SUMMER INTERNSHIP 2020
TATA STEEL LIMITED

DATA MINING AND ADVANCED ANALYTICS

UTKARSH

SOE , CUSAT

TABLE OF CONTENTS

ABOUT THE COMPANY

Tata Steel Limited, formerly **Tata Iron and Steel Company Limited (TISCO)**, is an Indian multinational steel-making company headquartered in Kolkata, West Bengal, India, and a subsidiary of the Tata Group.

It is one of the top steel producing companies globally with annual crude steel deliveries of 27.5 million tonnes (in FY17), and the second largest steel company in India (measured by domestic production) with an annual capacity of 13 million tonnes after SAIL.

Tata Steel operates in 26 countries with key operations in India, Netherlands and United Kingdom, and employs around 80,500 people. Its largest plant (10 MTPA capacity) is located in Jamshedpur, Jharkhand. In 2007, Tata Steel acquired the UK-based steel maker Corus.

It was ranked 486th in the 2014 Fortune Global 500 ranking of the world's biggest corporations.

It was the seventh most valuable Indian brand of 2013 as per Brand Finance.

In July 2019 Tata Steel Kalinganagar (TSK) has been included in the list of World Economic Forum's (WEF's) Global Lighthouse Network, showing leadership in applying Fourth Industrial Revolution technologies to drive financial and operational impact

Corus in 2007: On 20 October 2006, Tata Steel signed a deal with Anglo-Dutch company, Corus to buy 100% stake at £4.3 billion ($8.1 billion) at 455 pence per share. On 19 November 2006, the Brazilian steel company Companhia Siderúrgica Nacional (CSN) launched a counter offer for Corus at 475 pence per share, valuing it at £4.5 billion. On 11 December 2006, Tata preemptively upped its offer to 500 pence per share, which was within hours trumped by CSN's offer of 515 pence per share, valuing the deal at £4.9 billion. The Corus board promptly recommended both the revised offers to its shareholders. On 31 January 2007, Tata Steel won their bid for Corus after offering 608 pence per share, valuing Corus at £6.7 billion ($12 billion).

PRODUCTS

The steel plant produces:

- Iron
- Soft iron
- Cast iron
- Alloy
- Bearings
- Pipes
- Precision Tubes

They also produce:

a. Locomotive parts
b. Agricultural equipment

c. Machinery, tinplate
d. Cable and wire
e. Rebars
f. Branded products and solutions like Pravesh Doors, Nest-in building structures
g. Tata Astrum
h. Ferromag
i. IBMD
j. Tata Pipes
k. Tata Precision Tube
l. Raw Material

CUSTOMER SEGMENTS

Some of the key segments that company targets in Indian markets are Construction, Automotive, General Engineering and Industrial Products & Agriculture apart from serving other sectors such as Packaging, Consumer Goods etc. Developing new solutions in the emerging sectors has provided a big fillip to expanding the scope of businesses. Branded products and retail solutions segment is targeted to provide an end-to-end customer service and has expanded its base to provide unique services to its existing & new customers.

Developing new solutions in the emerging sectors has provided a big fillip to expanding the scope of businesses. Branded products and retail solutions segment is targeted to provide an end-to-end customer service and has expanded its base to provide unique services to its existing & new customers.

Introduction

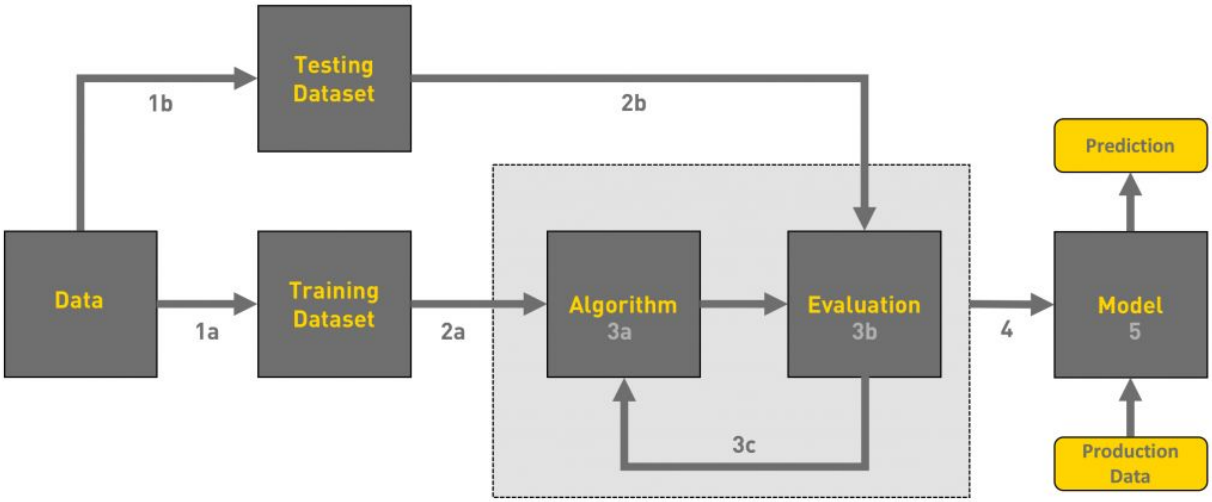
Machine Learning

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: ***The ability to learn***. Machine learning is actively being used today, perhaps in many more places than one would expect.

WORKFLOW OF A MACHINE LEARNING PROJECT

We will also go over data pre-processing, data cleaning, feature exploration and feature engineering and show the impact that it has on Machine Learning Model Performance. We will also cover a couple of the pre-modelling steps that can help to improve the model performance.

Python Libraries that would be need to achieve the task: 1. Numpy 2. Pandas 3. Sci-kit Learn 4. Matplotlib



Overview of the Workflow of ML

Understanding the machine learning workflow

We can define the machine learning workflow in 3 stages.

1. Gathering data
2. Data pre-processing
3. Researching the model that will be best for the type of data
4. Training and testing the model
5. Evaluation

Okay but first let's start from the basics

1. Gathering Data

The process of gathering data depends on the type of project we desire to make, if we want to make an ML project that uses real-time data, then we can build an IoT system that using different sensors data. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data. Therefore, to solve this problem Data Preparation is done.

2. Data pre-processing

Data pre-processing is one of the most important steps in machine learning. It is the most important step that helps in building machine learning models more accurately. In machine learning, there is an 80/20 rule. Every data scientist should spend 80% time for data pre-processing and 20% time to actually perform the analysis.

What is data pre-processing?

Data pre-processing is a process of cleaning the raw data i.e. the data is collected in the real world and is converted to a clean data set. In other words, whenever the data is gathered from different sources it is collected in a raw format and this data isn't feasible for the analysis.Therefore, certain steps are executed to convert the data into a small clean data set, this part of the process is called as data pre-processing.

Why do we need it?

As we know that data pre-processing is a process of cleaning the raw data into clean data, so that can be used to train the model. So, we definitely need data pre-processing to achieve good results from the applied model in machine learning and deep learning projects.

Most of the real-world data is messy, some of these types of data are:

1. **Missing data:** Missing data can be found when it is not continuously created or due to technical issues in the application (IOT system).

2. **Noisy data:** This type of data is also called outliners, this can occur due to human errors (human manually gathering the data) or some technical problem of the device at the time of collection of data.

3. **Inconsistent data:** This type of data might be collected due to human errors (mistakes with the name or values) or duplication of data.

Three Types of Data

1. Numeric e.g. income, age

2. Categorical e.g. gender, nationality

3. Ordinal e.g. low/medium/high

How can data pre-processing be performed?

These are some of the basic pre — processing techniques that can be used to convert raw data.

1. **Conversion of data:** As we know that Machine Learning models can only handle numeric features, hence categorical and ordinal data must be somehow converted into numeric features.

2. **Ignoring the missing values:** Whenever we encounter missing data in the data set then we can remove the row or column of data depending on our need. This method is known to be efficient but it shouldn't be performed if there are a lot of missing values in the dataset.

3. **Filling the missing values:** Whenever we encounter missing data in the data set then we can fill the missing data manually, most commonly the mean, median or highest frequency value is used.

4. **Machine learning:** If we have some missing data then we can predict what data shall be present at the empty position by using the existing data.

5. **Outliers detection:** There are some error data that might be present in our data set that deviates drastically from other observations in a data set. [Example: human weight = 800 Kg; due to mistyping of extra 0]

3. Researching the model that will be best for the type of data :

Supervised Learning

In supervised learning, we are given a data set and already know what our correct output should look like, having the idea that there is a relationship between the input and the output.

Supervised learning problems are categorized into "regression" and "classification" problems. In a regression problem, we are trying to predict results within a continuous output, meaning that we are trying to map input variables to some continuous function. In a classification problem, we are instead

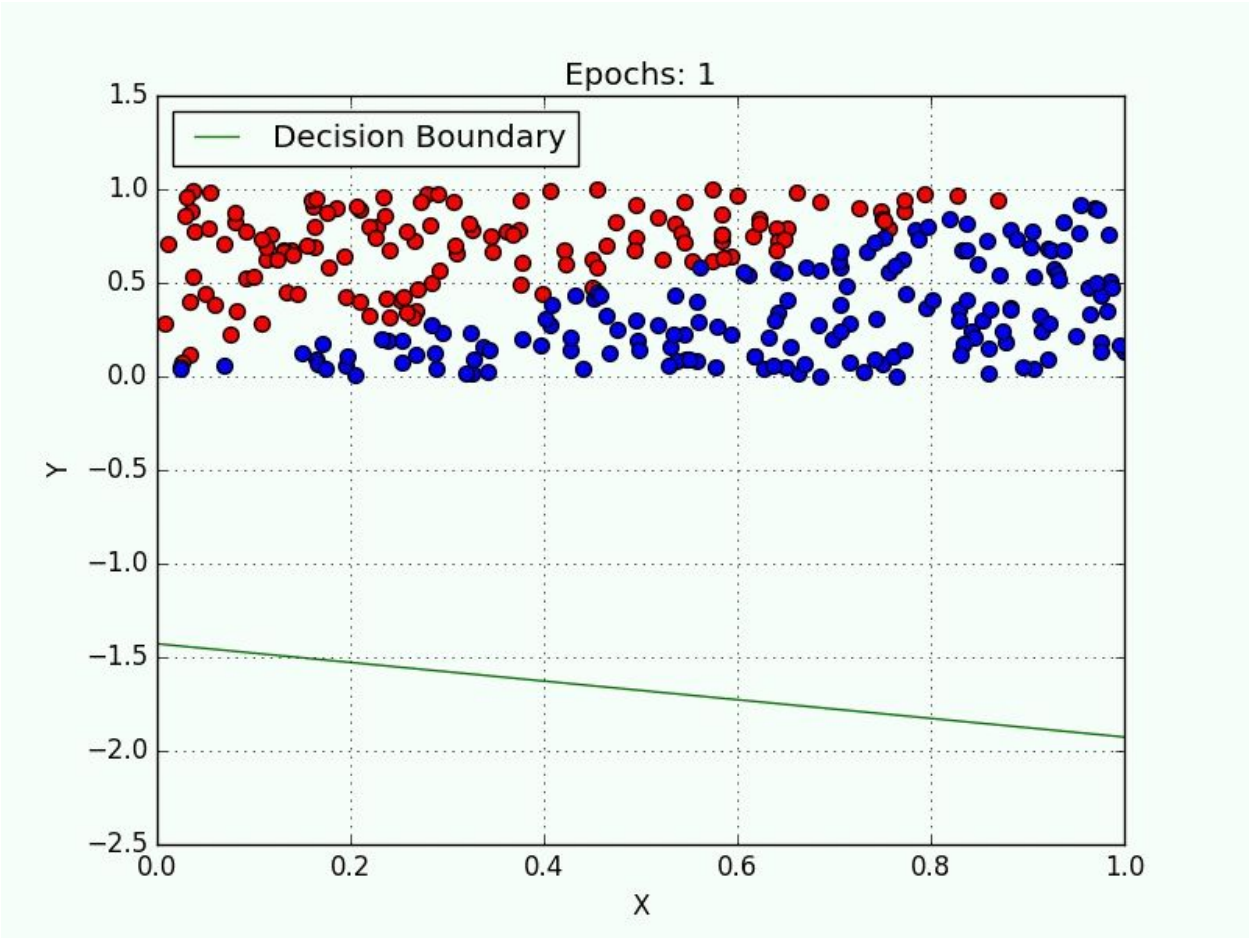
trying to predict results in a discrete output. In other words, we are trying to map input variables into discrete categories.

Example 1: Given data about the size of houses on the real estate market, try to predict their price. Price as a function of size is a continuous output, so this is a regression problem.

We could turn this example into a classification problem by instead making our output about whether the house "sells for more or less than the asking price." Here we are classifying the houses based on price into two discrete categories

Classification:

Classification problem is when the target variable is **categorical** (i.e. the output could be classified into classes — it belongs to either Class A or B or something else).A classification problem is when the output variable is a category, such as “red” or “blue” , “disease” or “no disease” or “spam” or “not spam”.



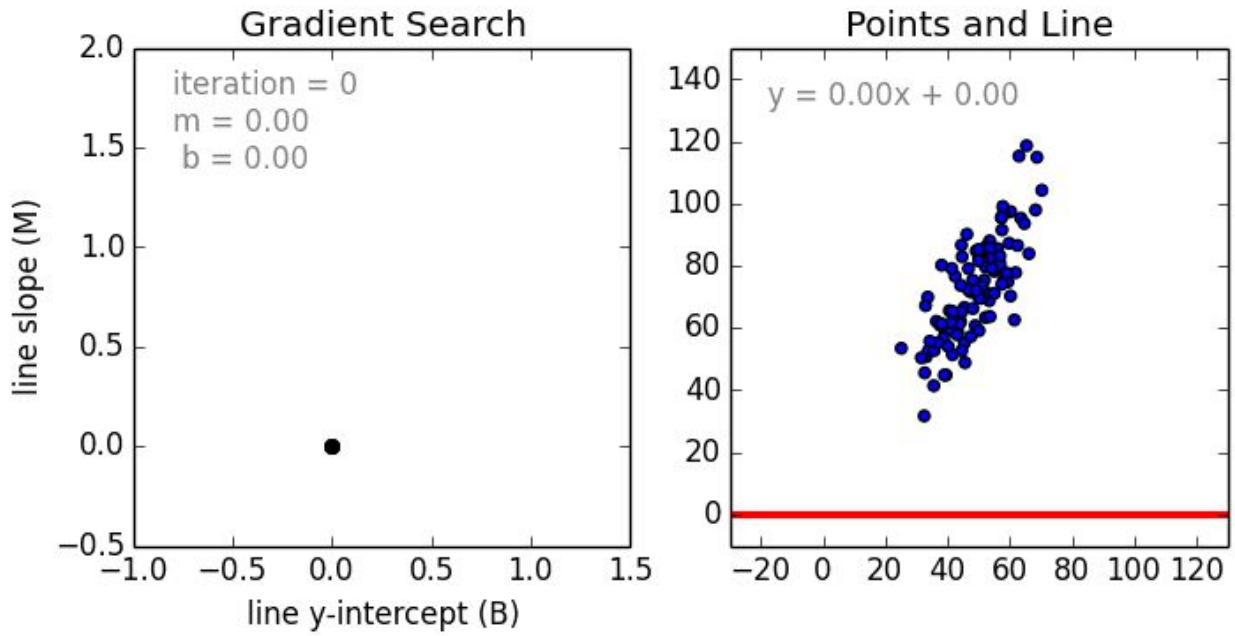
As shown in the above representation, we have 2 classes which are plotted on the graph i.e. red and blue which can be represented as 'setosa flower' and 'versicolor flower', we can image the X-axis as ther 'Sepal Width' and the Y-axis as the 'Sepal Length', so we try to create the best fit line that separates both classes of flowers.

These some most used classification algorithms

1. **.K-Nearest Neighbor**
2. **Naive BayesDecision**

3. **Trees/Random Forest**
4. **Support Vector Machine**
5. **Logistic Regression**

Regression : While a **Regression** problem is when the target variable is **continuous** (i.e. the output is numeric).



As shown in the above representation, we can imagine that the graph's X-axis is the 'Test scores' and the Y-axis represents 'IQ'. So we try to create the best fit line in the given graph so that we can use that line to predict any approximate IQ that isn't present in the given data.

These some most used regression algorithms.

1. Linear Regression

2. Support Vector Regression

3. Decision Tress/Random Forest

4. Gaussian Progresses

2. Unsupervised Learning

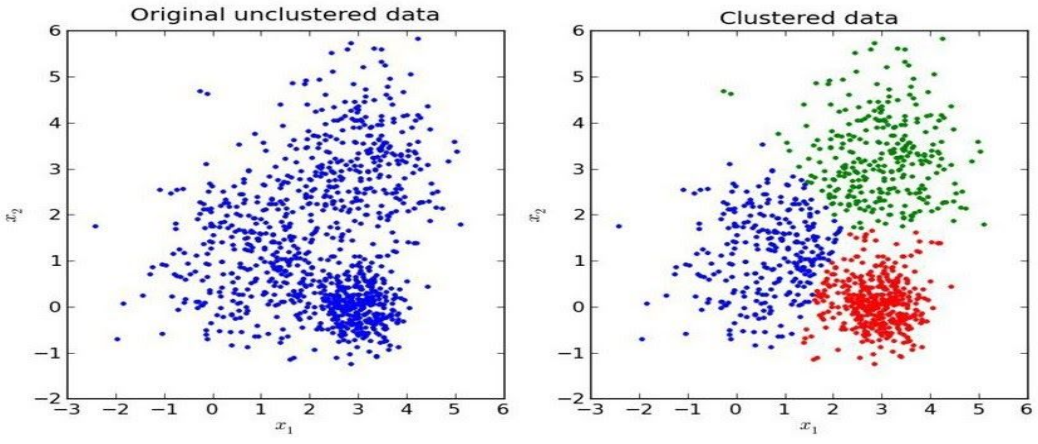
Unsupervised learning allows us to approach problems with little or no idea what our results should look like. We can derive structure from data where we don't necessarily know the effect of the variables.

We can derive this structure by clustering the data based on relationships among the variables in the data.

With unsupervised learning there is no feedback based on the prediction results.

Example: Clustering: Take a collection of 1,000,000 different genes, and find a way to automatically group these genes into groups that are somehow similar or related by different variables, such as lifespan, location, roles, and so on.

Non-clustering: The "Cocktail Party Algorithm", allows you to find structure in a chaotic environment. (i.e. identifying individual voices and music from a mesh of sounds at



a cocktail party).

The unsupervised learning is categorized into 2 other categories which are "**Clustering**" and "**Association**".

Clustering

A set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.

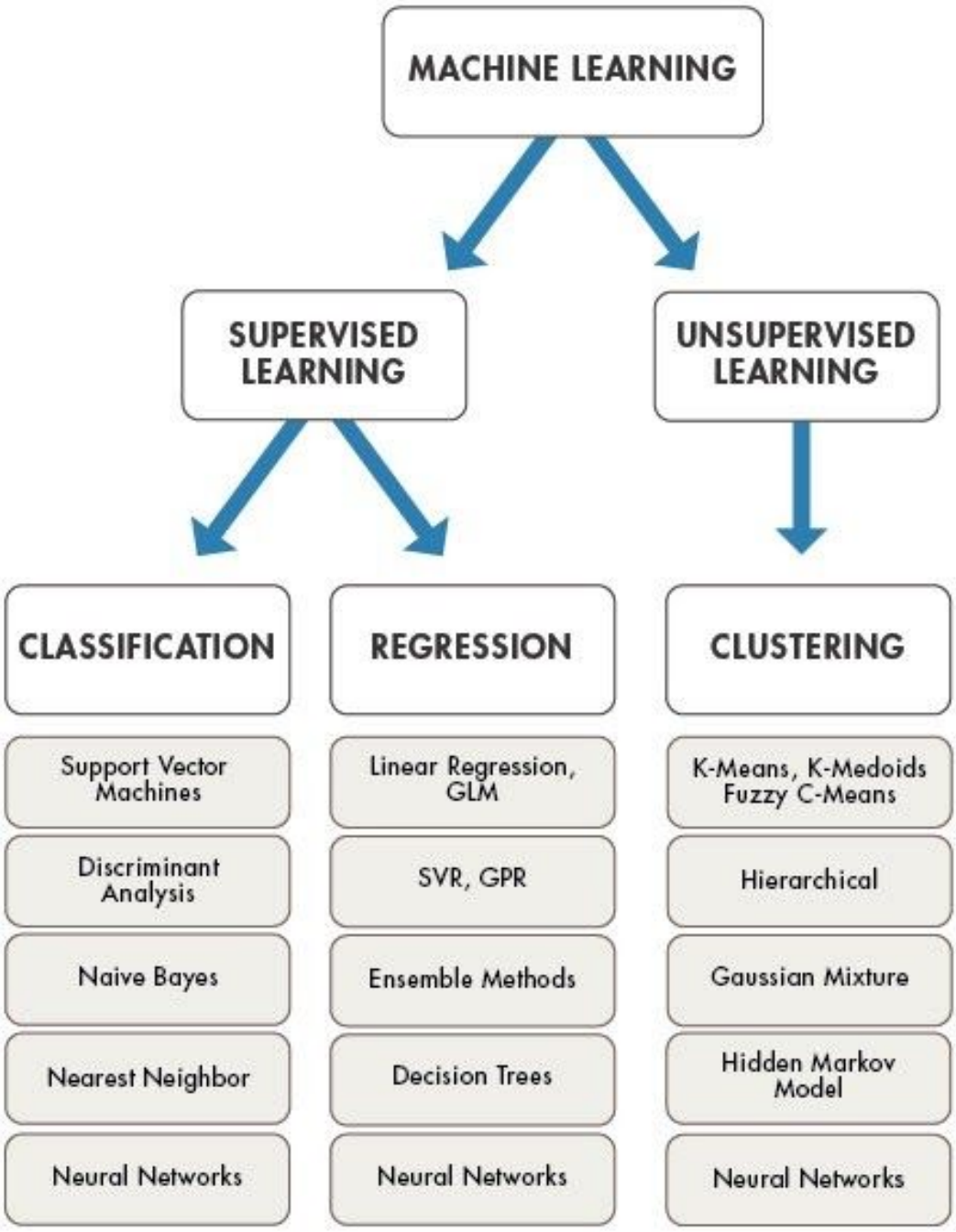
Methods used for clustering are:

1.Gaussian mixtures

2. K-Means Clustering **3. Boosting**

4. Hierarchical Clustering **5 . Spectral Clustering**

Overview of models under categories


MACHINE LEARNING
SUPERVISED LEARNING
UNSUPERVISED LEARNING
CLASSIFICATION
REGRESSION
CLUSTERING
Support Vector Machines
Discriminant Analysis
Naive Bayes
Nearest Neighbor
Neural Networks
Linear Regression, GLM
SVR, GPR
Ensemble Methods
Decision Trees
Neural Networks
K-Means, K-Medoids Fuzzy C-Means
Hierarchical
Gaussian Mixture
Hidden Markov Model
Neural Networks

4. Training and testing the model on data

For training a model we initially split the model into 3 three sections which are '**Training data**' ,'**Validation data**' and '**Testing data**'

You train the classifier using '**training data set**', tune the parameters using '**validation set**' and then test the performance of your classifier on unseen '**test data set**'. An important point to note is that during training the classifier only the training and/or validation set is available. The test data set must not be used during training the classifier. The test set will only be available during testing the classifier.

Training set: The training set is the material through which the computer learns how to process information. Machine learning uses algorithms to perform the training part. A set of data used for learning, that is to fit the parameters of the classifier.

Validation set: Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. A set of unseen data is used from the training data to tune the parameters of a classifier.

Test set: A set of unseen data used only to assess the performance of a fully-specified classifier.

Once the data is divided into the 3 given segments we can start the training process.

In a data set, a training set is implemented to build up a model, while a test (or validation) set is to validate the model built. Data points in the training set are excluded from the test (validation) set. Usually, a data set is divided into a training set, a validation set (some people use 'test set' instead) in each iteration, or divided into a training set, a validation set and a test set in each iteration.

The model uses any one of the models that we had chosen in step 3/ point 3. Once the model is trained we can use the same trained model to predict using the testing data i.e. the unseen data. Once this is done we can develop a confusion matrix, this tells us how well our model is trained. A confusion matrix has 4 parameters, which are '**True positives'**, **'True Negatives'**,**'False Positives'** and '**False Negative'**. We prefer that we get more values in the True negatives and true positives to get a more accurate model. The size of the Confusion matrix completely depends upon the number of classes.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

True positives : These are cases in which we predicted TRUE and our predicted output is correct.

True negatives : We predicted FALSE and our predicted output is correct.

False positives : We predicted TRUE, but the actual predicted output is FALSE.

False negatives : We predicted FALSE, but the actual predicted output is TRUE.

We can also find out the accuracy of the model using the confusion matrix.

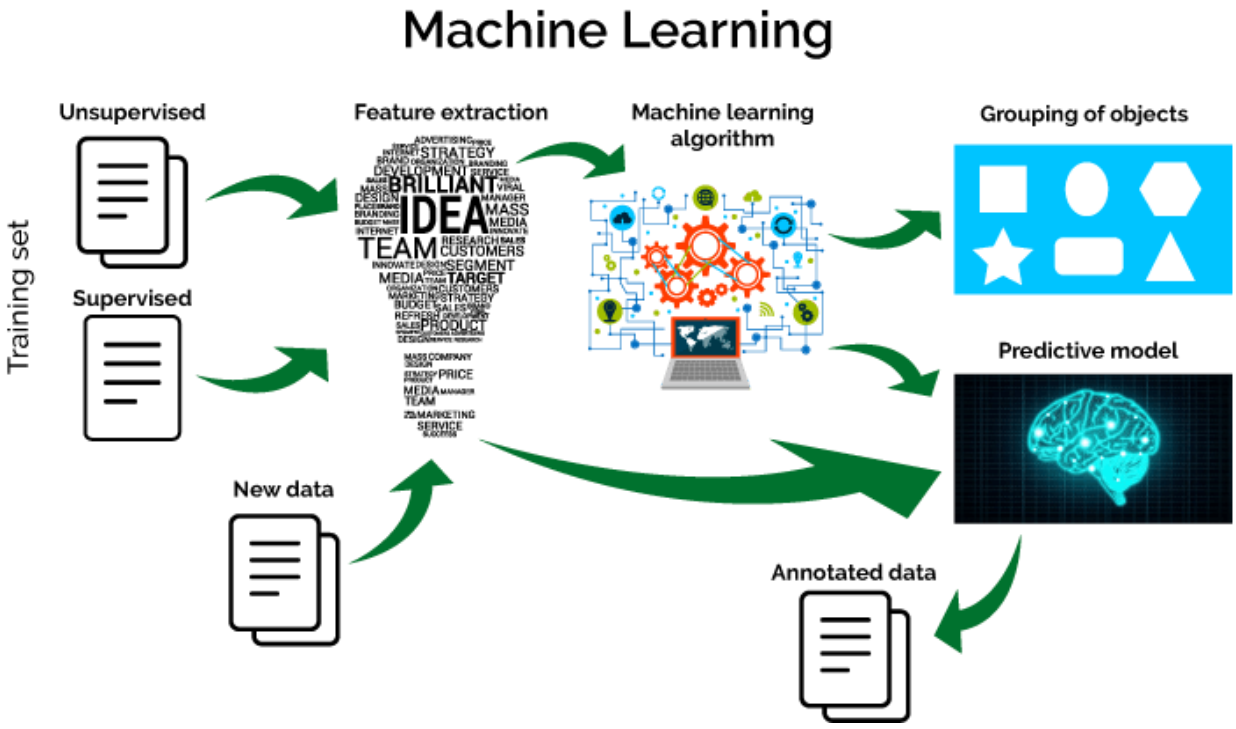
Accuracy = (True Positives +True Negatives) / (Total number of classes)

i.e. for the above example:

Accuracy = (100 + 50) / 165 = 0.9090 (90.9% accuracy)

5. Evaluation

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future.



To improve the model we might tune the hyper-parameters of the model and try to improve the accuracy and also looking at the confusion matrix to try to increase the number of true positives and true negatives.

Data Mining Algorithms

1. C4.5 Algorithm

C4.5 is one of the top data mining algorithms and was developed by Ross Quinlan. C4.5 is used to generate a classifier in the form of a decision tree from a set of data that has already been classified. Classifier here refers to a data mining tool that takes data that we need to classify and tries to predict the class of new data.

Every data point will have its own attributes. The decision tree created by C4.5 poses a question about the value of an attribute and depending on those values, the new data gets classified. The training dataset is labelled with lasses making C4.5 a supervised learning algorithm. Decision trees are always easy to interpret and explain making C4.5 fast and popular compared to other data mining algorithms.

2. K-mean Algorithm

One of the most common clustering algorithms, k-means works by creating a k number of groups from a set of objects based on the similarity between objects. It may not be guaranteed that group members will be exactly similar, but group members will be more similar as compared to non-group members. As per standard implementations, k-means is an unsupervised learning algorithm as it learns the cluster on its own without any external information.

3. Support Vector Machines

In terms of tasks, Support vector machine (SVM) works similar to C4.5 algorithm except that SVM doesn't use any decision trees at all. SVM learns the datasets and defineshyperplane to classify data into two classes. A hyperplane is an equation for a line that looks something

like "*y = mx + b".* SVM exaggerates to project your data to higher dimensions. Once projected, SVM defined the best hyperplane to separate the data into the two classes.

4. Apriori Algorithm

Apriori algorithm works by learning association rules. Association rules are a data mining technique that is used for learning correlations between variables in a database. Once the association rules are learned, it is applied to a database containing a large number of transactions. Apriori algorithm is used for discovering interesting patterns and mutual relationships and hence is treated as an unsupervised learning approach. Thought the algorithm is highly efficient, it consumes a lot of memory, utilizes a lot of disk space and takes a lot of time.

5. Expectation-Maximization Algorithm

Expectation-Maximization (EM) is used as a clustering algorithm, just like the k-means algorithm for knowledge discovery. EM algorithm work in iterations to optimize the chances of seeing observed data. Next, it estimates the parameters of the statistical model with unobserved variables, thereby generating some observed data.
Expectation-Maximization (EM) algorithm is again unsupervised learning since we are using it without providing any labelled class information

6. PageRank Algorithm

PageRank is commonly used by search engines like Google. It is a link analysis algorithm that determines the relative importance of an object linked within a network of objects. Link analysis is a type of network analysis that explores the associations among objects. Google search uses this algorithm by understanding the backlinks between web pages.

It is one of the methods Google uses to determine the relative importance of a webpage and rank it higher on google search engine. The PageRank trademark is proprietary of Google and the PageRank algorithm is patented by Stanford University. PageRank is treated as an unsupervised learning approach as it determines the relative importance just by considering the links and doesn't require any other inputs.

7. Adaboost Algorithm

AdaBoost is a boosting algorithm used to construct a classifier. A classifier is a data mining tool that takes data predicts the class of the data based on inputs. Boosting algorithm is an ensemble learning algorithm which runs multiple learning algorithms and combines them.

Boosting algorithms take a group of weak learners and combine them to make a single strong learner. A weak learner classifies data with less accuracy. The best example of a weak algorithm is the decision stump algorithm which is basically a one-step decision tree. Adaboost is perfect supervised learning as it works in iterations and in each iteration, it trains the weaker learners with the labelled dataset. Adaboost is a simple and pretty straightforward algorithm to implement. After the user specifies the number of rounds, each successive AdaBoost iteration redefines the weights for each of the best learners. This makes Adaboost a super elegant way to auto-tune a classifier. Adaboost is flexible, versatile and elegant as it can incorporate most learning algorithms and can take on a large variety of data.

8. kNN Algorithm

kNN is a lazy learning algorithm used as a classification algorithm. A lazy learner will not do anything much during the training process except for storing the training data. Lazy learners start classifying only when new unlabeled data is given as an input. C4.5, SVN and Adaboost, on the other hand, are eager learners that start to build the classification model during training itself. Since kNN is given a labelled training dataset, it is treated as a supervised learning algorithm.

9. Naive Bayes Algorithm

Naive Bayes is not a single algorithm though it can be seen working efficiently as a single algorithm. Naive Bayes is a bunch of classification algorithms put together. The assumption used by the family of algorithms is that every feature of the data being classified is independent of all other features that are given in the class. Naive Bayes is provided with a labelled training dataset to construct the tables. So it is treated as a supervised learning algorithm.

10. CART Algorithm

CART stands for classification and regression trees. It is a decision tree learning algorithm that gives either regression or classification trees as an output. In CART, the decision tree nodes will have precisely 2 branches. Just like C4.5, CART is also a classifier.

DATA ANALYSIS TYPES

There are four types of data analysis that are in use across all industries. While we separate these into categories, they are all linked together and build upon each other. As you begin moving from the simplest type of analytics, the degree of difficulty and resources required increases. At the same time, the level of added insight and value also increases.

There are 4 types of Data Analysis types/techniques : -

1. DESCRIPTIVE ANALYSIS

2 . DIAGNOSTIC ANALYSIS

3 . PREDICTIVE ANALYSIS

4 . PRESCRIPTIVE ANALYSIS

1. **Descriptive Analysis**

This analysis is the first type and it lays the foundation for the rest of the types. Descriptive analysis as the name suggests is just descriptive. They do not go generalizing beyond the data in hand. It is the simplest of all and with the most practical uses in the business today.

Descriptive analysis answers the “what happened” by summarizing past data usually in the form of dashboards. It helps to describe and present data in a format which can be easily understood by a wide variety of business readers. It rarely attempts to investigate or establish cause and effect relationships.

As this form of analytics doesn’t usually probe beyond surface analysis, the validity of results is more easily implemented. Some common methods employed in Descriptive Analytics are observations, case studies, and surveys. Thus, collection and interpretation of a large amount of data may be involved in this type of analytics.

An example of descriptive analysis includes : Video streaming platform such as Netflix, Amazon Prime have monitoring tools and sentiment analysis tools help you identify who your potential customers are, what matters most to them and how they tend to behave in different social platforms. Netflix today is using metadata not only to make your viewing better with its recommendations engine but also to improve the quality of the content itself by funding new productions based on what they see viewers asking for. And improving their own profitability.

2. **Diagnostic Analysis**

After laying the foundation for our analysis by asking “what happened” we generally want to know the reason as to “why did it happen?” This is answered by diagnostic analysis.

Diagnostic analysis takes the insight found from descriptive analytics and drills down to find the cause of that outcome. It can be characterized by techniques such as drill-down, data discovery, data mining and correlation. Organizations make use of this type of analytics to identify patterns of behaviour in data.

A critical aspect of the same is creating detailed information. Let’s take an example, your CEO asks you a question “Why did you have 20% fewer sales in October, despite investing more in LinkedIn marketing?”. Upon investigation, you notice that in the Event Actions/Goals some user from LinkedIn added the product to cart, but did not check out. The data shows that Exit Rates are through the roof when customers were asked to provide their shipping address and payment details. A lot of things could have gone wrong:

The form didn’t load correctly. The shipping fee was too high

The form was too long and non-mobile friendly. Not enough payment options available

The goal of diagnostic analytics is to help you locate the root cause of the problem. To do so, the algorithms use owned proprietary data, and leverage outside information to understand what exactly happened and help you find a quick fix.

3 . Predictive Analysis

Now the question arises “What is likely to happen?” Here predictive analysis helps us to make sense of why certain things happened and then build a model to project what could have happened in the near future.

Predictive analysis is one step up from the previous types. Where diagnostic analysis helps us to find the pattern in a well-organized data using descriptive analysis, here predictive makes use of both the types and try to make logical predictions of the outcome of events.

Predictive analysis is remarkably beneficial for businesses as it can serve as a guide in making their operations more efficient by cutting down on the costs. By optimizing marketing campaigns with predictive analytics, organizations can also generate new customer responses or purchases, as well as promote cross-sell opportunities. Predictive models can help businesses attract, retain and nurture their most valued customers.

Predictive analytics can also be used to detect and halt various types of criminal behaviour before any serious damage is inflected. By using predictive analytics to study user behaviours and actions, an organization can detect activities that are out of the ordinary, ranging from credit card fraud to corporate spying to cyberattacks.

4 . Prescriptive Analysis

Prescriptive analysis is the area of business analytics dedicated to finding the best course of action for any given situation. Apart from answering “What is likely to happen?” it also

tackles “what should we do to reach the desired outcome?” component. Prescriptive analysis is the frontier of data analysis, it takes into the insights of all the previous analyses to determine the best course of action for a problem or decision.

The best example of prescriptive analytics in action is Google’s self-driving car. The algorithms powering them takes into account several predictions for every trip. Predictions made keeping the rules of the road in mind, such as to drive in between the white lanes, maintain the speed of the car under the speed limit, speed of the vehicle required to maintain to reach from one destination to other, a safe distance between the travelling vehicles so as to avoid a crash and many more are at play.

As technology continues to improve and more professionals are educated in data, we will see more companies entering the data-driven realm.

DATA MINING PHASES/ TYPES

INTRODUCTION

Iron ores are rocks and minerals from which metallic iron can be economically extracted.

The ores are usually rich in iron oxides and vary in color from dark grey, bright yellow, or deep purple to rusty red.

The iron is usually found in the form of magnetite (Fe3O4, 72.4% Fe), hematite (Fe2O3, 69.9% Fe), goethite (FeO(OH), 62.9% Fe), limonite (FeO(OH)·n(H_2O), 55% Fe) or siderite ($FeCO_3$, 48.2% Fe).

Ores containing very high quantities of hematite or magnetite (greater than about 60% iron) are known as "natural ore" or "direct shipping ore", meaning they can be fed directly into iron-making blast furnaces. Iron ore is the raw material used to make pig iron, which is one of

the main raw materials to make steel—98% of the mined iron ore is used to make steel. In 2011 the Financial Times has speculated that iron ore is "more integral to the global economy than any other commodity, except perhaps oil".

Mining iron ore is a high-volume, low-margin business, as the value of iron is significantly lower than base metals. It is highly capital intensive, and requires significant investment in infrastructure such as rail in order to transport the ore from the mine to a freight ship. For these reasons, iron ore production is concentrated in the hands of a few major players.

SMELTING

Iron ores consist of oxygen and iron atoms bonded together into molecules. To convert it to metallic iron it must be smelted or sent through a direct reduction process to remove the oxygen. Oxygen-iron bonds are strong, and to remove the iron from the oxygen, a stronger elemental bond must be presented to attach to the oxygen. Carbon is used because the strength of a carbon-oxygen bond is greater than that of the iron-oxygen bond, at high temperatures. Thus, the iron ore must be powdered and mixed with coke, to be burnt in the smelting process.

Carbon monoxide is the primary ingredient of chemically stripping oxygen from iron. Thus, the iron and carbon smelting must be kept at an oxygen-deficient (reducing) state to promote burning of carbon to produce CO not CO
2.

- Air blast and charcoal (coke): $2\ C + O_2 \rightarrow 2\ CO$
- Carbon monoxide (CO) is the principal reduction agent.
    - Stage One: $3\ Fe_2O_3 + CO \rightarrow 2\ Fe_3O_4 + CO_2$
    - Stage Two: $Fe_3O_4 + CO \rightarrow 3\ FeO + CO_2$
    - Stage Three: $FeO + CO \rightarrow Fe + CO_2$

- Limestone calcining: $CaCO_3 \rightarrow CaO + CO_2$
- Lime acting as flux: $CaO + SiO_2 \rightarrow CaSiO_3$

SILICON

Silica (SiO_2) is almost always present in iron ore. Most of it is slagged off during the smelting process. At temperatures above 1,300 °C (2,370 °F) some will be reduced and form an alloy with the iron. The hotter the furnace, the more silicon will be present in the iron. It is not uncommon to find up to 1.5% Si in European cast iron from the 16th to 18th centuries.

The major effect of silicon is to promote the formation of grey iron. Grey iron is less brittle and easier to finish than white iron. It is preferred for casting purposes for this reason. Turner (1900, pp. 192–197) reported that silicon also reduces shrinkage and the formation of blowholes, lowering the number of bad castings.

This page looks at the use of the Blast Furnace in the extraction of iron from iron ore, and the conversion of the raw iron from the furnace into various kinds of steel.

EXTRACTING IRON FROM IRON ORE USING A BLAST FURNACE

The common ores of iron are both iron oxides, and these can be reduced to iron by heating them with carbon in the form of coke. Coke is produced by heating coal in the absence of air. Coke is cheap and provides both the reducing agent for the reaction and also the heat source. The most commonly used iron ores are hematite,

Fe_2O_3 Fe2O3, and magnetite,

Fe_3O_4

The Blast Furnace

The significant reactions occuring within the Blast Furnace can be described via the following steps showing how the reducing agent varies depending on the height in the furnace (i.e. on the Temperature).

At 500 °C

$3Fe_2O_3 + CO \rightarrow 2Fe_3O_4 + CO_2$

$Fe_2O_3 + CO \rightarrow 2FeO + CO_2$

At 850 °C

$Fe_3O_4 + CO \rightarrow 3FeO + CO_2$

At 1000 °C

$FeO + CO \rightarrow Fe + CO_2$

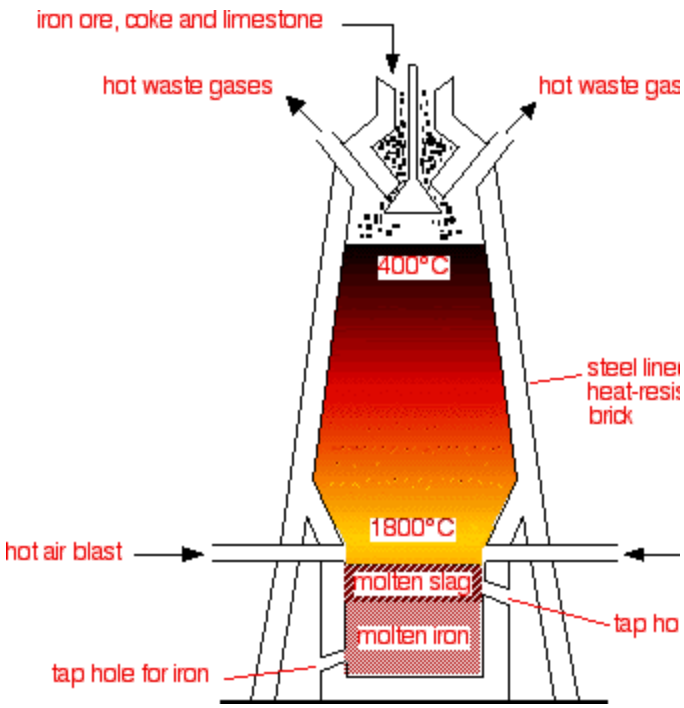
At 1300 °C

$CO_2 + C \rightarrow 2CO$

At 1900 °C

$C + O_2 \rightarrow CO_2$

$FeO + C \rightarrow Fe + CO$



The air blown into the bottom of the furnace is heated using the hot waste gases from the top. Heat energy is valuable, and it is important not to waste any. The coke (essentially impure carbon) burns in the blast of hot air to form carbon dioxide - a strongly exothermic reaction. This reaction is the main source of heat in the furnace.

$$C + O_2 \rightarrow CO_2 \quad (1)$$

THE REDUCTION OF THE ORE

At the high temperature at the bottom of the furnace, carbon dioxide reacts with carbon to produce carbon monoxide.

$$C+CO2 \rightarrow 2CO \quad (2)$$

It is the carbon monoxide which is the main reducing agent in the furnace.

$$Fe2O3+3CO \rightarrow 2Fe+3CO2$$

In the hotter parts of the furnace, the carbon itself also acts as a reducing agent. Notice that at these temperatures, the other product of the reaction is carbon monoxide, not carbon dioxide.

$$Fe2O3+3C \rightarrow 2Fe+3CO \quad (4)$$

The temperature of the furnace is hot enough to melt the iron which trickles down to the bottom where it can be tapped off.

THE FUNCTION OF THE LIMESTONE

Iron ore is not pure iron oxide - it also contains an assortment of rocky material that would not melt at the temperature of the furnace, and would eventually clog it up. The limestone is added to convert this into slag which melts and runs to the bottom. The heat of the furnace decomposes the limestone to give calcium oxide.

$$CaCO3 \rightarrow CaO+CO2 \quad (5)$$

This is an endothermic reaction, absorbing heat from the furnace. It is therefore important not to add too much limestone because it would otherwise cool the furnace. Calcium oxide is a basic

oxide and reacts with acidic oxides such as silicon dioxide present in the rock. Calcium oxide reacts with silicon dioxide to give calcium silicate.

$$CaO+SiO2 \rightarrow CaSiO3 \quad (6)$$

The calcium silicate melts and runs down through the furnace to form a layer on top of the molten iron. It can be tapped off from time to time as slag. Slag is used in road making and as "slag cement" - a final ground slag which can be used in cement, often mixed with Portland cement.

DEFINE THE PROBLEM

So the problem which we will dealing is to reduce hot metal silicon at GBF . So to go about this we use the data mining technique.

a) Business Case

- Silicon directly impacts the quality as well as the production rate of the hot metal. For e.g. It has a direct impact on the slag viscosity which is a transport property that relates to the reaction kinetics and the degree of reduction of the final slag. It also determines the slag– metal separation efficiency, and subsequently the metal yield and impurity removal capacity. In operation, the slag viscosity is indicative of the ease with which slag could be tapped from the furnace, and therefore relates to the energy requirement and profitability of the process.

Slag also plays a major role in determining the quality of hot metal obtained which, in turn is dependent on the formation of the slag and its mineralogical transformations. A good quality slag is necessary for a quality hot metal. Slag is a mixture of low melting chemical compounds formed by the chemical reaction of the gangue of the iron bearing burden and coke ash with the flux materials in the charge. All unreduced compounds such as silicates, aluminosilicates, and calcium alumino silicate etc. also join the slag. It is well known that the components of slag namely silica (SiO2) and alumina (Al2O3) increase the viscosity whereas the presence of calcium oxide reduces the viscosity. The melting zone of slag determines the cohesive zone of blast furnace and hence the fluidity and melting characteristics of slag play a major role in determining the blast furnace productivity.

Currently, Silicon levels are controlled through experience and some primary data analysis. An example can be that of Hot Metal temperature which shows a direct correlation with Si.

b) Problems in the existing process

- Following are the issues faced :-
i) Does not leverage the historical data to full extent.
ii) Action is based on experience rather than the actual data.
iii) Difficult to arrive at new parameters beyond what is already known.

Why Si?

- Directly impacts the quality and the production rate of hot metal
- e.g. affects slag viscosity
  - transport property related to the reaction kinetics and the degree of reduction of the final slag -> impurity removal capacity
  - the slag–metal separation efficiency -> metal yield

a) Overall objective

- Hence, it is an important factor that needs to be controlled. A data mining model needs to be developed which can help arrive at a set of predominant factors and their relationships for the purpose of reducing the mean value of Silicon. Output of this model will then be verified with the existing data and operating ranges of the final select process parameters need to be determined.

b) Scope

- Functional : Base data needs to be collated and uploaded in central database which will be the input for the model. One part of the data needs to be used for developing the model and the other for testing the model. System should have a generic design to include new factors and if possible deployable for similar data mining cases of other blast furnaces.

Technical : Following program structures need to be implemented :-
i) Model to draw correlation and narrow down the factors impacting Silicon
ii) Model to decide on the operating ranges for the actionable parameters

Locational : Following Business Units will be included :-
i) GBF

c) Benefits and KPI

- KPI : Silicon in Hot Metal
UOM : %
Base Level : 0.75
Target Level : 0.70
Tangible Benefit : Rs 2 Cr

IDENTIFY AND PREPARE THE DATA

a) Data Source and collection

- Relevant data was collected from iMonitor system. Data is primarily of 4 types which are process parameters, hot metal chemistry, input coke data and input sinter data. Frequency is date wise. Period under consideration is between 1 Jan 2014 to 15 Oct 2017. As data was extracted from different screens/reports, parameters were arranged as per the date. Also, effort was put in for data preparation as data is not ready for analysis. A total of 112 parameters were captured initially and all were numerical data except the date field.

b) Assumptions

- Following are the assumptions made :-
i) Si value>=0.70 is to be considered as High Silicon.
ii) Days with Availability > 95% are assumed to be days with stable operation.
iii) A ratio 1:2 for Low-High cases is good enough for prediction of Low cases.
iv) Data is relevant and accurate.
v) Conditions for high Silicon may not be entirely conversely true for conditions for low Silicon

Shortlist parameters which have to be considered

a. Domain knowledge
    i. Direct controllers
    ii. Indirect controllers (Indicators)

b. Extract relevant data

a. Availability>=95%
b. 17 months

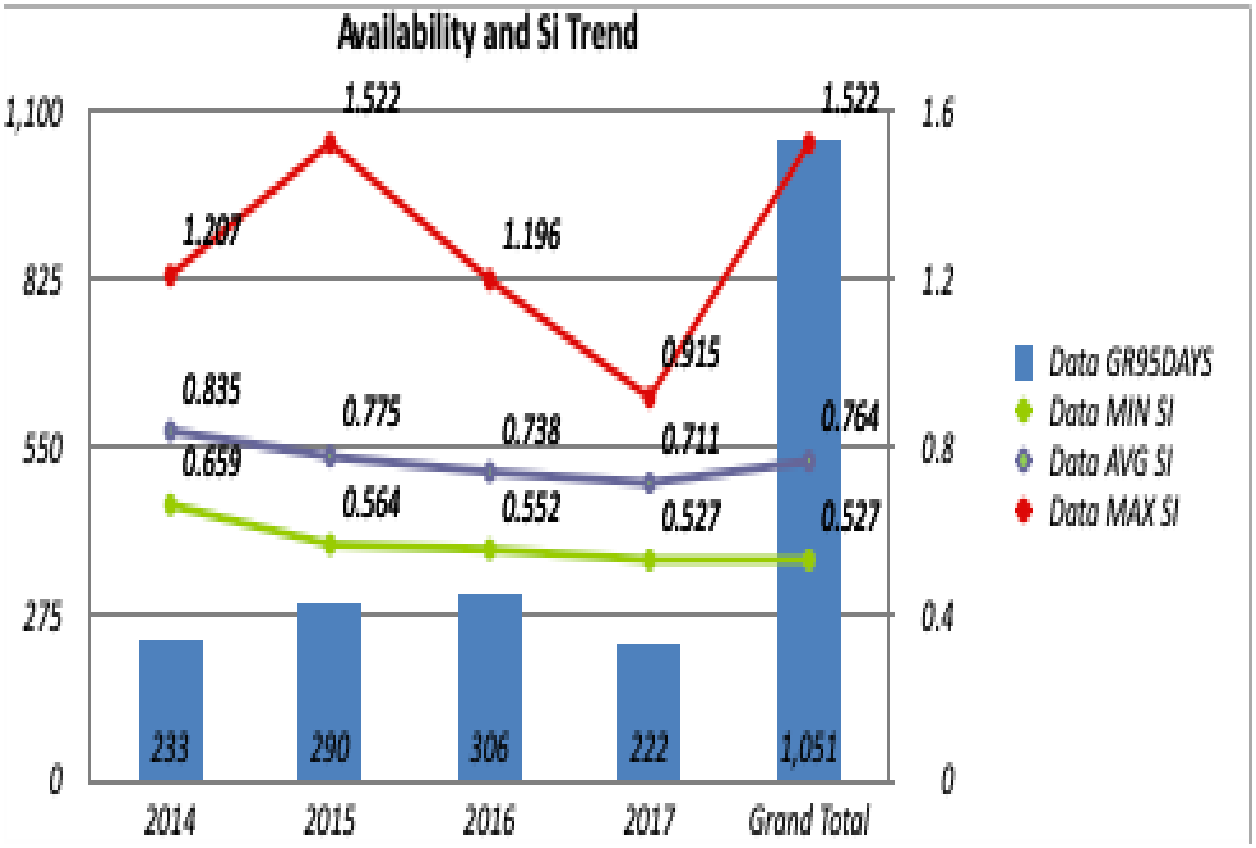
c. Data Sanitization

a. Remove outliers – manual
b. Address missing values/nulls – 0/Average
c. Address class imbalance problem
d. Remove inter-correlated data
e. Categorization
f. Low <=0.70(L)
g. High>0.70(H)

AVAILABILITY AND SI TREND

	Data			
YEAR	**GR95DAYS**	**MIN SI**	**AVG SI**	**MAX SI**
2014	**233**	**0.659**	**0.835**	**1.207**
2015	**290**	**0.564**	**0.775**	**1.522**
2016	**306**	**0.552**	**0.738**	**1.196**
2017	**222**	**0.527**	**0.711**	**0.915**
Grand Total	**1051**	**0.527**	**0.764**	**1.522**

The given data is the availability of silicon in the iron ore found after it undergoes extraction process.



GR95DAYS - No of days with Availability>=95

Data from 1 Jan 2014 to 31 Mar 2017

PREPARE & PROCESS THE DATA

SELECTING RELEVANT DATA – CLASS IMBALANCE PROBLEM

Count of HML_SI				
CLASS	YEAR	HML_SI_CAT	Total	Cum.% in Reverse
TESTING	2017	H	60	55
		L	49	45
	2017 Total		109	
TESTING Total			109	
TRAINING	2014	H	228	79
		L	5	21
	2014 Total		233	
	2015	H	240	73
		L	50	27
	2015 Total		290	
	2016	H	213	67
		L	93	33
	2016 Total		306	
	2017	H	66	58

		L	**47**	**42**
	2017 Total		**113**	
TRAINING Total			**942**	
Grand Total			**1051**	

Due to high bias, data from 2016 onwards was considered
(Trial also done with Training data from 2015 onwards but resulted in drop in accuracy of models)
We can also see the cum % in Reverse and for 2016 onwards the data is more appropriate and practical.

CORRELATION COEFFICIENT

The correlation coefficient (sometimes referred to as Pearson's correlation coefficient, Pearson's product-moment correlation, or simply *r*) measures the strength of the linear relationship between two variables. It is indisputably one of the most commonly used metrics in both science and industry. In science, it is typically used to test for a linear association between two dependent variables, or measurements. In industry, specifically in a machine-learning context, it is used to discover collinearity between features, which may undermine the quality of a model.

STEPS INVOLVED IN FINDING THE CORRELATION MATRIX

STEP 1 : `#importing libraries`

```
import pandas as pd

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

import seaborn as sns

import statsmodels.api as sm

%matplotlib inline

STEP 2 : #Read the dataset using pandas

df = pd.read_csv("base_data2.csv")

df.drop(['CLASS','DT','YEAR','HML_SI_CAT'],axis=1,inplace=True)

Step 3: #Define X and y

X=
df[['CORR_PRODUCTION','CORR_NC_RATE','CORR_COKE_RATE','CORR_COAL_RATE','HM
_TEMP','QUARTZ_RATE','LIMESTONE_RATE','SLAG_RATE','PYROXINITE_RATE','SINTE
R_PERC','PELLET_PERC','HOT_BLAST_TEMP','COLD_BLAST_O2_ENRICH','COLD_BLAST_
HUMID','FUR_RAFT_TEMP','COMP_COLD_BLAST_FLOW','UPTAKE_AVG','HL_TOTAL_GJ_HR
','UPPER_K','MIDDLE_K','LOWER_K','K','HML_S','HML_SIO2','HML_MGO','HML_AL2
O3','HML_CAO_SIO2','COK_MOISTURE','COK_CSR','COK_CRI','COK_AMS','SIN_CAO_S
IO2','SIN_MGO']]

y = df.HML_SI
```

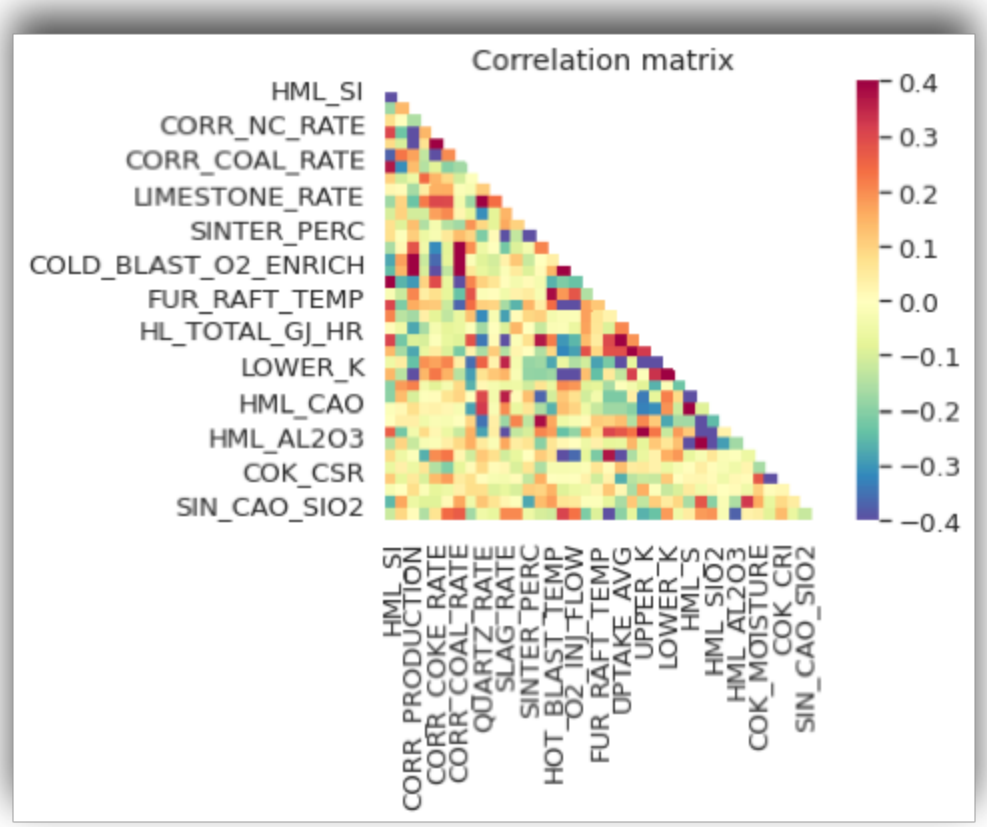
```
STEP 4 : #Define the correlation between each parameter.

df.corr().round(2)

#The correlation matrix for the following dataset

STEP 5 : #Plot the  correlation matrix.

corrs = df.corr()

mask = np.zeros_like(corrs)

mask[np.triu_indices_from(mask)] = True

sns.heatmap(corrs, cmap='Spectral_r', mask=mask, square=True, vmin=-.4,
vmax=.4)

plt.title('Correlation matrix')

STEP 6  : #Finding  the important features

#Correlation with output variable

cor_target =
abs(cor[['HML_SI','CORR_PRODUCTION','CORR_NC_RATE','CORR_COKE_RATE','CORR_
COAL_RATE','HM_TEMP','QUARTZ_RATE','LIMESTONE_RATE','SLAG_RATE','PYROXINIT
E_RATE','SINTER_PERC','PELLET_PERC','HOT_BLAST_TEMP','COLD_BLAST_O2_ENRICH
','COLD_BLAST_HUMID','FUR_RAFT_TEMP','COMP_COLD_BLAST_FLOW','UPTAKE_AVG','
HL_TOTAL_GJ_HR','UPPER_K','MIDDLE_K','LOWER_K','K','HML_S','HML_SIO2','HML
_MGO','HML_AL2O3','HML_CAO_SIO2','COK_MOISTURE','COK_CSR','COK_CRI','COK_A
MS','SIN_CAO_SIO2','SIN_MGO']])
```

```
#Selecting highly correlated features

relevant_features = cor_target[cor_target>0.3]

relevant_features
```

FINAL FEATURES GIVEN BY PEARSON

PARAMETER 1	PARAMETER 2	**FACTOR**
COK_CSR	COK_CRI	0.770252
HML_CAO_SIO2	HML_CAO	0.845912
O2_INJ_FLOW	COLD_BLAST_O2_ENRICH	0.966968

Final data set to be used for model construction had 33 parameters (excluding Si and Availability) and sample set = 419



Exclusions with rationale

- 1) From a set of 112 parameters, a set of 38 parameters were shortlisted on the basis of domain knowledge of the business users. If all the parameters are considered for the subsequent model development, they will lead to the following problems :-

i) Resultant model may give importance to parameters which actually do not play any role in the actual process to control Si.

ii) Since the model development considers the entire data set to determine the important parameters, the non-impacting parameters will also affect the final set.

iii) The base data is loaded using pandas and we drop the parameters 'CLASS','DT','YEAR','HML_SI_CAT to obtain the our final dataset which we will use the train our modelss.

```
df.drop(['CLASS','DT','YEAR','HML_SI_CAT'],axis=1,inplace=True)
df.head(3)
```

List of parameters considered for final set is provided below

Sl No	Name *	Tag
1	COK_AMS	Indicator
2	COK_CRI	Indicator
3	COK_CSR	Indicator
4	COK_MOISTURE	Indicator
5	COLD_BLAST_HUMID	Indicator
6	COLD_BLAST_O2_ENRICH	Control
7	COMP_COLD_BLAST_FLOW	Control
8	CORR_COAL_RATE	Control
9	CORR_COKE_RATE	Control
10	CORR_FUEL_RATE	Control
11	CORR_NC_RATE	Control
12	CORR_PRODUCTION	Control
13	FUR_RAFT_TEMP	Control
14	HL_TOTAL_GJ_HR	Control
15	HM_TEMP	Indicator
16	HML_AL2O3	Indicator
17	HML_CAO	Indicator
18	HML_CAO_SIO2	Control
19	HML_MGO	Control
20	HML_S	Indicator
21	HML_SI	Y
22	HML_SIO2	Indicator
23	HOT_BLAST_TEMP	Indicator
24	K	Indicator
25	LIMESTONE_RATE	Control
26	LOWER_K	Indicator
27	MIDDLE_K	Indicator
28	O2_INJ_FLOW	Indicator
29	PELLET_PERC	Control
30	PYROXINITE_RATE	Control
31	QUARTZ_RATE	Control
32	SIN_CAO_SIO2	Indicator
33	SIN_MGO	Indicator

Missing data recover in python

Pandas series is a One-dimensional ndarray with axis labels. The labels need not be unique but must be a hashable type. The object supports both integer- and label-based indexing and provides a host of methods for performing operations involving the index.

Pandas `Series.fillna()` function is used to fill NA/NaN values using the specified method.

*Syntax: Series.fillna(value=None, method=None, axis=None, inplace=False, limit=None, downcast=None, **kwargs)*

Parameter : value : Value to use to fill holes

method : Method to use for filling holes in reindexed Series pad / ffill axis : {0 or 'index'}

inplace : If True, fill in place.

limit : If method is specified, this is the maximum number of consecutive NaN values to forward/backward fill

downcast : dict, default is None

Returns : filled : Series

Step 1: Read the data

```
df = pd.read_csv("base_data3.csv")
```

Step 2: Find the median of each column and fill it in column using the fillna().

```
df.UPTAKE_AVG = df.UPTAKE_AVG.fillna(q)

df.CORR_COAL_RATE = df.CORR_COAL_RATE.median()

df.HM_TEMP = df.HM_TEMP.median()

df.COK_CSR = df.COK_CSR.median()

df.COK_CRI = df.COK_CRI.median()

df.SIN_CAO_SIO2 = df.SIN_CAO_SIO2.median()

df.SIN_MGO = df.SIN_MGO.median()

df.head(30)
```

As we can see in the output, the `Series.fillna()` function has successfully filled out the missing values in the given series object.

MODEL THE DATA

Metrics To Evaluate Machine Learning Algorithms

The metrics that you choose to evaluate your machine learning algorithms are very important.

Choice of metrics influences how the performance of machine learning algorithms is measured and compared. They influence how you weight the importance of different characteristics in the results and your ultimate choice of which algorithm to choose.

Classification Metrics

Classification problems are perhaps the most common type of machine learning problem and as such there are a myriad of metrics that can be used to evaluate predictions for these problems.

In this section we will review how to use the following metrics:

1. **Classification Accuracy.**
2. **Log Loss.**
3. **Area Under ROC Curve.**
4. **Confusion Matrix.**
5. **Classification Report.**

1. Classification Accuracy

Classification accuracy is the number of correct predictions made as a ratio of all predictions made.

This is the most common evaluation metric for classification problems, it is also the most misused. It is really only suitable when there are an equal number of observations in each class (which is rarely the case) and that all predictions and prediction errors are equally important, which is often not the case.

2. Log Loss

Logistic loss **(or log loss) is a performance metric for evaluating the predictions of probabilities of membership to a given class.**

The scalar probability between 0 and 1 can be seen as a measure of confidence for a prediction by an algorithm. Predictions that are correct or incorrect are rewarded or punished proportionally to the confidence of the prediction.

Smaller log loss is better with 0 representing a perfect log loss. As mentioned above, the measure is inverted to be ascending when using the *cross_val_score()* function.

3. Area Under ROC Curve

Area Under ROC Curve (or ROC AUC for short) is a performance metric for binary classification problems.

The AUC represents a model's ability to discriminate between positive and negative classes. An area of 1.0 represents a model that made all predictions perfectly. An area of 0.5 represents a model as good as random.

A ROC Curve is a plot of the true positive rate and the false positive rate for a given set of probability predictions at different thresholds used to map the probabilities to class labels. The area under the curve is then the approximate integral under the ROC Curve.

You can see the the AUC is relatively close to 1 and greater than 0.5, suggesting some skill in the predictions.

4. Confusion Matrix

The confusion matrix is a handy presentation of the accuracy of a model with two or more classes.

The table presents predictions on the x-axis and accuracy outcomes on the y-axis. The cells of the table are the number of predictions made by a machine learning algorithm.

For example, a machine learning algorithm can predict 0 or 1 and each prediction may actually have been a 0 or 1. Predictions for 0 that were actually 0 appear in the cell for

prediction=0 and actual=0, whereas predictions for 0 that were actually 1 appear in the cell for prediction = 0 and actual=1. And so on.

Although the array is printed without headings, you can see that the majority of the predictions fall on the diagonal line of the matrix (which are correct predictions).

5. Classification Report

Scikit-learn does provide a convenience report when working on classification problems to give you a quick idea of the accuracy of a model using a number of measures.

The *classification_report()* function displays the precision, recall, f1-score and support for each class.

Regression Metrics

In this section will review 3 of the most common metrics for evaluating predictions on regression machine learning problems:

1. **Mean Absolute Error.**
2. **Mean Squared Error.**
3. **R^2.**

1. Mean Absolute Error

The Mean Absolute Error (or MAE) is the average of the absolute differences between predictions and actual values. It gives an idea of how wrong the predictions were.

The measure gives an idea of the magnitude of the error, but no idea of the direction (e.g. over or under predicting).

A value of 0 indicates no error or perfect predictions. Like logloss, this metric is inverted by the *cross_val_score()* function.

2. Mean Squared Error

The Mean Squared Error (or MSE) is much like the mean absolute error in that it provides a gross idea of the magnitude of error.

Taking the square root of the mean squared error converts the units back to the original units of the output variable and can be meaningful for description and presentation. This is called the Root Mean Squared Error (or RMSE).

3. R^2 Metric

The R^2 (or R Squared) metric provides an indication of the goodness of fit of a set of predictions to the actual values. In statistical literature, this measure is called the coefficient of determination.

This is a value between 0 and 1 for no-fit and perfect fit respectively.

You can see that the predictions have a poor fit to the actual values with a value close to zero and less than 0.5.

Summary

You learned about 3 classification metrics:

- **Accuracy.**
- **Log Loss.**
- **Area Under ROC Curve.**

Also 2 convenience methods for classification prediction results:

- **Confusion Matrix.**
- **Classification Report.**

And 3 regression metrics:

- **Mean Absolute Error.**
- **Mean Squared Error.**
- **R^2.**

TRAIN AND TEST

I used three algorithms to train and test the dataset to obtain the output ：

1. LINEAR REGRESSION

in statistics, linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression.

There are two types of supervised machine learning algorithms: Regression and classification. The former predicts continuous value outputs while the latter predicts discrete outputs

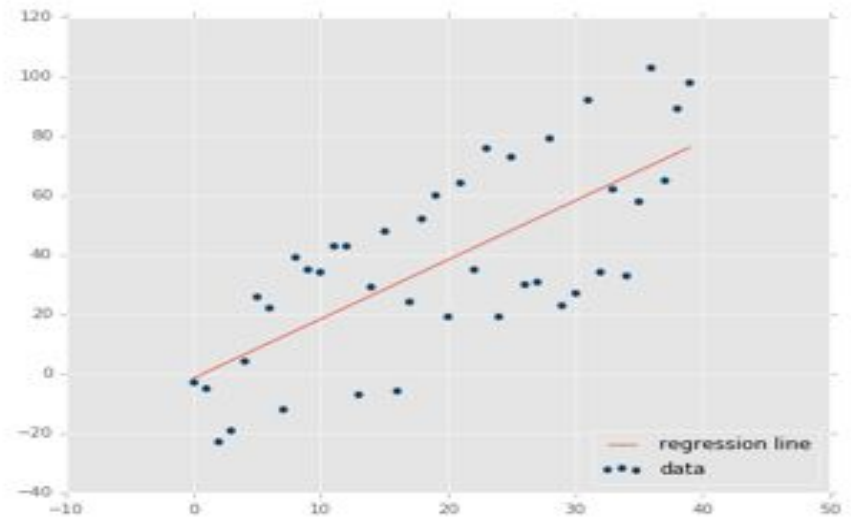
The term “linearity” in algebra refers to a linear relationship between two or more variables. If we draw this relationship in a two-dimensional space (between two variables), we get a straight line.

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

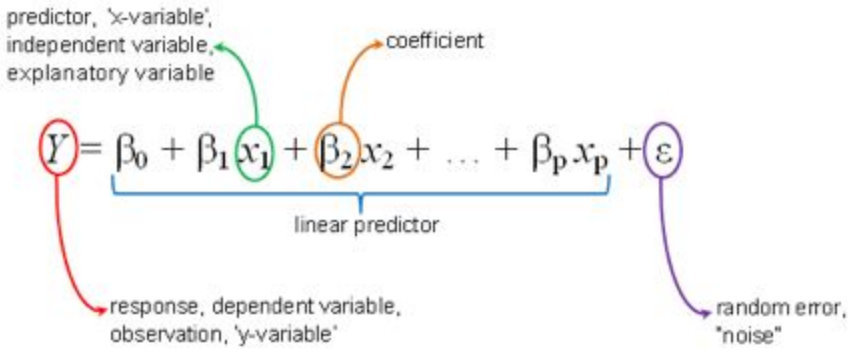
If we plot the independent variable (x) on the x-axis and dependent variable (y) on the y-axis, linear regression gives us a straight line that best fits the data points, as shown in the figure below.
We know that the equation of a straight line is basically:

The equation of the line is :

$$Y = mx + b$$



The dataset here has 34 different parameters , so we can't use simple linear regression, So we choose to apply multiple Multiple Linear Regression.



STEPS INVOLVED IN TRAINING THE MODEL :

Step 1 : import all the required libraries required :

```
import pandas as pd

import numpy as np

from sklearn import linear_model

import matplotlib.pyplot as plt

from sklearn import metrics
```

Step 2 : Read the dataset :

```
df = pd.read_csv("base_data2.csv")
```

Step 3 : Define X and y

```
X=df[['CORR_PRODUCTION','CORR_NC_RATE','CORR_COKE_RATE','CORR_COAL_RATE','HM_TEMP','QUARTZ_RATE','LIMESTONE_RATE','SLAG_RATE','PYROXINITE_RATE','SINTER_PERC','PELLET_PERC','HOT_BLAST_TEMP','COLD_BLAST_O2_ENRICH','COLD_BLAST_HUMID','FUR_RAFT_TEMP','COMP_COLD_BLAST_FLOW','UPTAKE_AVG','HL_T
```

OTAL_GJ_HR','UPPER_K','MIDDLE_K','LOWER_K','K','HML_S','HML_SIO2','HML_MGO','HML_AL2O3','HML_CAO_SIO2','COK_MOISTURE','COK_CSR','COK_CRI','COK_AMS','SIN_CAO_SIO2','SIN_MGO']]

y = df.HML_SI

Step 4 : Use train test split from sklearn

Using train_test_split from sklearn split the dataset

Into X_train , X_test , y_train, y_test.

Into X_train , X_test , y_train, y_test.

Step 5 : lets train our model and fit the dataset.

regressor = LinearRegression()

regressor.fit(X_train, y_train)

Step 6: Now check the accuracy of the model

print(reg.score(X_test, y_test))

Step 7: Now lets you test how well your model fits the data set.

print(metrics.mean_absolute_error(y_test, y_pred))

print(metrics.mean_squared_error(y_test, y_pred))

print(np.sqrt(metrics.mean_squared_error(y_test, y_pred)))

Step 8: Finally get a full summary of your model

model_1 = sm.OLS(y, X)

result_1 = model_1.fit()

result_1.summary()

PARAMETER	VALUE
MEAN ABSOLUTE ERROR	0.0380
MEAN SQUARED ERROR	0.002
ROOT MEAN SQUARED ERROR	0.04
ACCURACY	0.56
R^2	0.56
MAX_ERRO	1

Dep. Variable:	**HML_SI**	**R-squared:**	**0.486**
Model:	**OLS**	**Adj. R-squared:**	**0.478**
Method:	**Least Squares**	**F-statistic:**	**61.28**
Date:	**Fri, 19 Jun 2020**	**Prob (F-statistic):**	**3.90E-70**
Time:	**20:20:14**	**Log-Likelihood:**	**841.41**
No. Observations:	**528**	**AIC:**	**-1665**
Df Residuals:	**519**	**BIC:**	**-1626**
Df Model:	**8**		
Covariance Type:	**nonrobust**		

coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**	
const	**-2.0598**	**0.461**	**-4.466**	**0.000**	**-2.966**	**-1.154**
COLD_BLAST_HUMID	**0.0037**	**0.001**	**6.374**	**0.000**	**0.003**	**0.005**
HML_S	**-3.6725**	**0.455**	**-8.065**	**0.000**	**-4.567**	**-2.778**
HM_TEMP	**0.0028**	**0.000**	**10.069**	**0.000**	**0.002**	**0.003**
HML_CAO_SIO2	**-0.9250**	**0.102**	**-9.044**	**0.000**	**-1.126**	**-0.724**
SLAG_RATE	**-0.0014**	**0.000**	**-7.029**	**0.000**	**-0.002**	**-0.001**
SIN_CAO_SIO2	**-0.0426**	**0.015**	**-2.810**	**0.005**	**-0.072**	**-0.013**
CORR_COKE_RATE	**0.0012**	**0.000**	**9.029**	**0.000**	**0.001**	**0.001**
HML_MGO	**-0.0347**	**0.004**	**-8.054**	**0.000**	**-0.043**	**-0.026**

Omnibus:	**3.666**	**Durbin-Watson:**	**1.517**

Prob(Omnibus):	**0.160**	**Jarque-Bera (JB):**	**3.645**
Skew:	**0.133**	**Prob(JB):**	**0.162**
Kurtosis:	**3.307**	**Cond. No.**	**3.94E+05**

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.94e+05. This might indicate that there are strong multicollinearity or other numerical problems.

LINEAR REGRESSION(BEST SUBSET)

STEPS INVOLVED IN FINDING THE BEST SUBSET :

Step 1 : Import all the required libraries required :

```
from sklearn.datasets import load_boston

import pandas as pd

import numpy as np

import matplotlib

import matplotlib.pyplot as plt
```

```
import seaborn as sns

import statsmodels.api as sm

%matplotlib inline

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.feature_selection import RFE

from sklearn.linear_model import RidgeCV, LassoCV, Ridge, Lasso
```

Step 2 : Read the dataset :

```
df = pd.read_csv("base_data2.csv")
```

Step 3 : Define X and y

```
X=df[['CORR_PRODUCTION','CORR_NC_RATE','CORR_COKE_RATE','CORR_COAL_RAT
E','HM_TEMP','QUARTZ_RATE','LIMESTONE_RATE','SLAG_RATE','PYROXINITE_RATE','
SINTER_PERC','PELLET_PERC','HOT_BLAST_TEMP','COLD_BLAST_O2_ENRICH','COLD
_BLAST_HUMID','FUR_RAFT_TEMP','COMP_COLD_BLAST_FLOW','UPTAKE_AVG','HL_T
OTAL_GJ_HR','UPPER_K','MIDDLE_K','LOWER_K','K','HML_S','HML_SIO2','HML_MGO','H
ML_AL2O3','HML_CAO_SIO2','COK_MOISTURE','COK_CSR','COK_CRI','COK_AMS','SIN_
```

CAO_SIO2','SIN_MGO']]

y = df.HML_SI

Step 4 : Use train test split from sklearn

Using train_test_split from sklearn split the dataset

Into X_train , X_test , y_train, y_test.

Into X_train , X_test , y_train, y_test.

Step 5 : Here we are using OLS model which stands for “Ordinary Least Squares”. This model is used for performing linear regression.

```
#Adding constant column of ones, mandatory for sm.OLS model

X_1 = sm.add_constant(X)

#Fitting sm.OLS model

model = sm.OLS(y,X_1).fit()

model.pvalues
```

Step 6: Using the backward elimination process.

Hence we will remove this feature and build the model once again. This is an iterative process and can be performed at once with the help of loop.

```
#Backward Elimination

cols = list(X.columns)

pmax = 1

while (len(cols)>0):

   p= []

   X_1 = X[cols]

   X_1 = sm.add_constant(X_1)

   model = sm.OLS(y,X_1).fit()

   p = pd.Series(model.pvalues.values[1:],index = cols)

   pmax = max(p)

   feature_with_p_max = p.idxmax()

   if(pmax>0.05  ):

       cols.remove(feature_with_p_max)
```

```
    else:
        break
selected_features_BE = cols
print(selected_features_BE)
print(len(selected_features_BE))
```

Step 7 : The Recursive Feature Elimination (RFE) method works by recursively removing attributes and building a model on those attributes that remain.

```
model = LinearRegression()
#Initializing RFE model
rfe = RFE(model, 7)
#Transforming data using RFE
X_rfe = rfe.fit_transform(X,y)
#Fitting the data to model
model.fit(X_rfe,y)
print(rfe.support_)
```

```
print(rfe.ranking_)
```

Step 8 : Now we need to find the optimum number of features, for which the accuracy is the highest. We do that by using loop starting with 1 feature and going up to 13. We then take the one for which the accuracy is highest.

```
#no of features

nof_list=np.arange(1,13)

high_score=0

#Variable to store the optimum features

nof=0

score_list =[]

for n in range(len(nof_list)):

   X_train, X_test, y_train, y_test = train_test_split(X,y, test_size =
0.20604, random_state = 0)

   model = LinearRegression()

   rfe = RFE(model,nof_list[n])

   X_train_rfe = rfe.fit_transform(X_train,y_train)
```

```
    X_test_rfe = rfe.transform(X_test)

    model.fit(X_train_rfe,y_train)

    score = model.score(X_test_rfe,y_test)

    score_list.append(score)

    if(score>high_score):

        high_score = score

        nof = nof_list[n]

print("Optimum number of features: %d" %nof)

print("Score with %d features: %f" % (nof, high_score))

print()
```

Step 9 : As seen from above code, the optimum number of features is 10. We now feed 10 as number of features to RFE and get the final set of features given by RFE method, as follows:

```
cols = list(X.columns)

model = LinearRegression()

#Initializing RFE model
```

```
rfe = RFE(model, 10)

#Transforming data using RFE

X_rfe = rfe.fit_transform(X,y)

#Fitting the data to model

model.fit(X_rfe,y)

temp = pd.Series(rfe.support_,index = cols)

selected_features_rfe = temp[temp==True].index

print(selected_features_rfe)
```

Step 10 : Here we will do feature selection using Lasso regularization. If the feature is irrelevant, lasso penalizes it's coefficient and make it 0. Hence the features with coefficient = 0 are removed and the rest are taken.

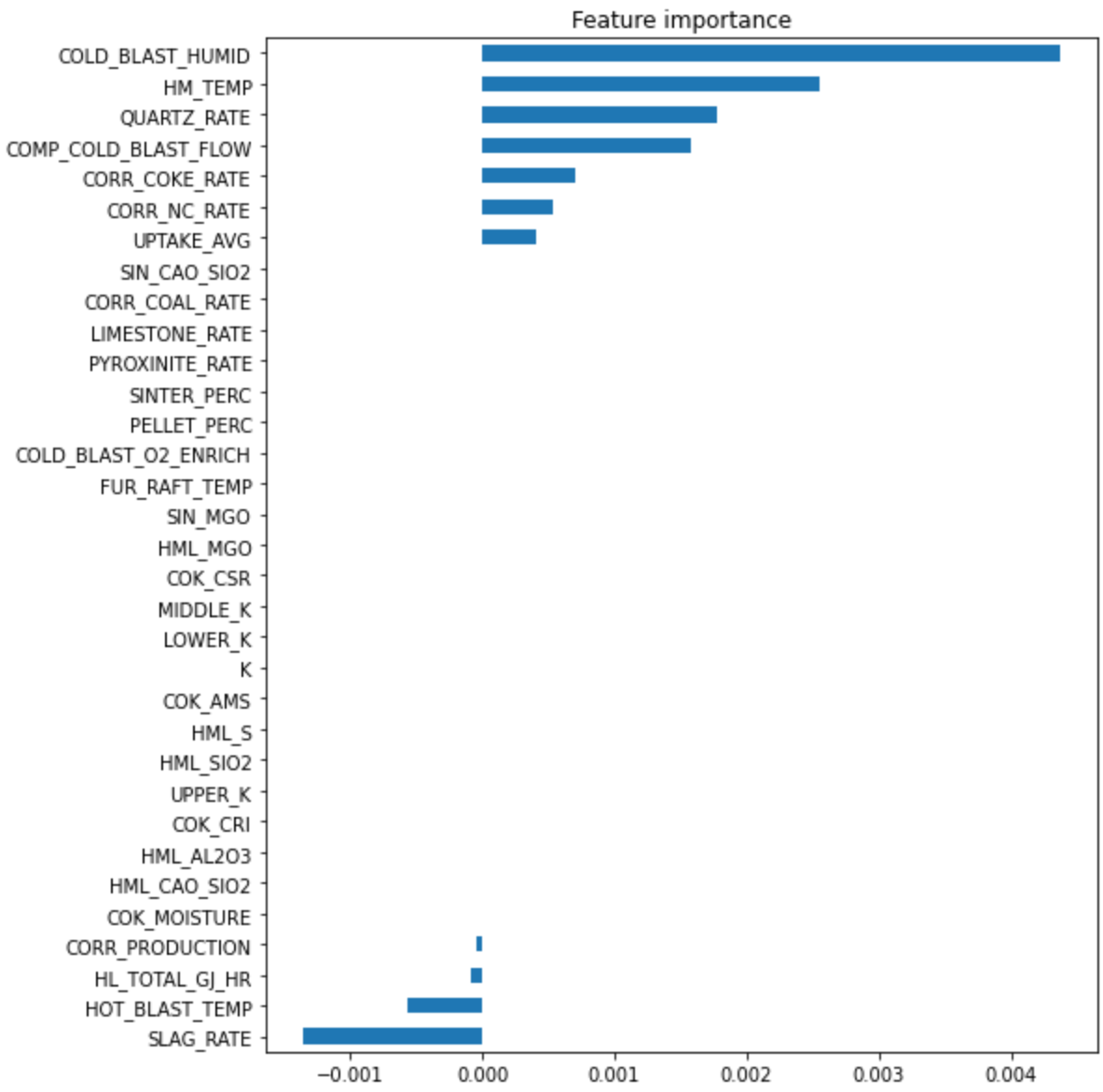
```
reg = LassoCV()

reg.fit(X, y)

print("Best alpha using built-in LassoCV: %f" % reg.alpha_)

print("Best score using built-in LassoCV: %f" %reg.score(X,y))

coef = pd.Series(reg.coef_, index = X.columns)
```

Step 11 :

```
print("Lasso picked " + str(sum(coef != 0)) + " variables and eliminated
the other " +  str(sum(coef == 0)) + " variables")
```

Step 12 : Plot the best subset

```
imp_coef = coef.sort_values()

import matplotlib

matplotlib.rcParams['figure.figsize'] = (8.0, 10.0)

imp_coef.plot(kind = "barh")

plt.title("Feature importance")
```



Step 13: Finally get a full summary of your model

model_1 = sm.OLS(y, X)

#Here X = `'HML_SI_BIN', 'UPPER_K', 'MIDDLE_K', 'LOWER_K', 'K', 'HML_S', 'HML_MGO','HML_AL2O3', 'HML_CAO_SIO2', 'SIN_CAO_SIO2'`

result_1 = model_1.fit()

result_1.summary()

Dep. Variable:	HML_SI	**R-squared:**	0.597
Model:	OLS	**Adj. R-squared:**	0.589
Method:	Least Squares	**F-statistic:**	76.48
Date:	Sat, 20 Jun 2020	**Prob (F-statistic):**	2.85E-95
Time:	08:26:42	**Log-Likelihood:**	905.56
No. Observations:	528	**AIC:**	-1789
Df Residuals:	517	**BIC:**	-1742
Df Model:	10		
Covariance Type:	nonrobust		

coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	1.2593	0.156	8.059	0.000	0.952	1.566
UPPER_K	1.0966	3.520	0.312	0.756	-5.819	8.012
MIDDLE_K	0.9797	3.520	0.278	0.781	-5.936	7.896
LOWER_K	0.9876	3.521	0.280	0.779	-5.930	7.905
K	-0.9839	3.521	-0.279	0.780	-7.902	5.934
HML_S	-2.3958	0.431	-5.564	0.000	-3.242	-1.550
HML_MGO	-0.0149	0.004	-3.558	0.000	-0.023	-0.007

HML_AL2O3	0.0126	0.004	3.502	0.001	0.006	0.020
HML_CAO_SIO2	-0.4806	0.095	-5.073	0.000	-0.667	-0.294
SIN_CAO_SIO2	-0.0255	0.014	-1.865	0.063	-0.052	0.001

Omnibus:	234.228	**Durbin-Watson:**	1.655
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2867.486
Skew:	1.599	**Prob(JB):**	0.00
Kurtosis:	13.959	**Cond. No.**	7.93E+04

2. LOGISTIC REGRESSION

In statistics, the logistic model (or logit model) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object being detected in the image would be assigned a probability between 0 and 1 and the sum adding to one.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression[1] (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent

variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1".

LOGISTIC REGRESSION ASSUMPTIONS

1.Binary logistic regression requires the dependent variable to be binary.

2.For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.

3.Only the meaningful variables should be included.

4.The independent variables should be independent of each other. That is, the model should have little or no multicollinearity.

5.The independent variables are linearly related to the log odds.

6.Logistic regression requires quite large sample sizes.

STEPS INVOLVED IN TRAINING THE MODEL :

Step 1 : import all the required libraries required :

```
import pandas as pd

import numpy as np

from sklearn import linear_model

import matplotlib.pyplot as plt
```

```
from sklearn import metrics
```

Step 2 : Read the dataset :

df = pd.read_csv("base_data2.csv")

Step 3 : Define X and y

X=df[['CORR_PRODUCTION','CORR_NC_RATE','CORR_COKE_RATE','CORR_COAL_RATE','HM_TEMP','QUARTZ_RATE','LIMESTONE_RATE','SLAG_RATE','PYROXINITE_RATE','SINTER_PERC','PELLET_PERC','HOT_BLAST_TEMP','COLD_BLAST_O2_ENRICH','COLD_BLAST_HUMID','FUR_RAFT_TEMP','COMP_COLD_BLAST_FLOW','UPTAKE_AVG','HL_TOTAL_GJ_HR','UPPER_K','MIDDLE_K','LOWER_K','K','HML_S','HML_SIO2','HML_MGO','HML_AL2O3','HML_CAO_SIO2','COK_MOISTURE','COK_CSR','COK_CRI','COK_AMS','SIN_CAO_SIO2','SIN_MGO']]

y = df.HML_SI

Step 4 : Use train test split from sklearn

Using train_test_split from sklearn split the dataset

Into X_train , X_test , y_train, y_test.

Into X_train , X_test , y_train, y_test.

Step 5 : Lets train our model and fit the dataset.

```
from sklearn.linear_model import LogisticRegression

reg = LogisticRegression()

reg.fit(X_train,y_train)
```

Step 6: Now check the accuracy of the model

print(reg.score(X_test, y_test))

Step 7: Now lets you test how well your model fits the data set.

print(metrics.mean_absolute_error(y_test, y_pred))

print(metrics.mean_squared_error(y_test, y_pred))

print(np.sqrt(metrics.mean_squared_error(y_test, y_pred)))

Step 8 : Finding the confusion matrix

CONFUSION MATRIX

The confusion matrix is a handy presentation of the accuracy of a model with two or more classes.

The table presents predictions on the x-axis and accuracy outcomes on the y-axis. The cells of the table are the number of predictions made by a machine learning algorithm.

For example, a machine learning algorithm can predict 0 or 1 and each prediction may actually have been a 0 or 1. Predictions for 0 that were actually 0 appear in the cell for prediction=0 and actual=0, whereas predictions for 0 that were actually 1 appear in the cell for prediction = 0 and actual=1. And so on.

```
from sklearn.metrics import confusion_matrix

confusion_matrix = confusion_matrix(y_test, y_pred)

print(confusion_matrix)
```

Step 8: Finally get a full summary of your model

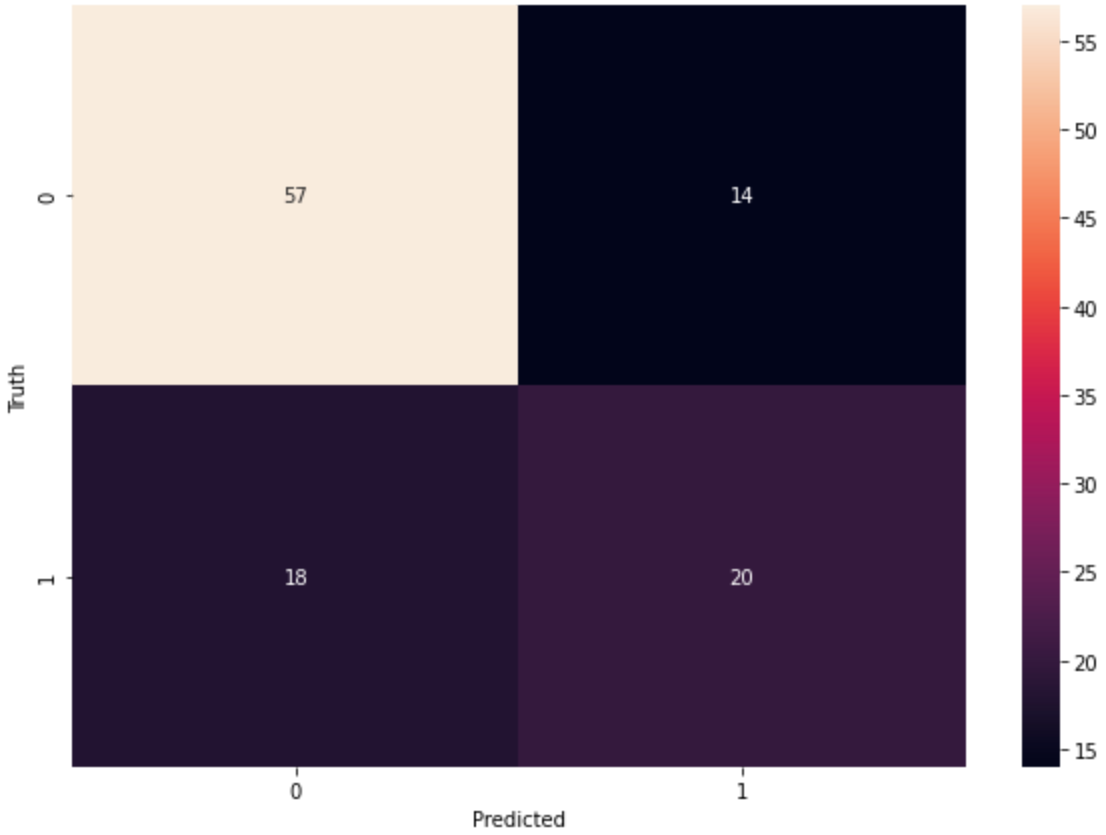
```
model_1 = sm.Logit(y, X)

result_1 = model_1.fit()

result_1.summary()
```

	PRECISION	RECALL	FL_SCORE	SUPPORT
0	0.70	0.78	0.74	69
1	0.53	0.42	0.49	40
ACCURACY			0.70	109
MACRO AVG	0.62	0.60	0.61	109
WEIGHTED AVG	0.64	0.65	0.64	109

PARAMETER	VALUE
MEAN ABSOLUTE ERROR	0.30275229357798167
MEAN SQUARED ERROR	0.30275229357798167
ROOT MEAN SQUARED ERROR	0.5502293099953707
ACCURACY	0.6972477064220184
R^2	-0.3888030888030889
MAX_ERRO	1
CONFUSION MATRIX	([[60, 14], [19, 16]])

LOGISTIC REGRESSION TRUTH VS PREDICTED

Text(69.0, 0.5, 'Truth



LOGISTIC REGRESSION (IMPORTANT PARAMETERS)

CORR_COKE_RATE	**-0.219178**
HM_TEMP	**-0.306938**
QUARTZ_RATE	**-0.153562**
SLAG_RATE	**0.127064**
COLD_BLAST_O2_ENRICH	**0.191980**
COMP_COLD_BLAST_FLOW	**-0.221053**
K	**-0.181855**
HML_S	**0.138720**
HML_MGO	**0.095085**
HML_CAO_SIO2	**0.095557**
SIN_CAO_SIO2	**0.143686**
SIN_MGO	**0.100610**
UPTAKE_AVG	**-0.110822**

COK_CRI	-0.033347

Dep. Variable:	HML_SI_BIN	No. Observations:	528
Model:	Logit	Df Residuals:	494
Method:	MLE	Df Model:	33
Date:	Sat, 20 Jun 2020	Pseudo R-squ.:	0.3797
Time:	04:15:53	Log-Likelihood:	-213.60
converged:	TRUE	LL-Null:	-344.38
Covariance Type:	nonrobust	LLR p-value:	2.240E-37

3. DECISION TREE

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

A decision tree is a flowchart-like structure in which each internal node represents a “test” on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome

of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

Decision trees are commonly used in operations research and operations management. If, in practice, decisions have to be taken online with no recall under incomplete knowledge, a decision tree should be paralleled by a probability model as a best choice model or online selection model algorithm. Another use of decision trees is as a descriptive means for calculating conditional probabilities.

Decision trees, influence diagrams, utility functions, and other decision analysis tools and methods are taught to undergraduate students in schools of business, health economics, and public health, and are examples of operations research or management science methods.

STEPS INVOLVED IN TRAINING THE MODEL :

Step 1 : Importing Required Libraries

```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
```

Step 2 : Loading the Data using pandas

```
df = pd.read_csv("base_data2.csv")
```

Step 3: Here, you need to divide given columns into two types of variables dependent(or target variable) and independent variable(or feature variables).

```
feature_cols =
['CORR_PRODUCTION','CORR_NC_RATE','CORR_COKE_RATE','CORR_COAL_RATE','HM_TE
MP','QUARTZ_RATE','LIMESTONE_RATE','SLAG_RATE','PYROXINITE_RATE','SINTER_P
ERC','PELLET_PERC','HOT_BLAST_TEMP','COLD_BLAST_O2_ENRICH','COLD_BLAST_HUM
ID','FUR_RAFT_TEMP','COMP_COLD_BLAST_FLOW','UPTAKE_AVG','HL_TOTAL_GJ_HR','
```

```
UPPER_K','MIDDLE_K','LOWER_K','K','HML_S','HML_SIO2','HML_MGO','HML_AL2O3'
,'HML_CAO_SIO2','COK_MOISTURE','COK_CSR','COK_CRI','COK_AMS','SIN_CAO_SIO2
','SIN_MGO']
X = df[feature_cols] # Features
y = df.HML_SI_BIN # Target variable
```

Step 4 : Splitting Data
To understand model performance, dividing the dataset into a training set and a test set is a good strategy.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.20604, random_state=1)
```

Step 5 : lets train our model and fit the dataset.

```
from sklearn import tree
model = tree.DecisionTreeClassifier()
model = model.fit(X_train,y_train)
```

Step 5 : lets find the y_predict and check the accuracy.

#5.1

```
y_pred = model.predict(X_test)
```

#5.2

```
from sklearn.metrics import accuracy_score
print ("Accuracy : ", accuracy_score(y_test, y_pred))
```

Step 6 : Building Decision Tree Model
Let's create a Decision Tree Model using Scikit-learn.

#6.1

```
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, y)
tree.plot_tree(clf)
```

#6.2

Create Decision Tree classifer object
clf = DecisionTreeClassifier()

#6.3

Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

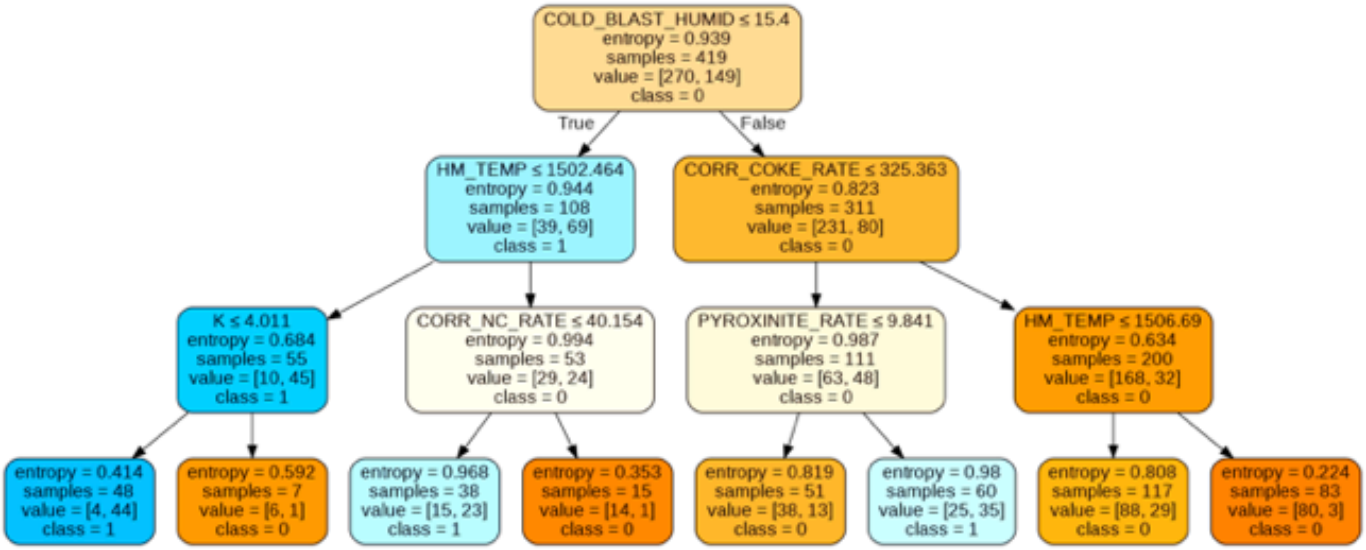
#6.4
Predict the response for test dataset
y_pred = clf.predict(X_test)

Step 7 : Visualizing Decision Trees

```
from IPython.display import Image

from sklearn.tree import export_graphviz

import pydotplus

dot_data = StringIO()

export_graphviz(clf, out_file=dot_data,
```

```
                filled=True, rounded=True,

                special_characters=True, feature_names =
feature_cols,class_names=['0','1'])

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())

graph.write_png('dt.png')

Image(graph.create_png())
```

Visualizing Decision Tree



	PRECISION	RECALL	FL_SCORE	SUPPORT
0	0.70	0.78	0.74	69
1	0.53	0.42	0.49	40
ACCURACY			0.65	109
MACRO AVG	0.62	0.60	0.61	109
WEIGHTED AVG	0.64	0.65	0.64	109

PARAMETER	VALUE
MEAN ABSOLUTE ERROR	0.34862
MEAN SQUARED ERROR	0.34862
ROOT MEAN SQUARED ERROR	0.590
ACCURACY	0.65
R^2	0.64
MAX_ERRO	1

6. VERIFY AND DEPLOY

MODEL WISE IMPORTANT PARAMETERS

Summary of Findings

a) Key observations

Following were the conclusions derived from the assessment of results and are found to be consistent with the business objectives due to the reasons mentioned.

1) Availability and Si level are clearly seen to have an inverse relationship from Availability-Si Trend. Hence availability is key focus area.

2) Each model has provided its own set of important parameters and nearly all have a few in common such as HM_TEMP, CORR_COKE_RATE, SLAG_RATE and SIN_CAO_SIO2. These findings are consistent with the current knowledge and known to be used as levers to control Si directly/indirectly. This also indicated that the models are fairly reliable in their outcomes.

3) Ranges of important parameters are also consistent with the common knowledge. Ranges were verified as per the current knowledge of direct/inverse relationship. Analysis helped to pinpoint the range of the individual parameters with respect to Low/High Silicon.

4) Other important parameters can be considered as the new findings.

a) Approved model

- All 3 models were approved.

b) Way forward

- Analysis was found to be relevant and outcome of models against testing data was also found to be consistent. The offline models should be translated into an online prediction system which can be accessed directly by the end users. This will be the scope of the subsequent project.

	LINEAR REGRESSION	LOGISTIC REGRESSION	DECISION TREE

ACCURACY	0.56	0.72	0.65
IMPORTANT PARAMETERS	SIN_CAO_SIO2	CORR_COKE_RATE	COLD_BLAST_HUIMD
	UPPER_K	HM_TEMP	HM_TEMP
	MIDDLE_K	QUARTZ_RATE	CORR_NC_RATE
	LOWER_K	COLD_BLAST_O2_ENRICH	CORR_COKE_RATE
	K	COMP_COLD_BLAST_FLOW	PYROXINITE_RATE
	HML_S	K	K
	HML_MGO	HML_S	
	HML_AL2O3	HML_MGO	
	HML_CAO_SIO2	HML_CAO_SIO2	
		SIN_CAO_SIO2	
		SIN_MGO	
		UPTAKE_AVG	
		COK_CRI	
		HML_SI_BIN	

CONCLUSION

Out of the three models , model on decision tree was accepted to control Hot metal Silicon

The reasons are :

- It has the most reasonable accuracy.

- The relevance of it parameters in terms of impact and the number.

-It gives the least controlling levers and is actionable

-It gives branching at each value and thus helping us obtain which parameter needs to be tweaked so that we can control silicon.

-The outcome is easily communicable with the line manager as well as the operator.