClassifier()

# fit the model with the training data

model.fit(train_x,train_y)

# depth of the decision tree

print('Depth of the Decision Tree :', model.get_depth())

# predict the target on the train dataset

predict_train = model.predict(train_x)

print('Target on train data',predict_train)
```

```
# Accuray Score on train dataset

accuracy_train = accuracy_score(train_y,predict_train)

print('accuracy_score on train dataset : ', accuracy_train)

# predict the target on the test dataset

predict_test = model.predict(test_x)

print('Target on test data',predict_test)

# Accuracy Score on test dataset

accuracy_test = accuracy_score(test_y,predict_test)

print('accuracy_score on test dataset : ', accuracy_test)
```

RESULT:

(7668, 87)

Depth of the Decision Tree : 56

Target on train data [428 426 457 ... 466 466 452]

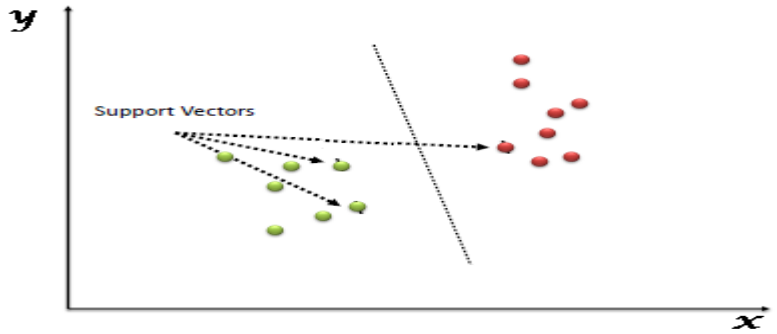
accuracy_score on train dataset : 0.9997999599919984

Target on test data [440 449 425 ... 488 444 501]

accuracy_score on test dataset : 0.02361319340329835

SUPPORT VECTOR MACHINE(SVM)

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot). Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

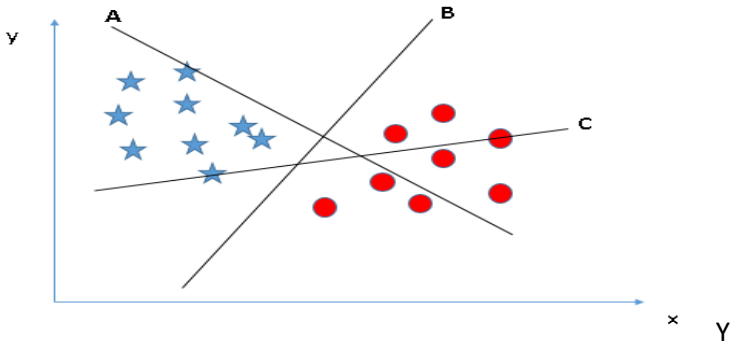


How does it work?

Above, we got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is “How can we identify the right hyper-plane?”. Don’t worry, it’s not as hard as you think!

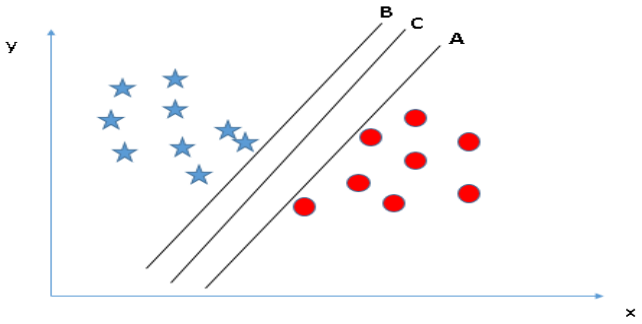
Let’s understand:

- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.

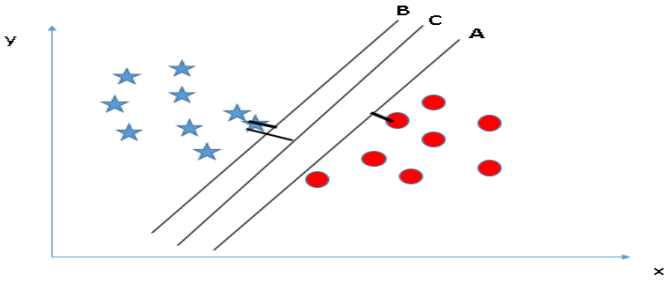


Y

- You need to remember a thumb rule to identify the right hyper-plane: "Select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job.
- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?



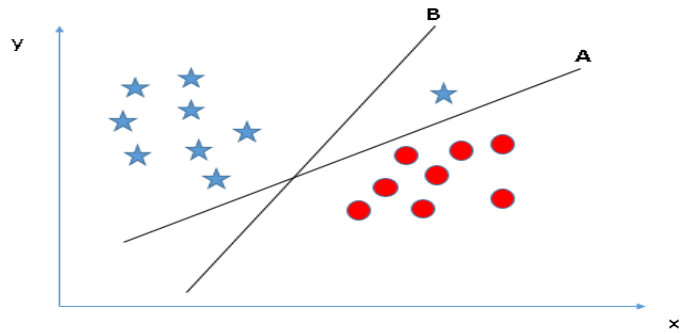
Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let's look at the below snapshot



:

Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

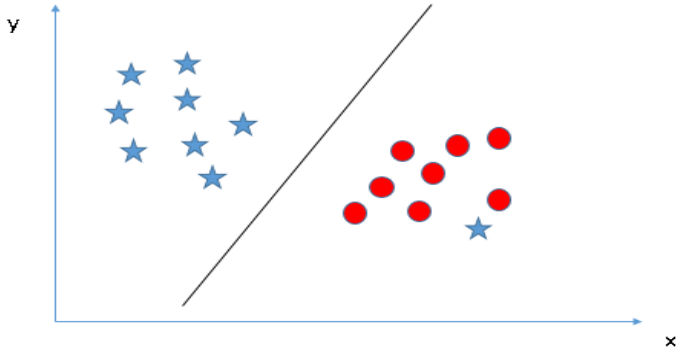
- **Identify the right hyper-plane (Scenario-3):**Hint: Use the rules as discussed in previous section to identify the right hyper-plane



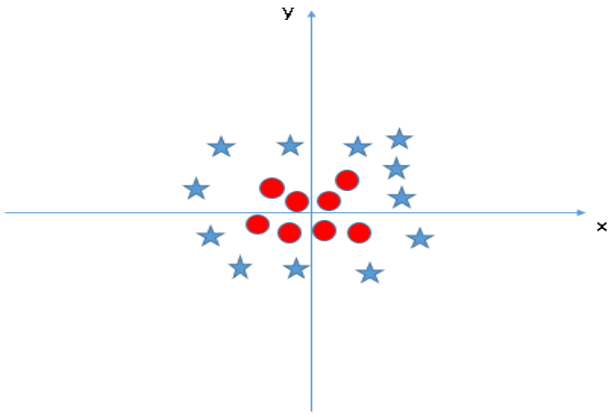
Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A.** But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

- **Can we classify two classes (Scenario-4)?:** Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other(circle) class as an outlier.

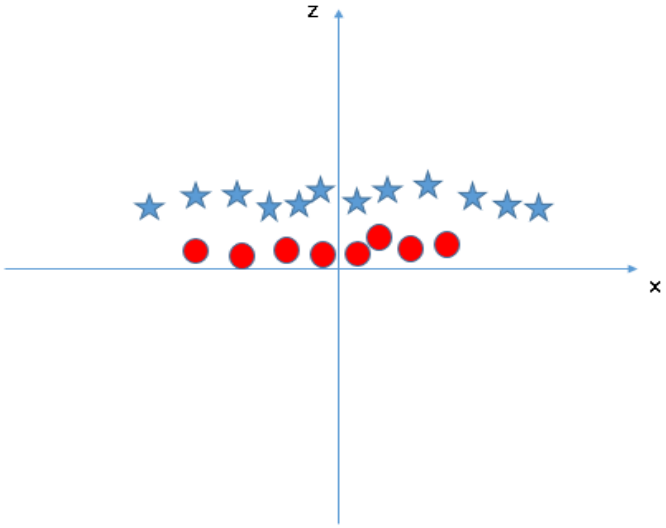
- As I have already mentioned, one star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.



- **Find the hyper-plane to segregate to classes (Scenario-5):** In the scenario below, we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.



- SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature z=x^2+y^2. Now, let's plot the data points on axis x and z:



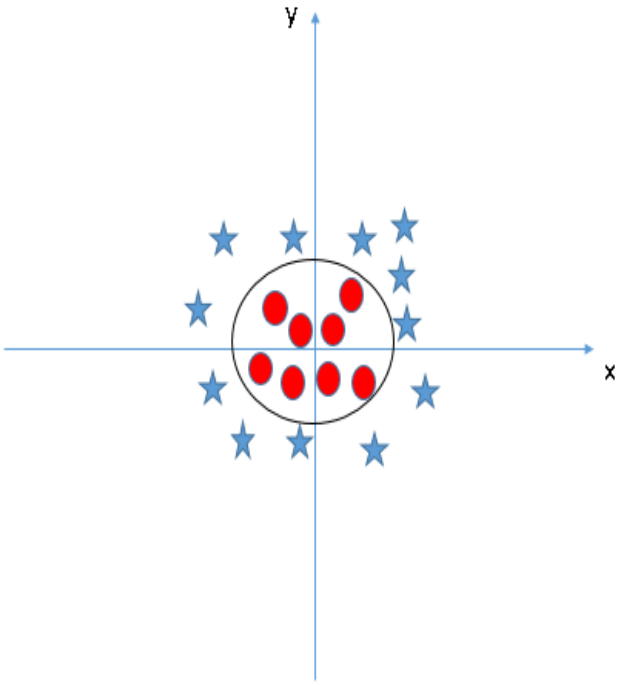
In above plot, points to consider are:

  - All values for z would be positive always because z is the squared sum of both x and y
  - In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

In SVM, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the **kernel trick**. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely

complex data transformations, then find out the process to separate the data based on the labels or outputs you've defined.

- When we look at the hyper-plane in original input space it looks like a circle:



Now, let's look at the methods to apply SVM algorithm in a data science challenge.

PROBLEM STATEMENT: TO PREDICT THE OPTIMUM TEMPERATION OF PLANT

CODE:

```
"""    Created on Mon DEC  16 17:49:52 2019

@author: SH   """

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn import svm

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import accuracy_score

import warnings

warnings.filterwarnings('ignore')

df =pd.read_excel('D:\MODEL DATA\CLEANED_DATA.xlsx')

print(df.shape)

train =df[0:4999]

test =df[5000:]

x_train=train.drop(['FUTURE_IH_TEMP','TIMESTAMP'],axis=1)

y_train=train['FUTURE_IH_TEMP']

x_test=test.drop(['FUTURE_IH_TEMP','TIMESTAMP'],axis=1)

y_test=test['FUTURE_IH_TEMP']

y_test=y_test.astype('int')

y_train=y_train.astype('int')

sc= StandardScaler()

x_train=sc.fit_transform(x_train)

x_test=sc.transform(x_test)
```

```
support = svm.SVC(random_state=0)


# Train the model using the training sets and check score on test dataset
support.fit(x_train, y_train)
predicted= support.predict(x_test)
score=accuracy_score(y_test,predicted)
print("Your Model Accuracy is", score)
train.to_csv( "pred.csv")
```

RESULT:

```
(7668, 87)
Your Model Accuracy is 0.01911544227886057
```

CLUSTERING

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the aim is to segregate groups with similar traits and assign them into clusters.

2. Types of Clustering

Broadly speaking, clustering can be divided into two subgroups :

- **Hard Clustering:** In hard clustering, each data point either belongs to a cluster completely or not. For example, in the above example each customer is put into one group out of the 10 groups.
- **Soft Clustering**: In soft clustering, instead of putting each data point into a separate cluster, a probability or likelihood of that data point to be in those clusters is assigned. For example, from the above scenario each costumer is assigned a probability to be in either of 10 clusters of the retail store.

3. Types of clustering algorithms

Since the task of clustering is subjective, the means that can be used for achieving this goal are plenty. Every methodology follows a different set of rules for defining the '*similarity'* among data points. In fact, there are more than 100 clustering algorithms known. But few of the algorithms are used popularly, let's look at them in detail:

- **Connectivity models:** As the name suggests, these models are based on the notion that the data points closer in data space exhibit more similarity to each other than the data points lying farther away. These models can follow two approaches. In the first approach, they start with classifying all data points into separate clusters & then aggregating them as the distance decreases. In the second approach, all data points are classified as a single cluster and then partitioned as the distance increases. Also, the choice of distance function is subjective. These models are very easy to interpret but lacks scalability for handling big datasets. Examples of these models are hierarchical clustering algorithm and its variants.

- **Centroid models:** These are iterative clustering algorithms in which the notion of similarity is derived by the closeness of a data point to the centroid of the clusters. K-Means clustering algorithm is a popular algorithm that falls into this category. In these models, the no. of clusters required at the end

have to be mentioned beforehand, which makes it important to have prior knowledge of the dataset. These models run iteratively to find the local optima.

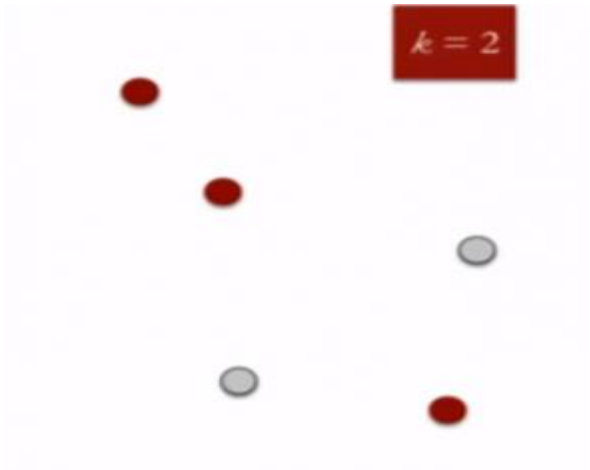
- **Distribution models:** These clustering models are based on the notion of how probable is it that all data points in the cluster belong to the same distribution (For example: Normal, Gaussian). These models often suffer from overfitting. A popular example of these models is Expectation-maximization algorithm which uses multivariate normal distributions.

- **Density Models:** These models search the data space for areas of varied density of data points in the data space. It isolates various different density regions and assign the data points within these regions in the same cluster. Popular examples of density models are DBSCAN and OPTICS.

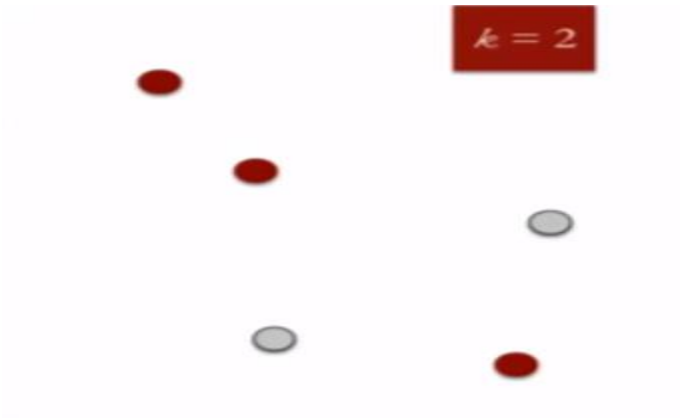
K Means Clustering

K means is an iterative clustering algorithm that aims to find local maxima in each iteration. This algorithm works in these 5 steps :

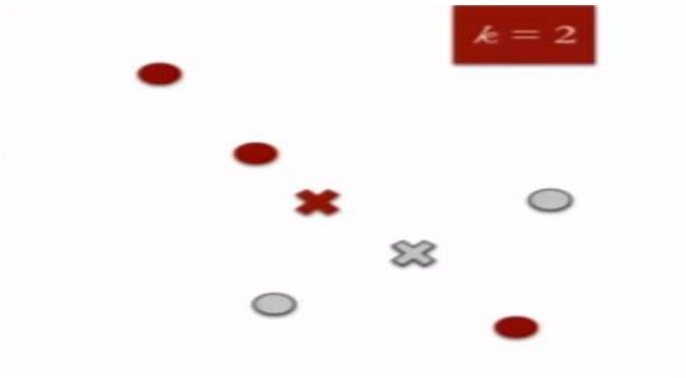
1. Specify the desired number of clusters K : Let us choose k=2 for these 5 data points in 2-D space.



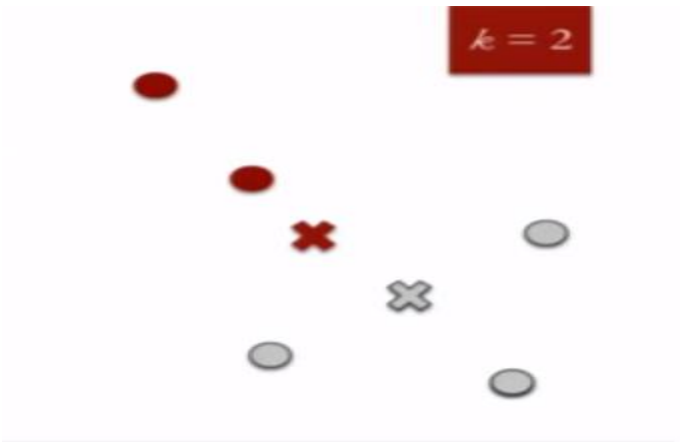
2. Randomly assign each data point to a cluster : Let’s assign three points in cluster 1 shown using red color and two points in cluster 2 shown using grey color.



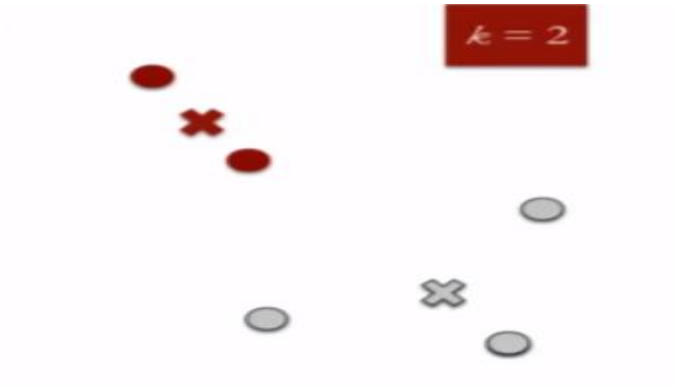
3. Compute cluster centroids : The centroid of data points in the red cluster is shown using red cross and those in grey cluster using grey cross.



4. Re-assign each point to the closest cluster centroid : Note that only the data point at the bottom is assigned to the red cluster even though its closer to the centroid of grey cluster. Thus, we assign that data point into grey cluster



5. Re-compute cluster centroids : Now, re-computing the centroids for both the clusters.

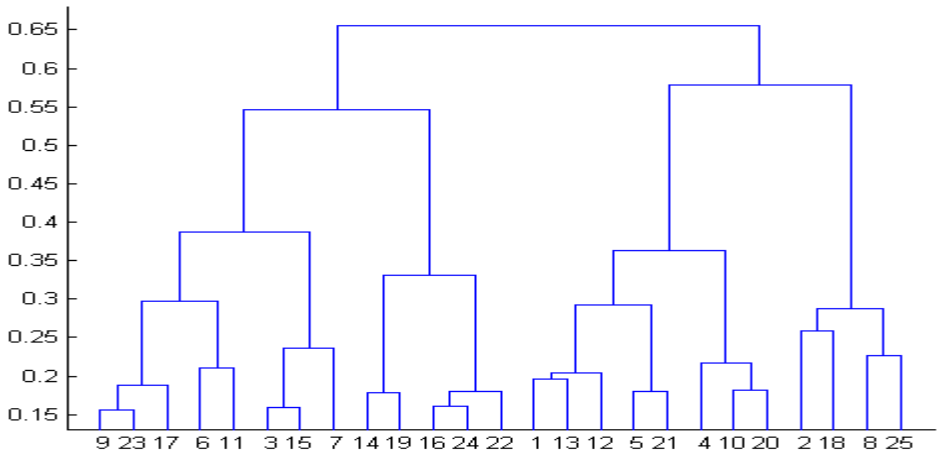


6. Repeat steps 4 and 5 until no improvements are possible : Similarly, we'll repeat the 4th and 5th steps until we'll reach global optima. When there will be no further switching of data points between two clusters for two successive repeats. It will mark the termination of the algorithm if not explicitly mentioned.

Hierarchical Clustering

Hierarchical clustering, as the name suggests is an algorithm that builds hierarchy of clusters. This algorithm starts with all the data points assigned to a cluster of their own. Then two nearest clusters are merged into the same cluster. In the end, this algorithm terminates when there is only a single cluster left.

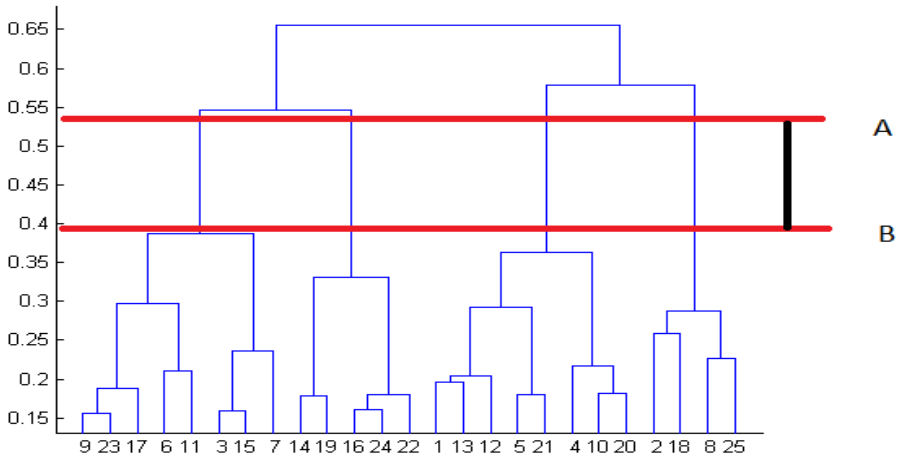
The results of hierarchical clustering can be shown using dendrogram. The dendrogram can be interpreted as:



At the bottom, we start with 25 data points, each assigned to separate clusters. Two closest clusters are then merged till we have just one cluster at the top. The height in the dendrogram at which two clusters are merged represents the distance between two clusters in the data space.

The decision of the no. of clusters that can best depict different groups can be chosen by observing the dendrogram. The best choice of the no. of clusters is the no. of vertical lines in the dendrogram cut by a horizontal line that can transverse the maximum distance vertically without intersecting a cluster.

In the above example, the best choice of no. of clusters will be 4 as the red horizontal line in the dendrogram below covers maximum vertical distance AB.



Two important things that you should know about hierarchical clustering are:

- This algorithm has been implemented above using bottom up approach. It is also possible to follow top-down approach starting with all data points assigned in the same cluster and recursively performing splits till each data point is assigned a separate cluster.
- The decision of merging two clusters is taken on the basis of closeness of these clusters. There are multiple metrics for deciding the closeness of two clusters :
    - Euclidean distance: $||a-b||_2 = \sqrt{(\Sigma(a_i-b_i))}$
    - Squared Euclidean distance: $||a-b||_2^2 = \Sigma((a_i-b_i)^2)$
    - Manhattan distance: $||a-b||_1 = \Sigma|a_i-b_i|$
    - Maximum distance:$||a-b||_{INFINITY} = max_i|a_i-b_i|$
    - Mahalanobis distance: $\sqrt{((a-b)^T S^{-1} (-b))}$ {where, s : covariance matrix}

Difference between K Means and Hierarchical clustering

- Hierarchical clustering can't handle big data well but K Means clustering can. This is because the time complexity of K Means is linear i.e. O(n) while that of hierarchical clustering is quadratic i.e. $O(n^2)$.
- In K Means clustering, since we start with random choice of clusters, the results produced by running the algorithm multiple times might differ. While results are reproducible in Hierarchical clustering.
- K Means is found to work well when the shape of the clusters is hyper spherical (like circle in 2D, sphere in 3D).
- K Means clustering requires prior knowledge of K i.e. no. of clusters you want to divide your data into. But, you can stop at whatever number of clusters you find appropriate in hierarchical clustering by interpreting the dendrogram

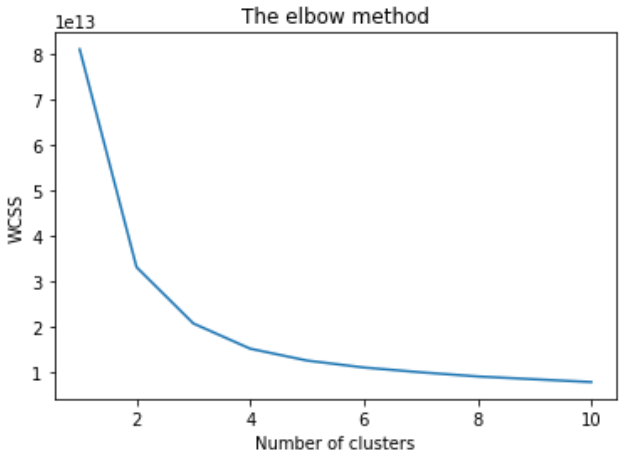
PROBLEM DEFINATION: TO PREDICT THE OPTIMUM TEMPERATION OF PLANT

CODE:

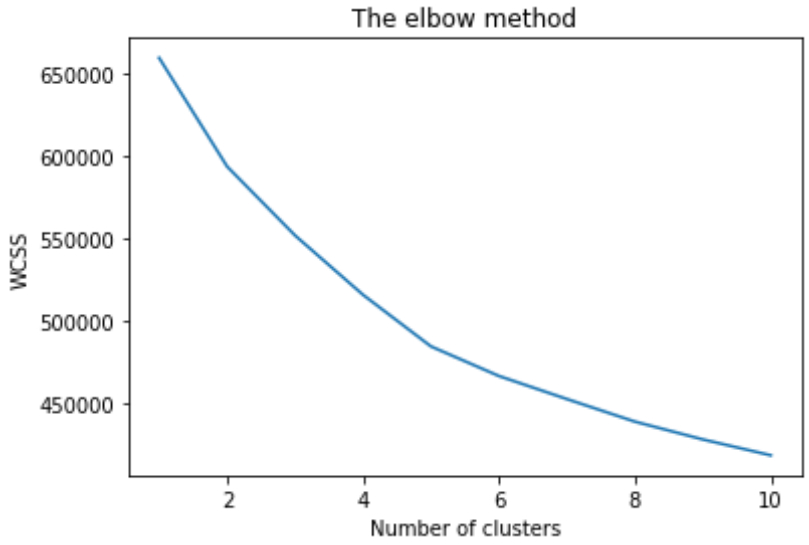
```
"""    Created on Fri DEC  20 15:25:52 2019

@author: SH   """

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler

df = pd.read_excel('D:\MODEL DATA\CLEANED_DATA.xlsx')

df.head()

x = df.drop(['TIMESTAMP'],axis=1)

x.shape

wcss = []

#elbow curve

for i in range(1, 11):

    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)

    kmeans.fit(x)

    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)

plt.title('The elbow method')

plt.xlabel('Number of clusters')

plt.ylabel('WCSS') #within cluster sum of squares

plt.show()

#using Standerd Scaler
```

```
scaler=StandardScaler()

sse=[]

Data_Scaled=scaler.fit_transform(x)

for i in range(1, 11):

    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)

    kmeans.fit(Data_Scaled)

    sse.append(kmeans.inertia_)

plt.plot(range(1, 11), sse)

plt.title('The elbow method')

plt.xlabel('Number of clusters')

plt.ylabel('WCSS') #within cluster sum of squares

plt.show()

kmeans = KMeans(n_clusters = 2, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)

y_kmeans = kmeans.fit_predict(x)

print('The score of the model is ',kmeans.inertia_)
```

RESULT:



USING STANDARD SCALER



CLUSTER =2

The score of the model is 33074987192453.223

FINAL MODEL

PROBLEM DEFINATION: TO PREDICT THE OPTIMUM TEMPERATION OF PLANT

CODE:

```
# -*- coding: utf-8 -*-
"""
Created on Mon DEC  30 14:28:24 2019

@author: SH
"""
print('STARTING.....................')

print('PACKAGE LOADING.....................')
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import warnings
warnings.filterwarnings("ignore")
import datetime
import seaborn as sns

columns = ['TIMESTAMP',
'IH_COMBUSTION_AIR_TEMP',
'MACHINE_SPEED',
'SINTER_TEMP_SM_DISCH',
'CO',
'O2',
'COOLER_SPEED',
'TOTAL_BMIX_FLOW',
```

```
'TOTAL_CLIME_FLOW',
'TOTAL_LIMESTONE_FLOW',
'TOTAL_RFINES_FLOW',
'TOTAL_SOLID_FUEL_FLOW',
'TEMP_AFTER_ESP',
'WINDBOX_10_TEMP',
                'WINDBOX_11_TEMP',
                'WINDBOX_12_TEMP',
                'WINDBOX_13_TEMP',
                'WINDBOX_14_TEMP',
                'WINDBOX_15B_TEMP',
                'WINDBOX_15A_TEMP',
                'WINDBOX_16A_TEMP',
                'WINDBOX_16B_TEMP',
                'WINDBOX_17A_TEMP',
                'WINDBOX_17B_TEMP',
                'WINDBOX_3_TEMP',
                'WINDBOX_4_TEMP',
               # 'WINDBOX_5_TEMP',
                #'WINDBOX_6_TEMP',
                'WINDBOX_7_TEMP',
                'WINDBOX_8_TEMP',
                'WINDBOX_9_TEMP',
                'WINDBOX_1_TEMP',
                'WINDBOX_2_TEMP']


SP4_DATA = pd.read_excel('SP4data.xlsx')


SP4_DATA = SP4_DATA[columns]
```

```
DATA_all = SP4_DATA


WINDBOX = DATA_all.loc[:,['WINDBOX_10_TEMP',
                'WINDBOX_11_TEMP',
                'WINDBOX_12_TEMP',
                'WINDBOX_13_TEMP',
                'WINDBOX_14_TEMP',
                'WINDBOX_15B_TEMP',
                'WINDBOX_15A_TEMP',
                'WINDBOX_16A_TEMP',
                'WINDBOX_16B_TEMP',
                'WINDBOX_17A_TEMP',
                'WINDBOX_17B_TEMP',
                'WINDBOX_3_TEMP',
                'WINDBOX_4_TEMP',
                #'WINDBOX_5_TEMP',
                #'WINDBOX_6_TEMP',
                'WINDBOX_7_TEMP',
                'WINDBOX_8_TEMP',
                'WINDBOX_9_TEMP',
                'WINDBOX_1_TEMP',
                'WINDBOX_2_TEMP']]


DATA_all['BTP_TEMP'] = WINDBOX.max(axis=1)


###############################################################################
DATA_all['total_feed_flow'] = DATA_all[['TOTAL_BMIX_FLOW',
                        'TOTAL_CLIME_FLOW',
                        'TOTAL_LIMESTONE_FLOW',
                        'TOTAL_RFINES_FLOW',
                        'TOTAL_SOLID_FUEL_FLOW']].sum(axis=1,skipna = True)
```

```
DATA_all['clime_percent'] = (DATA_all['TOTAL_CLIME_FLOW']/DATA_all['TOTAL_BMIX_FLOW'])*100

DATA_all['limestone_percent'] = (DATA_all['TOTAL_LIMESTONE_FLOW']/DATA_all['TOTAL_BMIX_FLOW'])*100

DATA_all['rfines_percent'] = (DATA_all['TOTAL_RFINES_FLOW']/DATA_all['TOTAL_BMIX_FLOW'])*100

DATA_all['sf_percent'] = (DATA_all['TOTAL_SOLID_FUEL_FLOW']/DATA_all['TOTAL_BMIX_FLOW'])*100

DATA_all['flux_percent'] = (DATA_all[['TOTAL_CLIME_FLOW','TOTAL_LIMESTONE_FLOW']].sum(axis=1)/DATA_all['TOTAL_BMIX_FLOW'])*100


#################### Exclude turn up and shut down records ####################

###############################################################################

temp = DATA_all

temp['row_num'] = range(0,len(temp))

temp['row_num'] += 1


temp_zero = temp.loc[temp['MACHINE_SPEED'] <= 0,]

temp_good = temp.loc[temp['MACHINE_SPEED'] > 2.4,]


temp_zero.shape, temp_good.shape #(0, 23), (8465, 23)

temp_good.loc[:,'row_num_next'] = temp_good['row_num'].shift(-1)

temp_good.loc[:,'row_num_next'] = temp_good['row_num_next'].fillna(0).astype(int)

temp_good['zero_flag'] = -99999


zero_flag = []

for i in range(0,len(temp_good)-1):

    print('#####################################')

    #print(i)

    sq = pd.Series(np.arange(temp_good.iloc[i,].row_num,temp_good.iloc[i,].row_num_next,1))

    zero_flag.append(sum(sq.isin(temp_zero['row_num'])))



zero_flag.append(-99999)
```

```
temp_good['zero_flag'] =  zero_flag


temp_good_sub = temp_good.loc[temp_good['zero_flag']>0,]


hala = []
for i in range(0,len(temp_good_sub)):
   print('########################################')
   sq1 = np.arange((temp_good_sub.iloc[i,].row_num + 1),temp_good_sub.iloc[i,].row_num_next,1)
   print(sq1.shape)
   #print(type(sq1))
   hala.extend(sq1)
   #print(hala)


len(hala) #1245
sum(temp_good_sub['row_num_next']-temp_good_sub['row_num']) #1245 + 81(len temp_good_sub)


temp = temp[~temp.row_num.isin(hala)]
temp.shape #(8216, 63)
temp = temp.drop(['row_num'],axis=1)
temp.shape #(9240, 38)


################################################################################
base_data_MS = temp
base_data_MS.shape #(8216, 62)
base_data_BTP_TEMP = base_data_MS[((base_data_MS.BTP_TEMP >= 350) & (base_data_MS.BTP_TEMP <=
510))]
base_data_BTP_TEMP.shape #(7215,62)
#base_data_TOTAL_CR = base_data_BTP_TEMP[(base_data_BTP_TEMP.CRATE_SP4_GSN > 40)]
#base_data_TOTAL_CR.shape #(7108, 62)
```

```
#Missing Value Treatmenet
DATA2 = base_data_BTP_TEMP.dropna(axis=0)
DATA2.shape #(6143, 62)
desc1 = DATA2.describe()
#Outlier Treatment
Float = DATA2.select_dtypes(include = 'float64')
low = .01
high = .99
percentile = Float.quantile([low,high])

LOW_OL = Float.apply(lambda x: x.where(x>percentile.loc[low,x.name],percentile.loc[low,x.name]))
ALL_OL = LOW_OL.apply(lambda x: x.where(x<percentile.loc[high,x.name],percentile.loc[high,x.name]))

raw_data = pd.concat([DATA2.loc[:,['TIMESTAMP']], ALL_OL], axis=1)

desc2 = raw_data.describe()

raw_data.shape #(8991, 31)

q1 = raw_data['CO'].quantile(0.25)
q3 = raw_data['CO'].quantile(0.75)
IQR = q3-q1

raw_data = raw_data.query('(@q1 - 3*@IQR) <= CO <= (@q3 + 3*@IQR)')

desc3 = raw_data.describe()

###############################################################################

lag = 1 #buffer time #change
raw_data['TIMESTAMP'] = pd.to_datetime(raw_data['TIMESTAMP'])
```

```
raw_data['TIMESTAMP_NEXT'] = raw_data['TIMESTAMP'] + datetime.timedelta(hours = lag)

IH_COMB_AIR_TEMP = raw_data[['TIMESTAMP','IH_COMBUSTION_AIR_TEMP']]
IH_COMB_AIR_TEMP.columns = ['TIMESTAMP_NEXT','FUTURE_IH_TEMP']
raw_data = pd.merge(raw_data,IH_COMB_AIR_TEMP, on = 'TIMESTAMP_NEXT', how = 'left')



abs(raw_data.corr()['FUTURE_IH_TEMP']).sort_values()

DATA = raw_data[raw_data['FUTURE_IH_TEMP'] > 0]
DATA.shape #(7403, 40)



################################################################################
######################################### MODEL ##################################

RESPONSE =  ['FUTURE_IH_TEMP']
PREDICTOR = ['SINTER_TEMP_SM_DISCH'
        ,'CO'
        #,'O2'
        #,'IH_COMBUSTION_AIR_FLOW'
        #,'IH_COMBUSTION_AIR_PRESSURE'
        #,'IH_PRESSURE'
        #,'MACHINE_SPEED'
        ,'COOLER_SPEED'
        #,'clime_percent'
        #,'limestone_percent'
        ,'total_feed_flow'
        ,'flux_percent'
        ,'rfines_percent'
        ,'sf_percent'
```

```
        #,'RMBBN'
        #,'TEMP_AFTER_ESP'
        #,'CRATE_SP4_GSN'
        #,'CRATE_PSW_RMBBN'
        #,'CRATE_CB_RMBBN'
        #,'CRATE_CB_SP4'
        #,'SUCTION_ESP_OUTLET'
        ,'BTP_TEMP'
         ]

mDATA = DATA[RESPONSE+PREDICTOR]
mDATA.index = DATA['TIMESTAMP']
mDATA = mDATA.dropna(axis=0)

#chk
corr = mDATA.corr()
plt.figure(figsize=(10,15))
sns.heatmap(corr,fmt=".2f",annot=True,cmap="YlGnBu")
plt.show()
corr_2hours = abs(corr['FUTURE_IH_TEMP']).sort_values()

print(min(mDATA.index))
print(max(mDATA.index))



################################################################################
#                      TRAIN TEST SPLIT                        #
################################################################################
train = mDATA[mDATA.index < '2019-07-20 00:00:00'].copy()
test = mDATA[mDATA.index >= '2019-07-20 00:00:00'].copy()
print(train.shape)
```

```
print(test.shape)

############################SCALING#################################################
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler().fit(train[PREDICTOR].values)

X_train_df = train[PREDICTOR]
X_train = scaler.transform(X_train_df.copy())
y_train_df = train[RESPONSE]

from scipy import stats
y_train,fitted_lambda = stats.boxcox(y_train_df)
scalerY = StandardScaler().fit(y_train)
y_train = scalerY.transform(y_train)

def de_boxcox(np_array,fitted_lambda):
    return np.power((fitted_lambda * np_array + 1 ),(1/fitted_lambda))

#y_train = np.array(y_train_df)
X_test_df = test[PREDICTOR].copy()
X_test = scaler.transform(X_test_df.copy())


JULY = X_test_df[(X_test_df.index >= '2019-07-01 00:00:00')].copy()
scaler_july = StandardScaler().fit(JULY.values)
JULY_test = scaler_july.transform(JULY.copy())


y_test_df = test[RESPONSE]
y_test = scalerY.transform(stats.boxcox(y_test_df , fitted_lambda))
```

```
#y_test = np.array(y_test_df)


fig, ax=plt.subplots(1,2)
sns.distplot(y_train, ax=ax[0])
sns.distplot(y_test, ax=ax[1])


print('')
print('********** DATA SPLIT SHAPE **********')
print('')
print('Training2 Features Shape:',X_train.shape)
print('Training2 Labels Shape:',y_train.shape)
print('Test Features Shape:',X_test.shape)
print('Test Labels Shape:',y_test.shape)
print('')
print('****************************************')


############################## MODEL FITTING ###################################
from sklearn.linear_model import LinearRegression

# Fit the train data
regression_model = LinearRegression()
regression_model.fit(X_train, y_train)



############################## MODEL EVALUATION ################################
train_eva = train[['FUTURE_IH_TEMP']].copy()
test_eva = test[['FUTURE_IH_TEMP']].copy()



train_eva['PRED_SINGLE'] =
de_boxcox(scalerY.inverse_transform(regression_model.predict(X_train)),fitted_lambda)
```

```
test_eva['PRED_SINGLE'] = de_boxcox(scalerY.inverse_transform(regression_model.predict(X_test)),fitted_lambda)


def MAPE_MAE(df,actual,pred):
    #Calculate MAE & MAPE
    MAE = np.mean(abs(df[actual] - df[pred]))
    MAPE = np.mean(100*(abs(df[actual]-df[pred])/df[actual]))
    return MAPE,MAE


print('')
print('********** TRAINING MODEL OUTPUT **********')
print('')
train_pred_single_MAPE, train_pred_single_MAE = MAPE_MAE(train_eva,'FUTURE_IH_TEMP','PRED_SINGLE')
print('Training Model MAE is:',round(train_pred_single_MAE,2))
print('Training Model MAPE is:',round(train_pred_single_MAPE,2))
print('')
print('*********************************************')
print('')
print('********** TESTING MODEL OUTPUT ***********')
print('')
test_pred_single_MAPE, test_pred_single_MAE = MAPE_MAE(test_eva,'FUTURE_IH_TEMP','PRED_SINGLE')
print('Test Single Model MAE is:',round(test_pred_single_MAE,2))
print('Test Single Model MAPE is:',round(test_pred_single_MAPE,2))
print('')
print('*********************************************')
print('')


###############################################################################
################################## LINEAR MODEL ################################
X_train_LM = pd.DataFrame(X_train,columns=X_train_df.columns,index = X_train_df.index)
import statsmodels.api as sm
```

```
#X_train_c =  sm.add_constant(X_train_LM)
X_train_c = X_train_LM.copy()
X_train_c['const'] =  1
sm_model = sm.OLS(y_train,X_train_c).fit()

print_model = sm_model.summary()
print(print_model)

################################################################################
############################### MODEL EVA GRAPH ################################
test_eva_july = test_eva[(test_eva.index >= '2019-07-20 00:00:00')].copy()
test_eva_july['PRED_SINGLE'] =
de_boxcox(scalerY.inverse_transform(regression_model.predict(JULY_test)),fitted_lambda)

test_eva2 = test_eva_july[(test_eva_july.index >= '2019-07-20 00:00:00') &
            (test_eva_july.index <= '2019-07-24 00:00:00')].copy()
plt.figure(figsize=(15,6))
plt.plot(np.array(range(len(test_eva2))),test_eva2['FUTURE_IH_TEMP'],
    #marker = 'o',markerfacecolor='blue', markersize=8,
    linewidth=3,linestyle = '-', color = 'blue')
plt.plot(np.array(range(len(test_eva2))),test_eva2['PRED_SINGLE'],
    color = 'red',linewidth=1,linestyle = '--',
    marker = 'o',markersize=8,markerfacecolor='yellow'
    )
plt.legend()
plt.show()


MAPE_JULY, MAE_JULY = MAPE_MAE(test_eva2,'FUTURE_IH_TEMP','PRED_SINGLE')

print('')
```

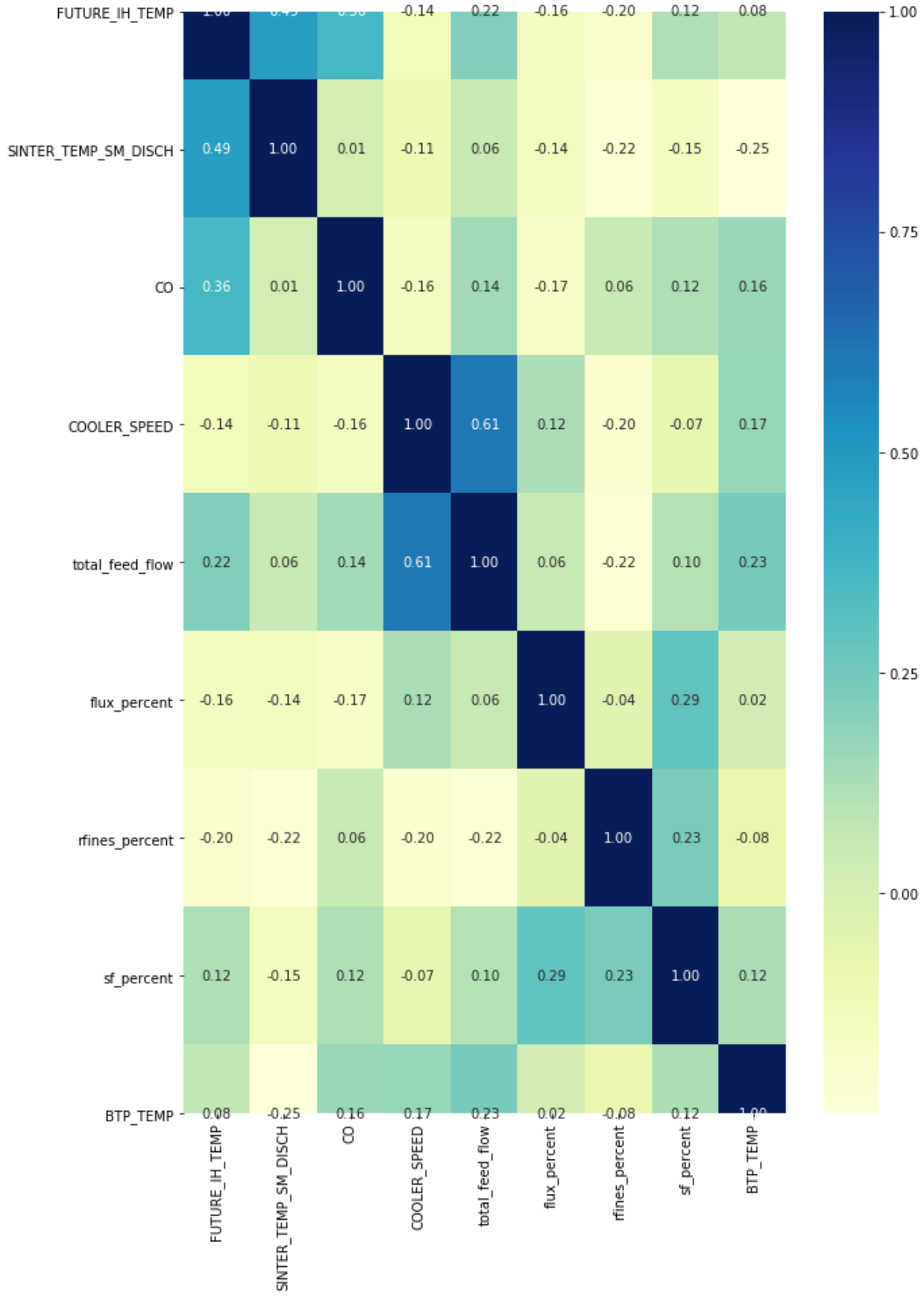
```
print('*******************************************')
print('************** USING JULY RANGE *************')
print('')
print('July MAPE :',MAPE_JULY)
print('July MAE :',MAE_JULY)

test_eva1 = test_eva[(test_eva.index >= '2019-07-20 00:00:00') &
                (test_eva.index <= '2019-07-24 00:00:00')].copy()
#test resul graph
plt.figure(figsize=(15,6))
plt.plot(np.array(range(len(test_eva1))),test_eva1['FUTURE_IH_TEMP'],
       #marker = 'o',markerfacecolor='blue', markersize=8,
       linewidth=3,linestyle = '-', color = 'blue')
plt.plot(np.array(range(len(test_eva1))),test_eva1['PRED_SINGLE'],
       color = 'red',linewidth=1,linestyle = '--',
       marker = 'o',markersize=8,markerfacecolor='yellow'
       )
plt.legend()
plt.show()

MAPE_JUNE, MAE_JUNE = MAPE_MAE(test_eva1,'FUTURE_IH_TEMP','PRED_SINGLE')

print('')
print('*******************************************')
print('**** TESTING ERROR USING OVERALL RANGE ******')
print('')
print('')
print('MAPE :',MAPE_JUNE)
print('MAE :',MAE_JUNE)
#############################################################################
plt.plot(np.array(range(len(X_test_df))),X_test_df['total_feed_flow'],
```

```
        #marker = 'o',markerfacecolor='blue', markersize=8,
        linewidth=3,linestyle = '-', color = 'blue')


sns.distplot(mDATA['FUTURE_IH_TEMP'], hist=True, kde=True,
            bins=int(180/5), color = 'darkblue',
            hist_kws={'edgecolor':'black'},
            kde_kws={'linewidth': 4})
plt.show()


JUNE = mDATA[(mDATA.index >= '2019-06-01 00:00:00') &
                (mDATA.index <= '2019-06-30 00:00:00')].copy()

JULY =  mDATA[(mDATA.index >= '2019-07-01 00:00:00') &
                (mDATA.index <= '2019-07-30 00:00:00')].copy()


JUNE_DESC = JUNE.describe()
JUNE_DESC.to_excel('JUNE_DESC.xlsx')


JULY_DESC = JULY.describe()
JULY_DESC.to_excel('JULY_DESC.xlsx')
```

RESULT:

STARTING.....

PACKAGE LOADING.....


FUTURE_IH_TEMP
SINTER_TEMP_SM_DISCH
CO
COOLER_SPEED
total_feed_flow
flux_percent
rfines_percent
sf_percent
BTP_TEMP
1.00
0.49
0.01
-0.11
0.06
-0.14
-0.22
-0.15
-0.25
0.36
0.01
1.00
-0.16
0.14
-0.17
0.06
0.12
0.16
-0.14
-0.11
-0.16
1.00
0.61
0.12
-0.20
-0.07
0.17
0.22
0.06
0.14
0.61
1.00
0.06
-0.22
0.10
0.23
-0.16
-0.14
-0.17
0.12
0.06
1.00
-0.04
0.29
0.02
-0.20
-0.22
0.06
-0.20
-0.22
-0.04
1.00
0.23
-0.08
0.12
-0.15
0.12
-0.07
0.10
0.29
0.23
1.00
0.12
0.08
-0.25
0.16
0.17
0.23
0.02
-0.08
0.12
1.00
1.00
0.75
0.50
0.25
0.00

2018-07-01 00:00:00

2019-07-24 09:00:00

(7301, 9)

(102, 9)

********** DATA SPLIT SHAPE **********

Training2 Features Shape: (7301, 8)

Training2 Labels Shape: (7301, 1)

Test Features Shape: (102, 8)

Test Labels Shape: (102, 1)

********** TRAINING MODEL OUTPUT **********

Training Model MAE is: 17.26

Training Model MAPE is: 3.96

********** TESTING MODEL OUTPUT **********

Test Single Model MAE is: 54.55

Test Single Model MAPE is: 11.92

OLS Regression Results

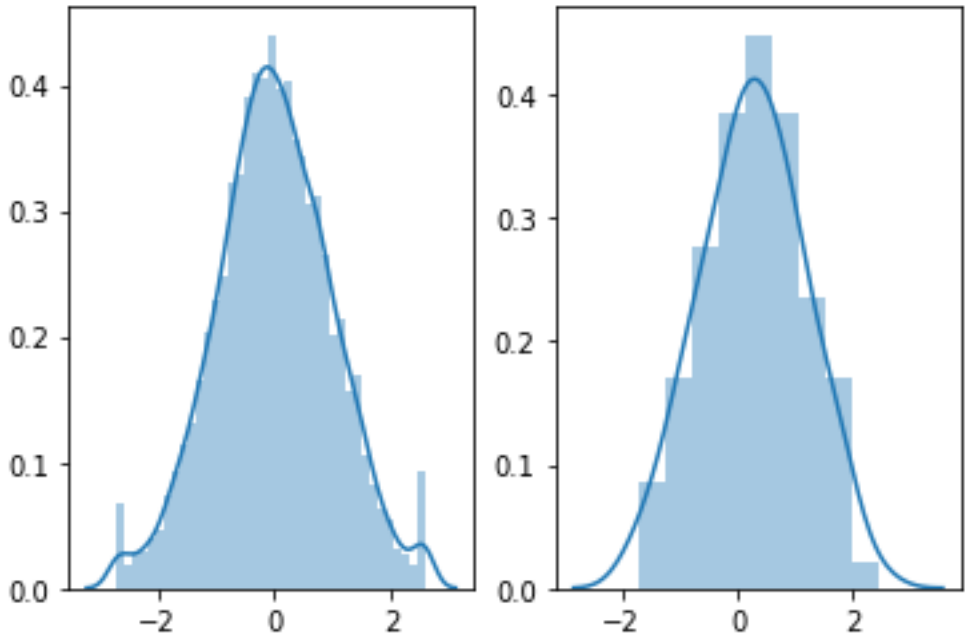
Dep. Variable:	y	R-squared:	0.478
Model:	OLS	Adj. R-squared:	0.477
Method:	Least Squares	F-statistic:	834.3
Date:	Wed, 01 Jan 2020	Prob (F-statistic):	0.00
Time:	19:33:02	Log-Likelihood:	-7987.2
No. Observations:	7301	AIC:	1.599e+04
Df Residuals:	7292	BIC:	1.605e+04
Df Model:	8		
Covariance Type:	nonrobust		

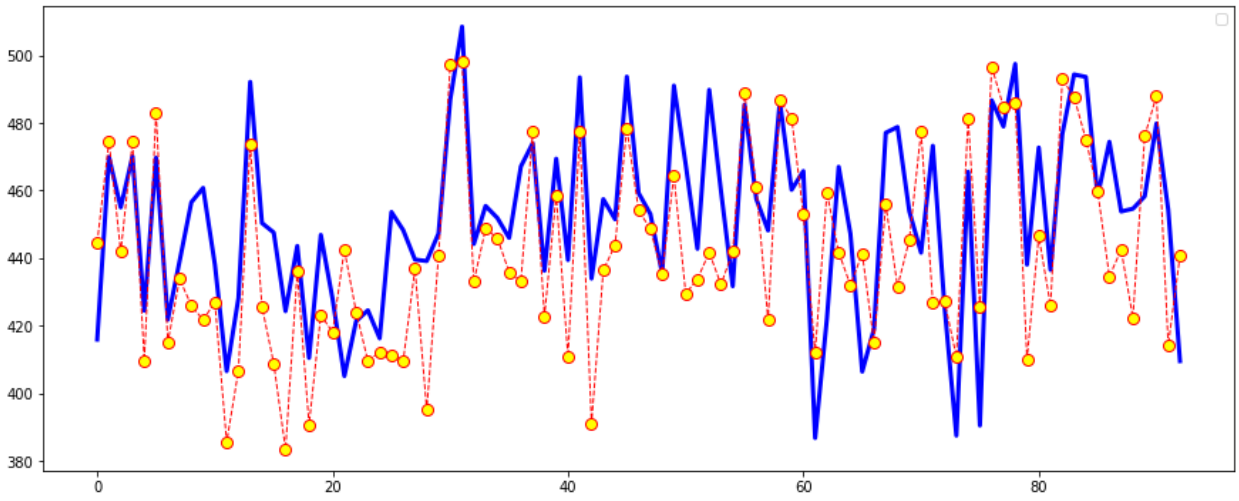
==========

Warnings:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

No handles with labels found to put in legend.



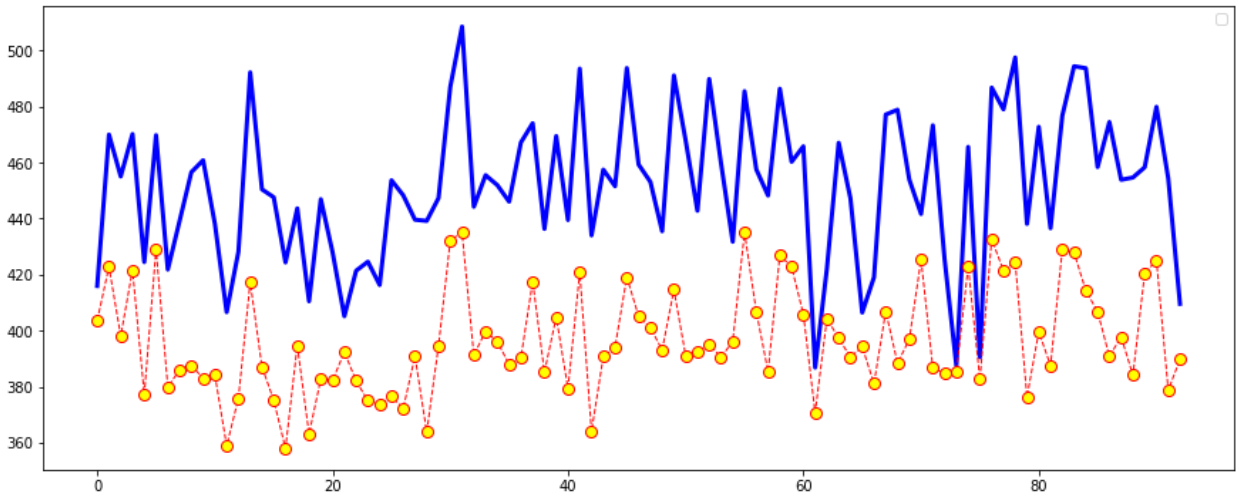


No handles with labels found to put in legend.

********** USING JULY RANGE **********

July MAPE : 4.248197294102037

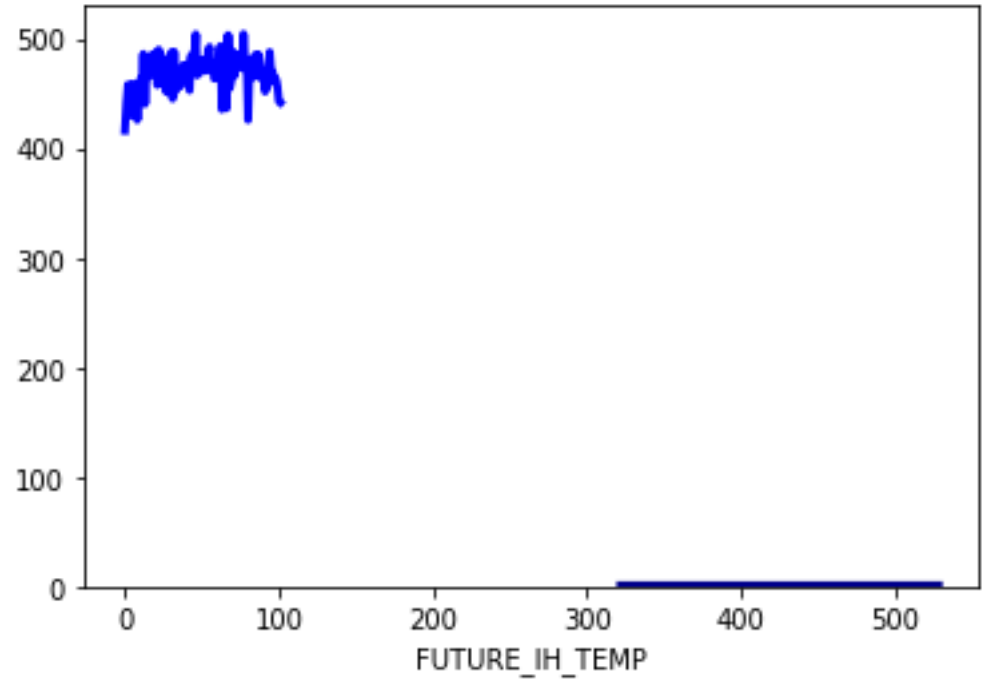
July MAE : 19.02101449702968



**** TESTING ERROR USING OVERALL RANGE ******

MAPE : 12.045181393105283

MAE : 55.00974663368612



CONCLUSION

It was a wonderful learning experience at Tata Steel's Jamshedpur operations site .Throughout my training I visited various systems within the plant specially the LD Shop area and observed various new technologies that are present in the industry .It was indeed a very good learning experience. I gained a lot of insight about the various processes of primary steel making and overall steel production. This training will go a long way in paving my way for future endeavours. This training provided me an opportunity to witness the practical implementation of things which I have studied in my college. It also gave me a chance to see such a huge plant from grass rootlevel.

BIBLIOGRAPHY

- **WEBSITES**
    - **www.google.com**
    - **www.wikipedia.com**
    - **www.analyticsvidhya.com**
    - **www.towardsdatascience.com**
    - **www.geeksforgeeks.com**
- **BOOKS**
    - ISLR
    - PYTHON PROGRAMMING