A project report on

IRON & STEEL QUALITY & CONSUMPTION PREDICTION USING MACHINE LEARNING

Submitted in partial fulfillment for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

by

KASHISH (19BCE1760)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2023

IRON & STEEL QUALITY & CONSUMPTION PREDICTION USING MACHINE LEARNING

Submitted in partial fulfillment for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

by

KASHISH (19BCE1760)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2023

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

DECLARATION

I hereby declare that the thesis entitled “IRON & STEEL QUALITY & CONSUMPTION PREDICTION USING MACHINE LEARNING” submitted by me, for the award of the degree of Bachelor of Technology in Computer Science and Engineering, Vellore Institute of Technology, Chennai, is a record of bonafide work carried out by me under the supervision of Guide Name

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Chennai

Date: 24.04.23 Signature of the Candidate

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled **"Iron & Steel Quality & consumption prediction using machine Learning"** is prepared and submitted by **Kashish** (**19BCE1760**) to Vellore Institute of Technology, Chennai, in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering programme** is a bonafide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Signature of the Guide:

Name:

Date:

Signature of the Examiner 1	Signature of the Examiner 2
Name:	Name:
Date:	Date:

Approved by the Head of Department
B. Tech. CSE

Name: Dr. Nithyanandam P

Date: 24 – 04 – 2023

(Seal of SCOPE)

ABSTRACT

In order to increase its strength and malleability in comparison to other kinds of iron, steel is an alloy of iron combined with a few tenths of carbon. The efficient forecast of steel production plays a critical role in streamlining the steel preparation process and minimizing energy and waste. Steel manufacturing accounts for a sizeable portion of the world's energy consumption. In this research, we investigate how machine learning techniques can be used to forecast steel energy usage. We examine the pertinent literature on the subject and evaluate the predictive power of different machine learning algorithms, such as multi-regression, decision trees, and KNN. To train and assess these algorithms, I used a dataset of historical steel production and energy consumption data. Additionally, I have looked into how various feature engineering techniques, including feature scaling and feature selection, affect how well the algorithms predict outcomes. My findings show that machine learning algorithms can estimate the energy consumption of steel with high accuracy, with some algorithms doing better than others. Additionally, I've outlined the critical elements that have a large impact on steel's energy use and offer insights into how machine learning may be used to streamline the manufacturing process and use less energy. The promise of machine learning for predicting steel energy consumption is highlighted by my findings, which also offer future research areas for increasing prediction accuracy and putting machine learning models to use in the steel sector. Mining quality expectations for iron metal are a fundamental task in the mining industry because they have a significant impact on the efficiency and value of mining tasks. Conventional quality expectation solutions usually rely on manual review and examination, which can be time-consuming, work-escalated, and subject to human biases. In this research, we look into the use of AI techniques to predict the makeup of iron minerals during mining operations.

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to Mr. Sridhar Ranganathan,
Associate Professor, SCOPE, Vellore Institute of Technology, Chennai, for his constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavor. My association with him is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Web mining.

It is with gratitude that I would like to extend thanks to our honorable Chancellor, Dr. G. Viswanathan, Vice Presidents, Mr. Sankar Viswanathan, Dr. Sekar Viswanathan and Mr. G V Selvam, Assistant Vice-President, Ms. Kadhambari S. Viswanathan, Vice-Chancellor, Dr. Rambabu Kodali, Pro-Vice Chancellor, Dr. V. S. Kanchana Bhaaskaran and Additional Registrar, Dr. P.K. Manoharan for providing an exceptional working environment and inspiring all of us during the tenure of the course.

Special mention to Dean, Dr. Ganesan R, Associate Dean Academics, Dr. Parvathi R and Associate Dean Research, Dr. Geetha S, SCOPE, Vellore Institute of Technology, Chennai, for spending their valuable time and efforts in sharing their knowledge and for helping us in every aspect.

In jubilant mood I express ingeniously my whole-hearted thanks to Dr. Nithyanandam P, Head of the Department, Project Coordinators, Dr. Abdul Quadir Md, Dr. Priyadarshini R and Dr. Padmavathy T V, B. Tech. Computer Science and Engineering, SCOPE, Vellore Institute of Technology, Chennai, for their valuable support and encouragement to take up and complete the thesis.

My sincere thanks to all the faculties and staff at Vellore Institute of Technology, Chennai, who helped me acquire the requisite knowledge. I would like to thank my parents for their support. It is indeed a pleasure to thank my friends who encouraged me to take up and complete this task.

Place: Chennai

Date: Name of the student

CONTENTS

CHAPTER 1

INTRODUCTION

CHAPTER 2

LITERATURE REVIEW

CHAPTER 3

ALGORITHM USED

CHAPTER 5

IMPLEMENTATION AND RESULT

CHAPTER 6

CONCLUSION AND FUTURE WORK

::

::

::

LIST OF FIGURES

LIST OF ACRONYMS

Abbreviation	Expansion
CRC	Cold Rolled Coil
TISCO	Tata Iron & Steel Enterprise
KNN	K-nearest Neighbor
MLR	Multiple Linear Regression
DT	Decision Tree
RF	Random Forest
LR	Linear Regression

CHAPTER 1

INTRODUCTION

1.1 WHAT IS STEEL?

Steel is a compound comprised of iron with regularly a couple of tenths of a percent of carbon to work on its solidarity and break opposition contrasted with different types of iron. It is a flexible and generally utilized material known for its solidarity, strength, and capacity to be molded into different structures. Steel is usually utilized in development, car fabricating, hardware, machines, and a large number of different applications because of its helpful mechanical properties, including high elasticity, sturdiness, and erosion obstruction. The production of steel involves melting iron ore in a furnace and adding carbon and other alloying elements to achieve the desired properties. Steel can be additionally handled through different strategies like rolling, manufacturing, and projecting to acquire the ideal shapes and sizes for various applications. The organization and handling of steel can be shifted to create various grades and kinds of steel with particular properties, making it a flexible material utilized in a great many ventures.

1.2 WHAT IRON?

Iron is a chemical element with the symbol Fe (from the Latin word "ferrum") and atomic number 26. It is a transition metal, belonging to the group 8 of the periodic table. Iron is one of the most abundant elements on Earth, making up a significant portion of the Earth's crust.

Iron has been utilized by people for millennia and is a fundamental component of different organic cycles. It is a major area of strength for a moldable metal with a gleaming dim appearance when newly cleaned. Iron is known for its high softening and edges of boiling over, as well as it's capacity to direct intensity and power.

Iron has numerous applications in various organizations, including advancement, auto, flight, and collecting. It is ordinarily used in the improvement of steel, which is a compound of iron and carbon and is comprehensively used in the improvement of designs, frameworks, vehicles, and equipment. Iron is moreover used in the advancement of devices, equipment, and machines, as well as in the creation of electrical parts and stuff.

Iron is gotten from iron minerals through a cycle called iron purifying, where the minerals are warmed in a heater with a decreasing specialist to remove the iron metal. Iron minerals are tracked down in different land arrangements and are mined in numerous nations all over the planet. The creation and utilization of iron have critical financial, social, and natural ramifications, and iron mining and handling are dependent upon severe guidelines and ecological contemplations.

1.3 BYPRODUCTS OF STEEL & IRON

Iron and steel have been produced by humans for a very long period. Steel was a major factor in the modern revolution and continues to be the cornerstone of modern industrialized economies. It's challenging to imagine a world without steel because it is always present in everything we manufacture and do, from construction to transportation to machinery. Steel is unusual from other materials because to its versatility in terms of both its form and qualities, its solidarity to weight ratio, and its ability to be greatly multicycle into new products, all of which have contributed to its ongoing success.

There are various products that are made by steel some of which are listed below:

a. Hot rolled coils

b. Hot rolled plates

c. Hot rolled sheets

d. Cold rolled coils (CRC)

e. Cold rolled sheets

f. Tin mill black plates

g. Galvanized plain and corrugated sheets

h. Oxygen gas

i. Hydrogen gas

j. Coke oven by-products

k. Steel Bars & Rods

l. Alloys

The production of iron and steel involves various processes that may result in the generation of by-products or secondary products. Some common by-products of iron production include:

a. Slag

b. Dust and Fumes

c. Scale

d. Sludge

e. Gas

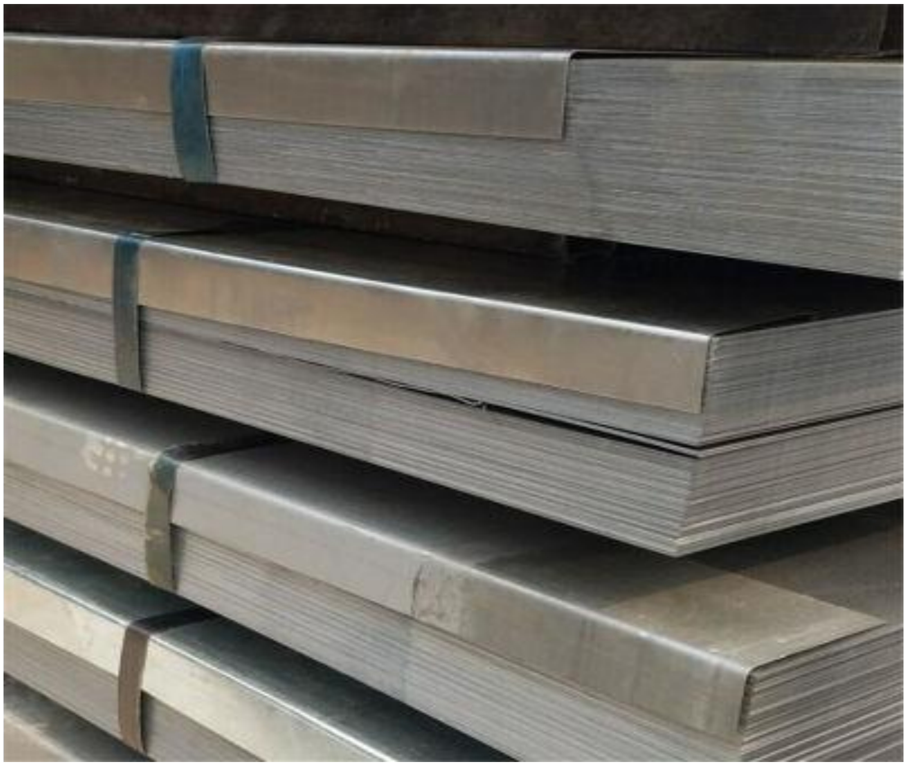


fig 1.1

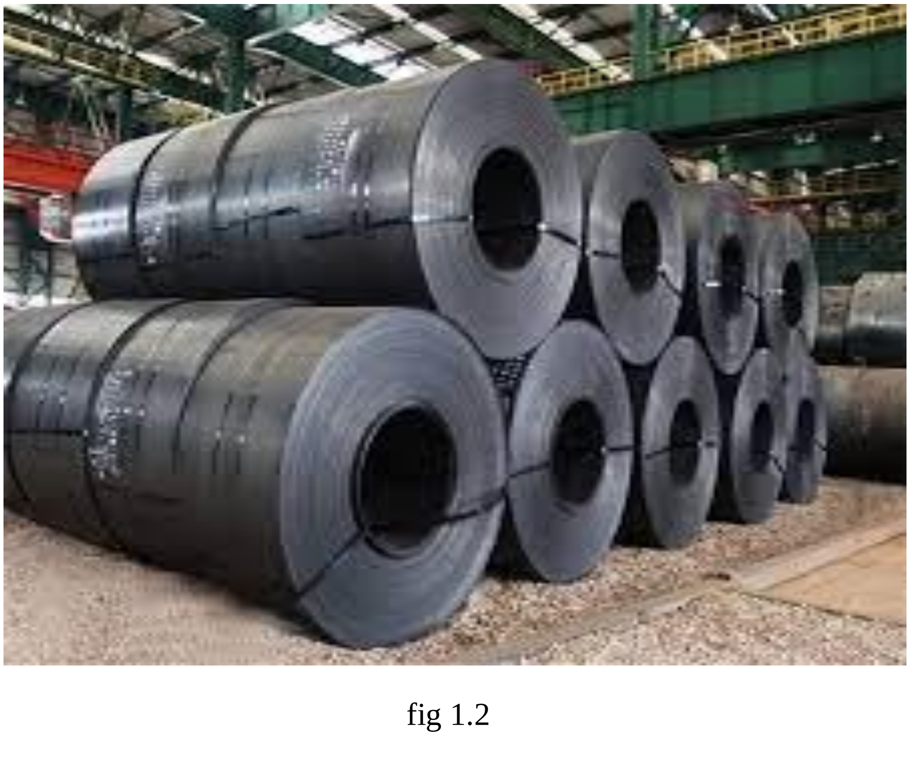


fig 1.2

1.4 ADVANTAGE

Steel is one of the most essential materials. It is fundamental for the growth of any nation as it forms the backbone of industrialization. The demand for metal comes ordinarily from the infrastructure, vehicle, and patron durables industry and the fortunes of metal are especially correlated with these user industries. Steel manufacturing in India started out with the setting up of TISCO in 1907. Knowledge of Steel Consumption and the amount of its requirement in various places will give the Steel industry a major breakthrough. Steel Companies then work according to the needs and consumptions of steel in various places to maximize their profits. Energy consumption predictions for industry gain an important place in the energy management and control system because of dynamic and seasonal changes in energy demand and supply. Predicting the quality of iron ore mining can provide numerous advantages to mining companies, including improved resource planning, enhanced productivity and yield, better quality control and compliance, increased market competitiveness, environmental sustainability, and enhanced decision-making. These advantages can contribute to improved operational efficiency, cost savings, profitability, and sustainability in the mining industry.

1.5 USAGE OF MACHINE LEARNING

The usage of machine learning in the metal industries can provide significant advantages, including improved energy efficiency, enhanced predictive maintenance, increased product quality, process optimization, faster decision-making, and reduced environmental impact, leading to more efficient and sustainable steel production processes.

1.6 DISADVANTAGES

While machine learning has the potential to offer significant benefits in the steel and steel energy industries, there are also challenges and potential problems associated with its usage. These challenges include data availability and quality, model interpretability, model robustness and generalization, expertise and workforce, cost and infrastructure, and ethical considerations. Addressing these challenges effectively is crucial for the successful implementation of machine learning solutions in the steel and steel energy industries. Predicting the quality of iron ore mining can offer several advantages, there are also potential disadvantages to consider, including the complexity of predictive models, uncertainty and variability of ore quality, reliance on data accuracy and availability, regulatory and legal considerations, human factors and subjectivity, and operational and economic considerations. It is important to carefully address these challenges and considerations in the development, implementation, and utilization of predictive models for iron ore quality prediction.

1.7 PROBLEM STATEMENT

Steel is a mixture of iron with typically a few tenths of a percent of carbon to improve its tensile strength and remove blockage compared to other types of iron.

The mining industry considers the iron mineral quality forecast to be a fundamental task because it plays a significant role in determining the utility and value of the metal that is removed.

Operational expenses are decreased, product quality is raised, and income is constantly increasing with the application of ML in the metal business. By reusing the data and patterns that have previously been collected, digital technologies can also enhance the models that now estimate and anticipate occurrences.

The problem statement is to develop a machine learning model to analyse and forecast steel consumption in diverse contexts, including geographical regions, steel type, energy usage, and also determining the mining quality of iron ore.

1.8 SCOPE OF THE PROJECT

This project is a prototype which when implemented on a large scale which will give a boost to the steel industry by letting the producer know how much end product (Steel & it’s by products and iron) has to be made to meet the needs of the different countries. It will also help in increasing the product quality and help in optimizing the process of steel making. It will also reduce Environmental impact by optimizing energy usage and production process through machine learning.

CHAPTER 2

LITERATURE REVIEW

2.1 RESEARCH OVERVIEW

This segment intends to provide information about the undertaken research related to the project. The chapter includes researches from various sources about varied topics needed to learn about the project. It consists knowledge about varied topics needed to learn about the project. It consists of knowledge about various machine learning algorithms used in steel making and its by-products and iron ore quality management explains some of the applications of the same.

2.2 EXISTING TECHNOLOGICAL SOLUTION

The incorporation of Machine Learning into these kinds of ecosystems has recently piqued the interest of various works. However, very few studies have focused on how machine learning technology may assist in fulfilling the requirements. In this section, we will look at a few of the works that aim to achieve this integration and demonstrate how ML can help in achieving better various results in this field.

Dianmin Z. proposed a system that forecasts the daily electricity consumption of a large steel corporation [7]. In this study, by using Non negative least square method (NNLS) and Random Approximated Greedy search method (RAGS) it is observed that these algorithms are stable and the forecasting accuracy is significantly improved.

Malanichev A.G. proposed a system that shows a regression model for forecasting global average annual steel prices [8].

Sharbanti P. proposed a system in which the performance of the industry in terms of production, consumption and foreign trade is depicted [10]. This study also exhibits the trends of the industry for a period of twenty year since 1991-92 to 2010-11. A number of graphical visualizations and CAGR i.e., Compound annual growth rate was observed to state that India has all potential to become top producer of steel in near future.

Yanni X. proposed a system that observes changes in crude steel production, steel scrap consumption per ton steel, and steel scrap consumption from 1980 to 2012 [9]. This study, used IPAT model to serve the purpose.

Shailendra K.C. proposed a system to review and compare the growth prospects for Indian Steel industry in pre and post liberalized context and to know the impact of Indian Government policies and FDI on growth prospects of Indian Steel industry [11]. Various graphical charts were used in this study.

Popli G. S. proposed a system that aims to find out the scope and opportunity for attracting foreign direct investment in the steel sector in India and to understand the concern, overall requirements, modernization and complete perception of the manufacturers of steel sector in India [12].

Leo S.C. proposed a system to investigate if supervised Machine Learning models can be practically useful when applied in the context of steel processes [13]. In this study, Artificial Neural Network (ANN), Random Forest (RF) and Multivariate Linear Regression (MLR) is used to show that ML plays a significant role in steel industry.

Giacomo P. proposed a system that works on Artificial Intelligence to provide a breakthrough innovation in the sector of steel and energy [14]. In this study, AI was used to show that Steel making industry presents some very specific requirements which pose some technological challenges to the adoption of AI, limiting the spread of such Technologies.

In order to solve the filtering of data using non-predictive characteristics and feature ratings, Moonyong L. presented a method to show and analyse the predictive models of the data-driven system for the uses of appliances [15]. In this work, the most significant product, Skelp, is determined using the general linear regression model (GLM), support vector machine with the radial kernel (SVM RBF), and boosting tree (BT).

Sarawatula S. A. proposed a system in which statistical and ML techniques are used to model energy consumption [16]. In this study, MLR, DT, RF & XG Boost are used to benchmark the energy consumption of factories and identifying opportunities to improve energy efficiency.

Zhao J. proposed a system to focus on the sound energy scheduling and allocation which is of paramount significance for the current steel industry [17]. In this study, Fuzzy C-means clustering method is used to meet practically useable predictions.

Zhang Y. proposed a system to predict the energy consumption level of ironmaking process [18]. In this study, Support vector machine & PSO is used for prediction. Particle swarm optimization (PSO) was introduced to optimize the parameters of SVM.

On the basis of the grey system theory, Zeng B. suggested a system to derive and build a homologous grey prediction model with one variable and one first order equation (HGEM (1,1)) for forecasting the overall energy consumption of Chinese manufacturing [19]. The results of this study show that although Chinese manufacturing's overall energy usage is slowing, it is still too high.

A system to forecast energy usage in production at various scales was put forth by Reimann J. [20]. This study employs a tree-based compositional technique, which encourages arbitrary levels based on the structure of the machine or outside factors, such as corporate rules. This technique is very extendable because the designs are contained in ontologies.

Sen P. suggested a method for predicting GHG emissions and energy use [21]. The pig iron manufacturing organisation in India's greenhouse gas emissions and energy consumption are forecasted in this study using Autoregressive Integrated Moving Average (ARIMA).

P. Karthigaikumar proposed a study for quality expectation and examination utilizing the LISP assessment [22]. In this review, both single and multi-point approaches are talked about while thinking about MMP attributes. The LISP mining and arrangement process is utilized for building a quality expectation model. The MMP property is addressed in an effective way by playing out a suitable choice of a list of capabilities and highlights so that the relationship between the effect on the eventual outcome and the workstations are dissected by utilizing classifier learning.

Chapter 3
ALGORITHMS USED

3.1 OVERSIGHT

Global energy consumption is significantly influenced by the manufacture of steel and the quality of the iron mined, and accurate predictions of these energy requirements can be vital for streamlining the production of steel and iron while minimizing energy waste. Operational expenses are decreased, product quality is raised, and income is constantly increasing with the application of ML in the metal business. By reusing the data and patterns that have previously been collected, digital technologies can also enhance the models that now estimate and anticipate occurrences. This makes Python one of the most popular programming languages. You'll learn how technology can enhance the manufacture of steel in this article.

3.2 KNN

The supervised machine learning technique known as KNN, or "k-nearest neighbors," is used for classification and regression applications. It is a form of memory-based or instance-based learning technique in which the model automatically learns from the training data by storing the full dataset in memory.

Finding the k closest neighbors to a new data point in the training dataset using a distance metric (like Euclidean distance) or similarity measure yields the prediction for a new data point in a KNN. The majority class or average value of the k closest neighbors determines the anticipated class or value for the new data point.

KNN is an easy-to-understand technique that may be applied to both regression and classification tasks. It can handle non-linear relationships and makes no assumptions about the distribution of the underlying data. Since KNN does not require an explicit training phase, it is computationally efficient throughout training. KNN is also a lazy learner.

Using an example, we can better comprehend KNN. Assume we have a picture of a flower that resembles both a rose and a marigold. Therefore, since the KNN method uses a similarity measure, it can be used for this identification. In order to classify the new data into the rose or marigold category, our KNN model compares the new data's features to those in the rose and marigold images.

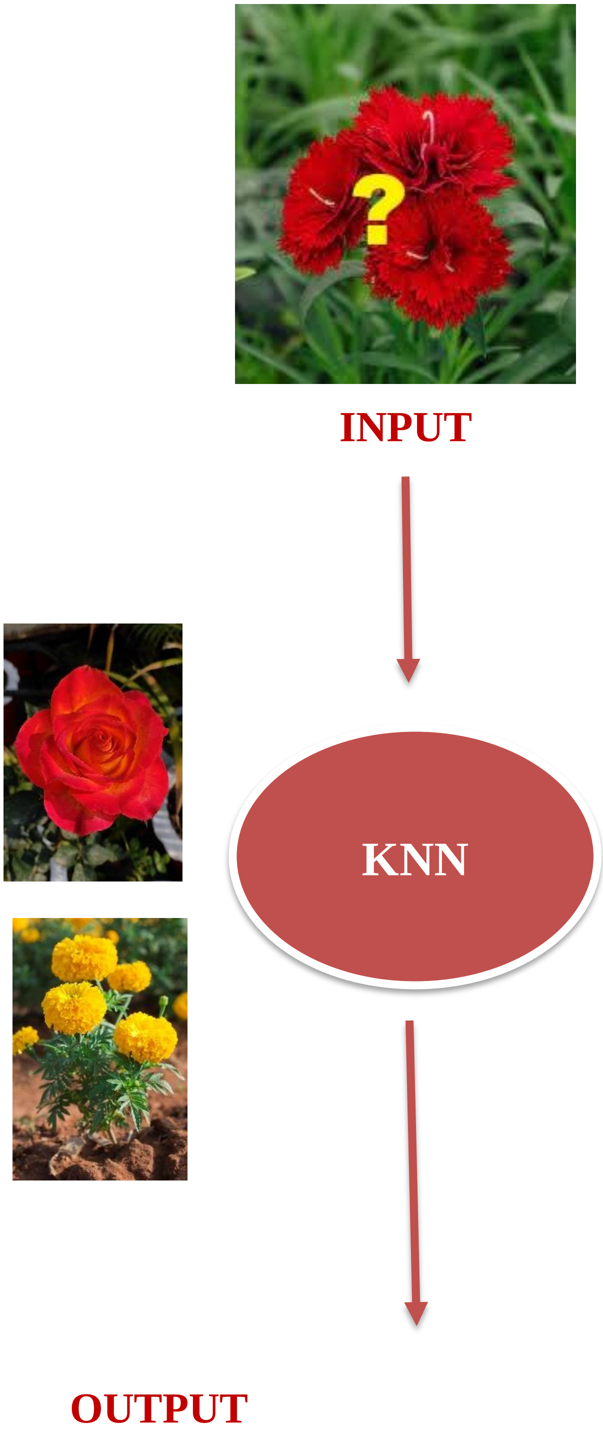


Fig 3.2.1: EXAMPLE OF KNN

3.2.1 HOW TO CHOOSE THE VALUE OF K

The following are a few focuses to recall while choosing the value of K in the K-NN calculation:

- There is no specific method for deciding the best value for "K", so we want to attempt certain values to find the best out of them. The most favored value for K is 5.
- An exceptionally low value for K like K=1 or K=2, can be uproarious and lead to the impacts of exceptions in the model.
- Huge values for K are great, however, it might discover a few troubles.

3.2.2 ADVANTAGES OF KNN

K-nearest neighbors (KNN) has several advantages as a machine learning algorithm:

a. Simplicity: KNN is a simple and intuitive algorithm that is easy to understand and implement. It does not require explicit model training, making it a lazy learner, and thus does not require time-consuming training phase.

b. No assumptions about data distribution: KNN does not make any assumptions about the underlying data distribution, making it suitable for both linear and nonlinear relationships in the data. It can handle complex patterns in the data without assuming any specific model structure.

3.2.3 DISADVANTAGES OF KNN

K-nearest neighbors (KNN) also has some limitations as a machine learning algorithm:

a. Computational efficiency: KNN can be computationally expensive, especially for large datasets, as it requires calculating the distances between the new data point and all the training data points during prediction. This can bring about more slow forecast times, particularly while managing high-layered information.

b. Sensitivity to hyperparameters: The performance of KNN is heavily dependent on the choice of hyperparameters, such as the value of k (number of neighbors) and the distance metric used. Poorly chosen hyperparameters can lead to suboptimal results. Selecting the optimal hyperparameters can be challenging and may require experimentation and tuning.

3.3 MULTIPLE LINEAR REGRESSION

There two types of linear regression:

1. Simple Linear Regression: The linear regression model gives a sloped straight line addressing the connection between the factors. Consider the picture:

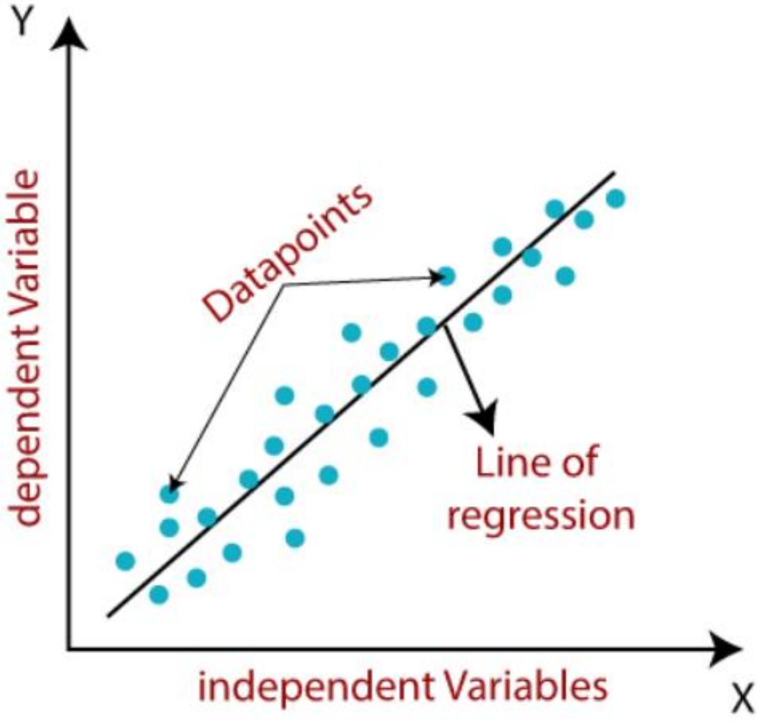
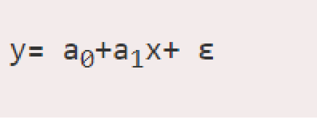


Fig 3.3.1

Mathematically, we can represent linear regression as:



$$y= a_0+a_1x+ \varepsilon$$

Y= Dependent Variable

X= Independent Variable

a0= intercept of the line

a1 = Linear regression coefficient

ε = random error

2. Multiple Linear Regression: MLR is a factual strategy used to demonstrate the connection between at least two independent factors (predictor factors) and a reliant variable (response variable). It broadens the idea of basic straight relapse, which models the connection between one free factor and a reliant variable, to consolidate various autonomous factors.

In MLR, the objective is to track down a direct condition that best fits the noticed information by assessing the coefficients (otherwise called relapse coefficients or model boundaries) for every independent factor. The assessed coefficients address the strength and course of the connection between the independent factors and the reliant variable. The MLR condition can then be utilized to make predictions of the reliant variable based on the independent variables.

Mathematically, the MLR model can be represented as:

$Y = \beta0 + \beta1X1 + \beta2X2 + ... + \beta p*Xp + \varepsilon$

where:

Y is the dependent variable (response variable) that is being predicted, β0 is the intercept or constant term,

β1, β2, ..., βp are regression coefficients associated with the independent variables X1, X2, ..., Xp, respectively,

X1, X2, ..., Xp are the independent variables (predictor variables) used in the model,

ε is the error term that represents the residual or unexplained variability in the dependent variable,

p is the number of independent variables in the model.

The reason for MLR is to comprehend the values of the regression coefficients that best describe the association between the independent factors and the reliant variable. This assessment is regularly done utilizing different factual procedures, like ordinary least squares (OLS), maximum likelihood estimation (MLE), or gradient descent, among others. When the coefficients are assessed, they can be utilized to make predictions of the reliant variable for new observations with known values of the independent variables.

3.3.1 TYPES OF SLOPE

A regression line, also known as a fitted line or a line of best fit, is a straight line that represents the relationship between two variables in a scatter plot. It is used in regression analysis, a statistical technique used to model the relationship between a dependent variable and one or more independent variables.

There are several different types of slopes, depending on the specific application or context. Some common types of slopes include:

a. Positive slope: A positive slope indicates that as the value of one variable increases, the value of another variable also increases. In a graph or scatter plot, a positive slope is represented by a line that rises from left to right.

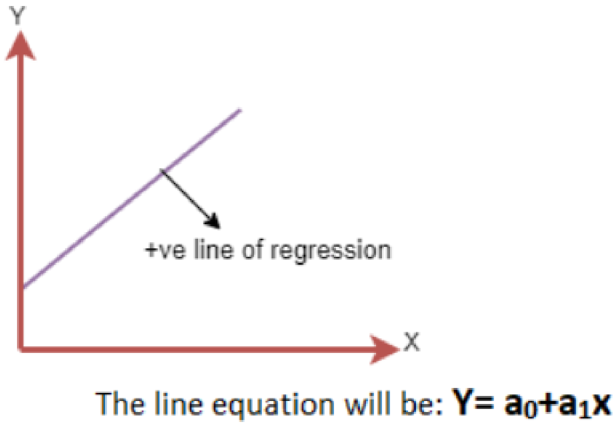


Fig 3.3.1.1

b. Negative slope: A negative slope indicates that as the value of one variable increases, the value of another variable decreases. In a graph or scatter plot, a negative slope is represented by a line that falls from left to right.

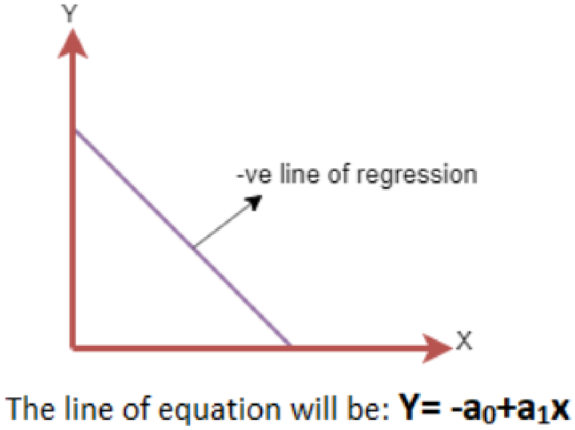


Fig 3.3.1.2

3.3.2 FINDING THE BEST FIT LINE

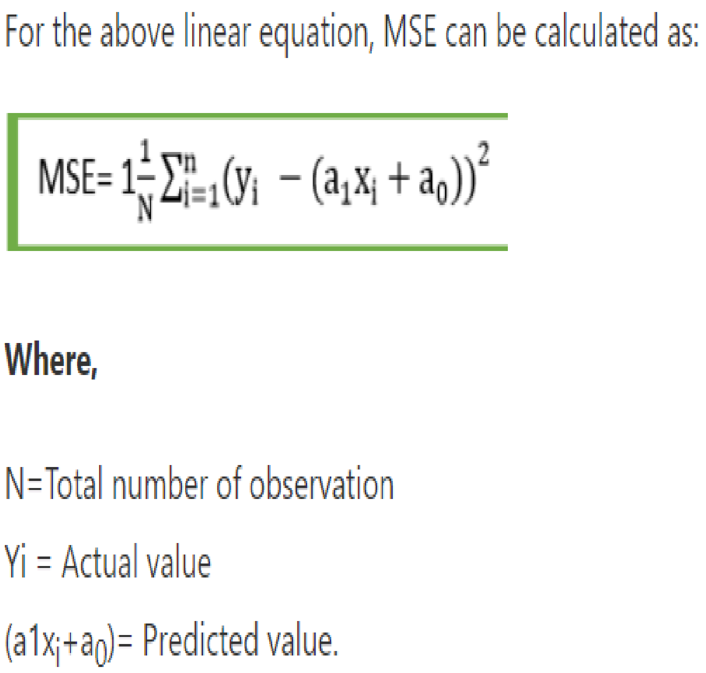
Finding the best-fit line is the goal of linear regression. Using the cost function to identify the best-fit line in a linear regression model is described as follows:

A. Describe the cost function. The cost function is a mathematical function that measures the inaccuracy or disparity between the values of the dependent variable that are predicted and the actual values. The cost functions that can be employed in linear regression include the mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and others. It is often stated as a mathematical equation. The particular problem at hand and the demands of the analysis determine the cost function to be used.

B. Initialize the coefficients: The linear regression model has two coefficients - the intercept (often denoted as β0) and the slope (often denoted as β1) - that determine the position and orientation of the regression line. These coefficients are initially assigned random values or set to zero.

C. Compute the predicted values: Using the initial values of the coefficients, the predicted values of the dependent variable (Y) are computed using the linear regression equation Y = β0 + β1*X, where X is the independent variable.

D. Calculate the cost: The error or disagreement between the anticipated and actual values of the dependent variable is then calculated using the cost function. The common method for doing this is to calculate the average error by taking the difference between the anticipated and actual values, squaring them (in the case of MSE or RMSE), adding them all up, and dividing by the total number of data points.



For the above linear equation, MSE can be calculated as:

$$\boxed{\text{MSE}= 1\frac{1}{N}\sum_{i=1}^{n}(y_i \;-(a_1x_i + a_0))^2}$$

Where,

N=Total number of observation

Yi = Actual value

$(a1x_i+a_0)$= Predicted value.

Fig 3.3.2.1

E. Update the coefficients: The next step is to update the values of the coefficients to minimize the cost function. This is normally done utilizing an enhancement calculation, for example, gradient descent, which iteratively changes the coefficients toward the steepest decline in the cost function. The greatness of the not entirely set in stone by the learning rate, which is a hyperparameter that controls the step size of the improvement calculation.

F. Repeat steps 3 to 5: Steps 3 to 5 are repeated iteratively until a convergence criterion is met, such as a maximum number of iterations or a small change in the cost function between iterations. At the end of the iterations, the final values of the coefficients that minimize the cost function are obtained.

G. Evaluate the model: Once the coefficients are obtained, the final regression line is defined by the values of the intercept and slope. The model can be evaluated using various performance metrics, such as R-squared, adjusted R-squared, and other relevant metrics, to assess the goodness of fit and predictive accuracy of the model.

3.3.3 GRADIENT DESCENT

Gradient descent is an optimization algorithm commonly used in machine learning, including linear regression, to find the optimal values of the coefficients (intercept and slope) that minimize the cost function or the prediction error. It is an iterative algorithm that adjusts the values of the coefficients in the direction of the steepest decrease in the cost function, until a convergence criterion is met.

Gradient descent is an efficient and widely used optimization algorithm for finding the optimal coefficients in a linear regression model. However, it may have some limitations, such as sensitivity to the choice of learning rate, potential convergence to local optima, and the need for proper initialization and regularization techniques to avoid overfitting. Therefore, careful tuning and experimentation may be necessary to achieve optimal performance in practice.

3.3.4 MODEL PERFORMANCE

There are a few normal strategies for assessing the exhibition of a linear regression model:

a. Mean Squared Error (MSE): MSE is the normal of the squared contrasts between the actual values and the predicted values. It estimates the average squared error between the predicted values and the actual values. Lower MSE values indicate better model performance, with zero indicating a perfect fit. MSE is widely used as a primary evaluation metric for linear regression models.

b. Root Mean Squared Error (RMSE): RMSE is the square root of MSE, which gives the error in the original units of the dependent variable. It is a widely used metric for evaluating the performance of linear regression models, as it is interpretable in the same units as the dependent variable.

c. R-squared (R2) coefficient: R2 is a proportion of how well the linear regression model fits the information, with a worth somewhere in the range of 0 and 1. It addresses the extent of the all-out variety in the reliant variable that is made sense of by the model.

3.3.5 ASSUMPTIONS OF LINEAR REGRESSION MODEL

- Linearity: The dependent variable and independent variables should have a straight line connection. This suggests that the dependant variable's adjustment corresponds to the autonomous variable(s)'s adjustment.
- Error independence: The model's errors (also known as residuals) should be independent, which means they shouldn't exhibit any regular patterns or connections. This presumption guarantees that each observation makes a separate contribution to estimating the model parameters.
- Homoscedasticity: The model's errors must have a constant variance, which means that they must have the same variance at all levels of the independent variable(s). By making this supposition, the model's prediction errors are guaranteed to be consistent and unaffected by the quantities of the independent variable(s).
- Multicollinearity: If the independent variables are not perfectly correlated with one another, there should be little to no multicollinearity among them. It might be challenging to interpret the specific impacts of each independent variable when there is high multicollinearity since it can result in unstable estimations of the model parameters.
- Error normality: The model's mistakes ought to be normally distributed, which means that they ought to adhere to a bell-shaped normal distribution. For the computation of confidence intervals and hypothesis tests related to the model parameters, this assumption is crucial.
- Homoscedasticity of errors: Across all levels of the independent variable(s), the errors' variance should be constant. This presumption guarantees that the model's prediction errors have a stable range and are unaffected by the

quantities represented by the independent variable(s).

- Outliers: The existence of outliers in the data can have a substantial impact on estimations of the parameters of the model and lower the model's reliability. Before attempting to train a linear regression model, it is crucial to locate and deal with outliers in the data.

3.3.6 ADVANTAGES OF MLR

- Capturing complex relationships: MLR allows for capturing complex relationships between a dependent variable and multiple independent variables. It can account for the joint effects of multiple predictors on the outcome variable, which may not be possible with simple linear regression.

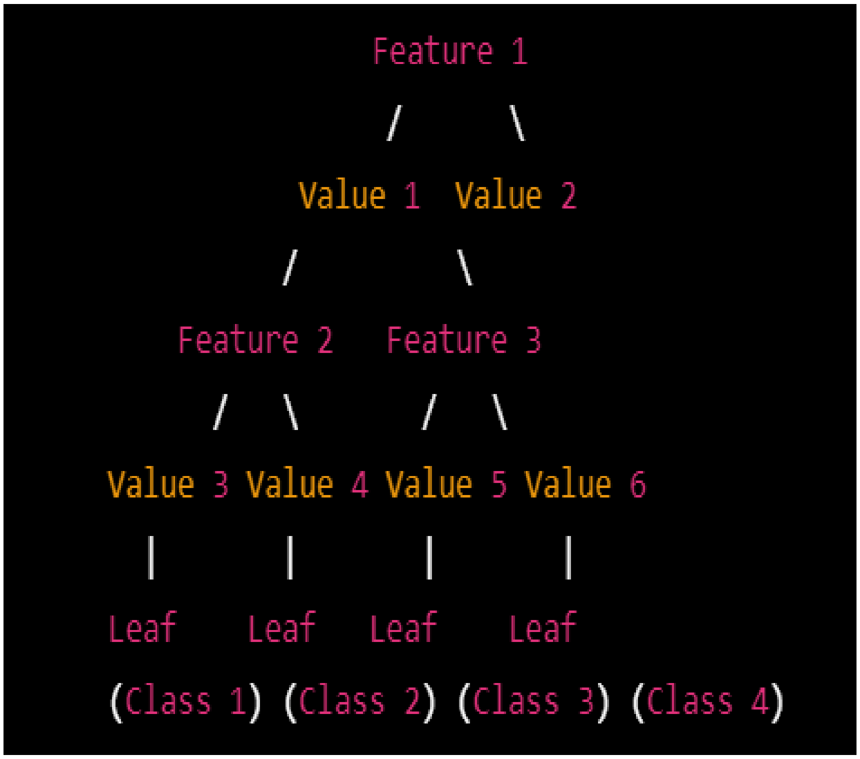
- Improved prediction accuracy: MLR can potentially result in improved prediction accuracy compared to simple linear regression when there are multiple predictors that contribute to the dependent variable. By considering multiple predictors, the model can capture the combined effect of all relevant variables, leading to more accurate predictions.

3.3.7 DISADVANTAGE OF MLR

- Assumptions: MLR relies on several assumptions, such as linearity, independence of errors, homoscedasticity, and normality of errors. Violation of these assumptions can lead to inaccurate or unreliable results. It is important to carefully assess and address these assumptions before interpreting the results of a MLR model.
- Overfitting: MLR models with a large number of predictors can be prone to overfitting, where the model may fit the noise in the data rather than the true underlying relationships. This can lead to poor generalization performance and reduced model accuracy when applied to new data.

3.4 DECISION TREE ALGORITHM

A well-liked supervised machine learning method for both classification and regression applications is the decision tree algorithm. Up until the subsets are completely or almost completely pure with regard to the target variable, it works by recursively dividing the data into subsets depending on the values of input features. These splits are then utilised by the decision tree to generate predictions or choices. Here is a simple graph that shows how a choice tree is fundamentally built:



```
               Feature 1
                /      \
           Value 1  Value 2
          /         \
    Feature 2   Feature 3
      /   \       /   \
Value 3 Value 4 Value 5 Value 6
  |       |       |       |
Leaf    Leaf   Leaf    Leaf
(Class 1) (Class 2) (Class 3) (Class 4)
```

Fig 3.4.1

In this diagram, the decision tree is split based on the values of Feature 1, Feature 2, and Feature 3, and the leaf nodes represent the predicted classes (or values) based on the decision rules learned from the data.

3.4.1 DECISION TREE TERMINOLOGIES

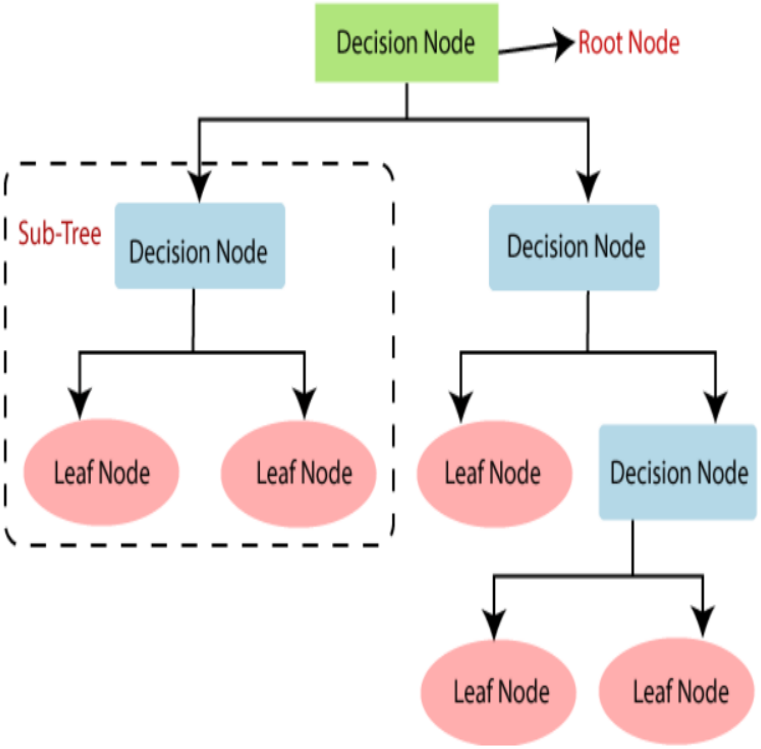


Fig 3.4.1.1

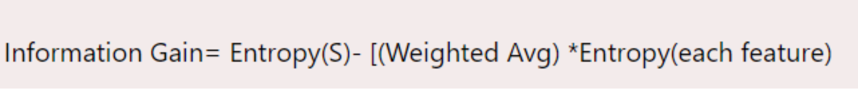
- Root Node: The topmost node of the decision tree, which represents the initial split based on the selected feature with the highest information gain, Gini impurity, or entropy.
- Split: The division of data into subsets based on the value of a selected feature. Each split creates a new branch in the decision tree.
- Node: Each split or decision point in the tree, represented by a rectangle or ellipse in a tree diagram.
- Leaf/Terminal Node: The end point of a branch in the decision tree where no further splitting occurs. Leaf nodes represent the final predicted class (for classification tasks) or predicted value (for regression tasks).
- Branch/Edge: The path connecting nodes in the decision tree, representing the decision rules learned from the data.

- Parent Node: The node that is split into child nodes during the construction of the decision tree.
- Child Node: The nodes resulting from the split of a parent node, representing the subsets of data based on the selected feature value.
- Depth: The number of levels or splits in the decision tree, starting from the root node. A deeper tree may be more complex and prone to overfitting.

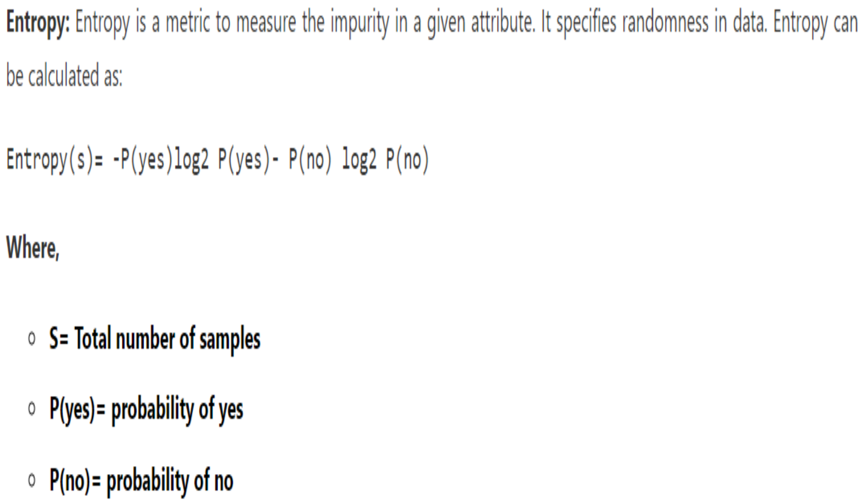
3.4.2 ATTRIBUTE SELECTION MEASURE

Attribute selection measures, also known as splitting criteria or splitting rules, are used in decision tree algorithms to determine the best feature to split the data at each node of the tree. The goal is to select the feature that maximizes the information gain or minimizes the impurity of the resulting subsets, leading to a more accurate and efficient decision tree. There are several common attribute selection measures used in decision tree algorithms, including:

- Information Gain: Information gain is a measure of the reduction in entropy or the amount of information gained by splitting the data on a particular feature. It calculates the difference between the entropy of the parent node and the weighted average of the entropies of the child nodes after the split. Information gain is widely used in decision tree algorithms, such as ID3, C4.5, and C5.0.



Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)



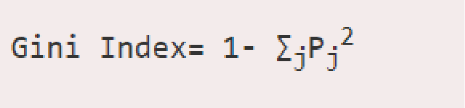
Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

- Gini Index: Gini index is a measure of impurity or the probability of misclassification of a randomly selected sample in a node. It works out the amount of the squared probabilities of each class in the parent hub and deducts the amount of the squared probabilities of each class in the youngster hubs after the split. The Gini index is commonly used in decision tree algorithms, such as CART (Classification and Regression Trees).



Gini Index= 1- $\sum_j P_j^2$

- Gain Ratio: Gain ratio is a measure of information gain normalized by the intrinsic information of the feature, which takes into account the number of possible splits or branches for a feature. It is determined as the proportion of data gain to part data, where parted data is the data expected to divide the information in light of a specific component.
- Chi-Square: Chi-square is a statistical measure used to test the independence of two categorical variables. In decision trees, chi-square can be utilized as a parting basis to quantify the relationship between an element and the objective variable. It calculates the chi-square statistic for each feature and selects the one with the highest value as the splitting feature.
- Regression Measures: For decision tree algorithms used in regression tasks, such as CART for regression, the attribute selection measures can be based

on regression-specific criteria such as mean squared error (MSE), mean absolute error (MAE), or variance reduction.

3.4.3 ADVANTAGES OF DECISION TREE

a. Interpretable and easy to understand: Decision trees generate simple and interpretable rules that can be easily visualized and understood by humans. The decision tree structure with nodes and branches can be visualized as a flowchart, making it easy to interpret and explain the decision-making process.

b. Handle both categorical and numerical data: Decision trees can deal with both all-out and mathematical information, making them flexible for a large number of information types and issue spaces. Decision tree algorithms can automatically handle feature selection, feature splitting, and feature interaction detection based on the data type and properties.

3.4.4 DISADVANTAGES OF DECISION TREE

- Overfitting: Decision trees are prone to overfitting, especially when the tree becomes deep and complex. Overfitting occurs when the tree captures noise or small patterns in the training data, leading to poor generalization performance on unseen data. This can be moderated by utilizing strategies like pruning, setting a most extreme profundity for the tree, or utilizing troupe techniques like irregular woods.
- Lack of robustness: Decision trees are sensitive to small changes in the training data, which can lead to different tree structures and predictions. This lack of robustness makes them vulnerable to data with small variations or noisy data, and they may not generalize well to unseen data.

3.5 RANDOM FOREST CLASSIFIER

A random forest classifier is a type of ensemble machine learning algorithm used for classification tasks. It combines multiple decision trees to create a more accurate and robust classifier. Each decision tree in the random forest is trained on a subset of the data, randomly sampled with replacement from the original dataset, and a random subset of features is considered for splitting at each node. This randomness helps to reduce overfitting and improves the model's ability to generalize to new data.

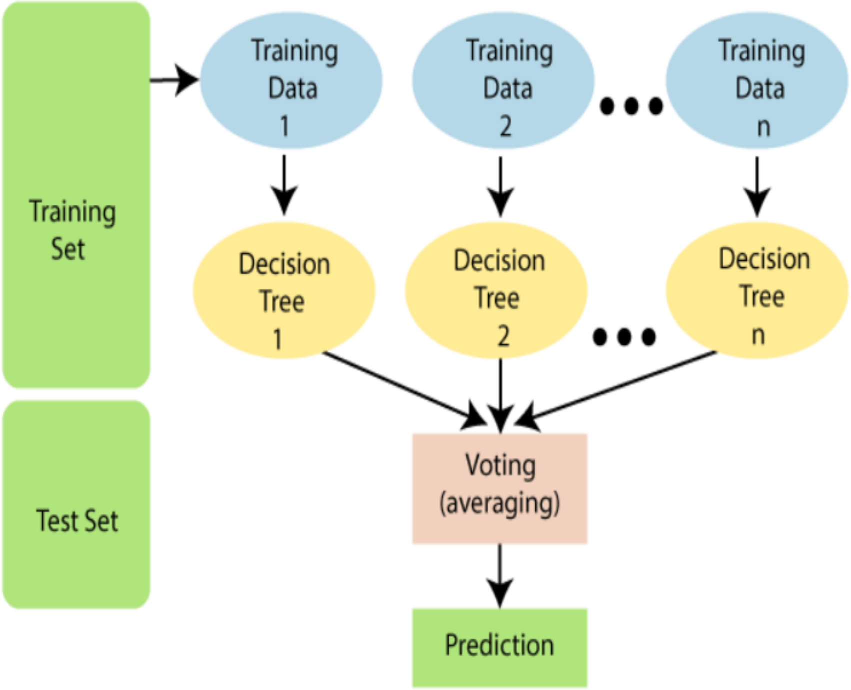


Fig 3.5.1

3.5.1 ASSUMPTIONS OF RANDOM FOREST ALGORITHM

- Independence of trees: The assumption of independence among the individual decision trees in the random forest. Each tree is trained on a random subset of data and features, and the predictions are combined through majority voting. This assumes that the trees are not overly correlated, meaning that they make decisions independently and do not rely too heavily on each other.

- Homogeneous features: Random forests assume that the features used for training are homogeneous in their importance and distribution. If some features have significantly different scales, units, or distributions, they may disproportionately influence the decisions made by the trees. Therefore, it's important to preprocess the data and normalize or scale the features appropriately before training a random forest classifier.

- No missing values: Random forests generally do not handle missing values well. If the dataset contains missing values, it's important to preprocess the data by imputing or handling missing values appropriately before training a random forest classifier. Missing values can negatively impact the accuracy and performance of the model if not properly addressed.

3.5.2 ADVANTAGES

- Robustness to overfitting: Random forests are less prone to overfitting compared to individual decision trees. The randomization in selecting subsets of data and features for each tree, as well as the majority voting for predictions, helps to reduce the risk of overfitting and improve generalization performance, making random forests more robust and accurate on unseen data.

- High accuracy: Random forests are known for their high accuracy and predictive performance. They can capture complex nonlinear relationships between features and target classes, and can handle both categorical and continuous features effectively. Random forests are particularly effective in handling high-dimensional data with a large number of features, making them suitable for a wide range of applications.

3.5.3 DISADVANTAGES

- Lack of interpretability: Random forests are considered to be "black box" models, meaning that their internal decision-making process is not easily interpretable or explainable. It tends to be trying to comprehend how individual trees in the timberland make expectations, which might restrict the capacity to acquire experiences into the hidden examples in the information or clear up the model's forecasts for partners.

- Computational complexity: Random forests can be computationally expensive, especially when dealing with large datasets or a large number of trees in the forest. Preparing different choice trees and consolidating their expectations can require critical computational assets and time, which might restrict their versatility for extremely enormous datasets or ongoing applications.

CHAPTER 4

METHODOLOGY AND PACKAGES

4.1 AGILE METHODOLOGY

Agile adheres to four principles, according to the Agile Philosophy Individuals and Communications above procedures and technological means.Operational and working software over comprehensive certification and documents. Close association with users and customers over contract diplomacy.Responding to the alterations over getting behind a fixed plan.

The Agile Methodology refocuses attention on consumers rather than papercertifications. Tasks and objectives are divided into 'client stories' in agile, where the working team strategizes for the tasks ahead and estimates how long the entire projectwill take to complete.

A story could be non-coding, like research, exploration, and design, or coding, such code rewriting and new feature and element development. These stories are then separated into sprints, each lasting two weeks. Throughout each sprint, the team collaborates extensively and decides on the tasks to be performed.

4.2 PACKAGES USED

A python bundle is an assortment of modules. Modules that are connected with one another are primarily placed in a similar bundle. At the point when a module from an outside bundle is expected in a program, that bundle can be imported and its modules can be put to utilize.

4.2.1 NumPy

NumPy, or Numerical Python, is a package that consists of multi-faceted cluster objects and a variety of routines for dealing with them. NumPy can be used to execute numerical and consistent clustering procedures.

NumPy is a Python distribution. It's short for 'Mathematical Python.' It's a library with a variety of multi-faceted cluster objects and timetables for managing exhibits. Utilizing NumPy, an engineer can play out the accompanying activities − Tasks connected with straight polynomial math. NumPy is widely used in conjunction with other Python packages such as SciPy (Scientific Python) and Matplotlib (plotting library). This combination is frequently used as a replacement for MatLab, a well- known platform for specialised registers.

4.2.2 PANDAS

Pandas is an open-source library designed primarily for using social or branded information in a quick and efficient manner. It provides many information designs and activities for managing time series and mathematical data. The NumPy library serves as the foundation for this library. Pandas is swift and provides clients with exceptional execution and efficiency. A one-layered marked exhibit called the Pandas Series is suitable for storing data of any kind, including numbers, strings, floats, Python objects, and so forth. The pivot names are typically referred to as files. Only a portion of a success sheet contains the Pandas Series. Marks don't have to be unique files.

4.2.3 Matplotlib.pyplot

Matplotlib.pyplot is a Python library that gives an assortment of capabilities to making perceptions, like plots, outlines, and charts. It is a well-known and broadly involved library for information perception in the field of information science, AI, and information examination.

Key features are:

- Plotting functions: matplotlib.pyplot provides various functions for creating different types of plots, such as line plots, scatter plots, bar plots, histograms, and more. These capabilities permit you to redo the presence of the plots, like setting names, titles, varieties, and markers.

- Wide range of plot types: matplotlib.pyplot supports a wide range of plot types, including line plots, scatter plots, bar plots, histograms, 3D plots, contour plots, and more.

4.2.4 SkLearn

Sklearn is the usually involved shortening for Scikit-learn, which is a famous open-source AI library in Python. It gives a great many devices and capabilities for different AI undertakings, like characterization, relapse, bunching, dimensionality decrease, and model assessment, and that's just the beginning.

Key features are:

- Preprocessing and feature extraction: sklearn offers a rich set of tools for data preprocessing and feature extraction, including data scaling, normalization, encoding categorical variables, handling missing values, feature selection, and more. These preprocessing techniques are crucial for preparing the data before feeding it to machine learning algorithms.
- Model evaluation and selection: sklearn provides functions for evaluating the performance of machine learning models, such as cross-validation, model evaluation metrics, model selection using techniques like grid search and randomized search, and more. These devices help in evaluating the exhibition of various models and choosing the best model for a given undertaking.

4.2.5 SEABORN

Seaborn is a Python information representation library in light of Matplotlib that gives an undeniable level connection point to making alluring and educational factual illustrations. Seaborn is based on top of matplotlib and gives extra usefulness to making outwardly engaging and enlightening plots for information examination and representation.

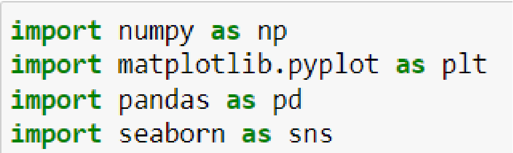
Seaborn improves on the most common way of making representations by giving a large number of inherent subjects and variety ranges, as well as capabilities for picturing complex factual connections, for example, dissipate plots, line plots, bar plots, box plots, violin plots, and some more. It likewise offers help for picturing clear-cut information, time series information, and multi-variable connections in a compact and proficient way.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 PACKAGES INSTALLED

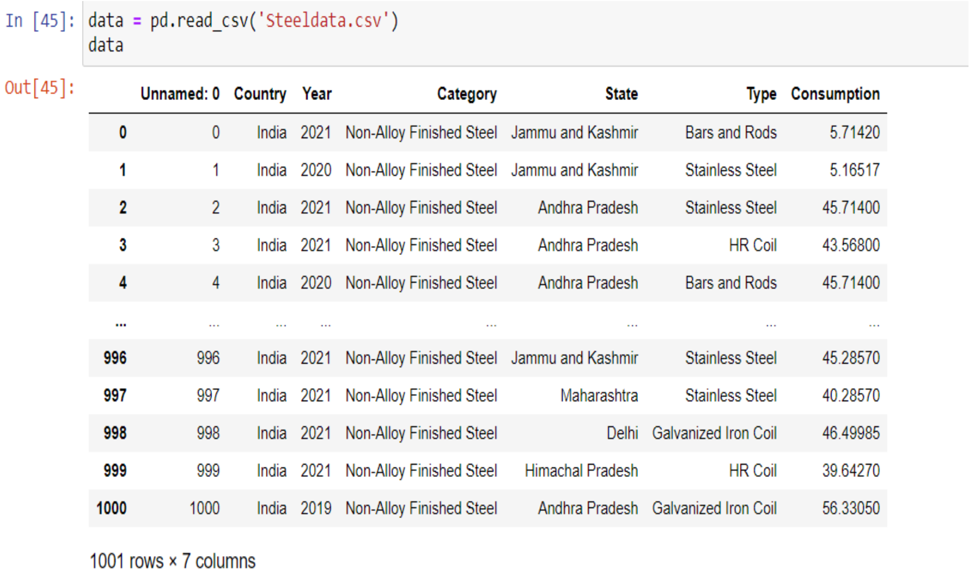
For all the three modules the same set of packages was installed and used i.e., NumPy, Pandas, Matplotlib.py, SkLearns except in case of iron ore quality in which we use seaborn to represent the relation between % iron Feed and % silica Feed.



```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

Fig. 5.1.1

5.2 Data is loaded



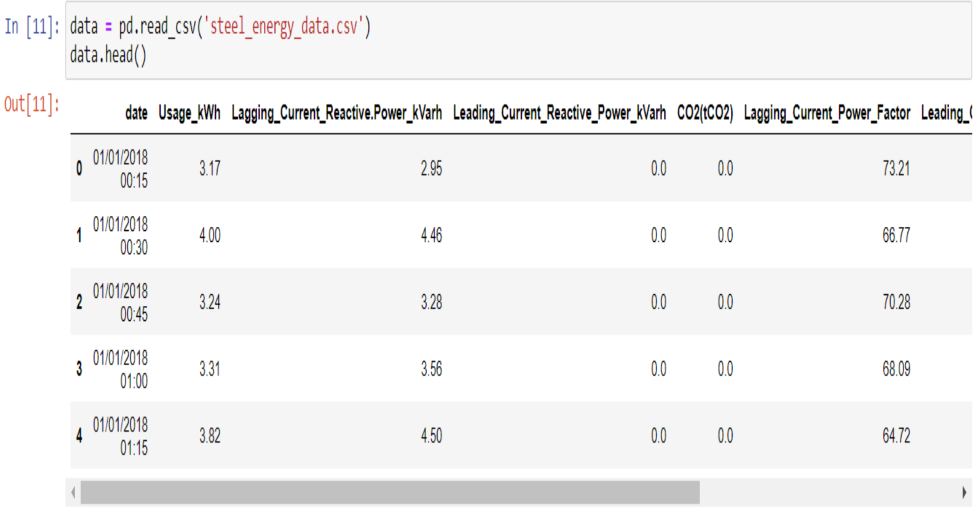
```
In [45]: data = pd.read_csv('Steeldata.csv')
         data
```

Out[45]:

	Unnamed: 0	Country	Year	Category	State	Type	Consumption
0	0	India	2021	Non-Alloy Finished Steel	Jammu and Kashmir	Bars and Rods	5.71420
1	1	India	2020	Non-Alloy Finished Steel	Jammu and Kashmir	Stainless Steel	5.16517
2	2	India	2021	Non-Alloy Finished Steel	Andhra Pradesh	Stainless Steel	45.71400
3	3	India	2021	Non-Alloy Finished Steel	Andhra Pradesh	HR Coil	43.56800
4	4	India	2020	Non-Alloy Finished Steel	Andhra Pradesh	Bars and Rods	45.71400
...	...	...	...	...	...	...	...
996	996	India	2021	Non-Alloy Finished Steel	Jammu and Kashmir	Stainless Steel	45.28570
997	997	India	2021	Non-Alloy Finished Steel	Maharashtra	Stainless Steel	40.28570
998	998	India	2021	Non-Alloy Finished Steel	Delhi	Galvanized Iron Coil	46.49985
999	999	India	2021	Non-Alloy Finished Steel	Himachal Pradesh	HR Coil	39.64270
1000	1000	India	2019	Non-Alloy Finished Steel	Andhra Pradesh	Galvanized Iron Coil	56.33050

1001 rows × 7 columns

Fig 5.2.1

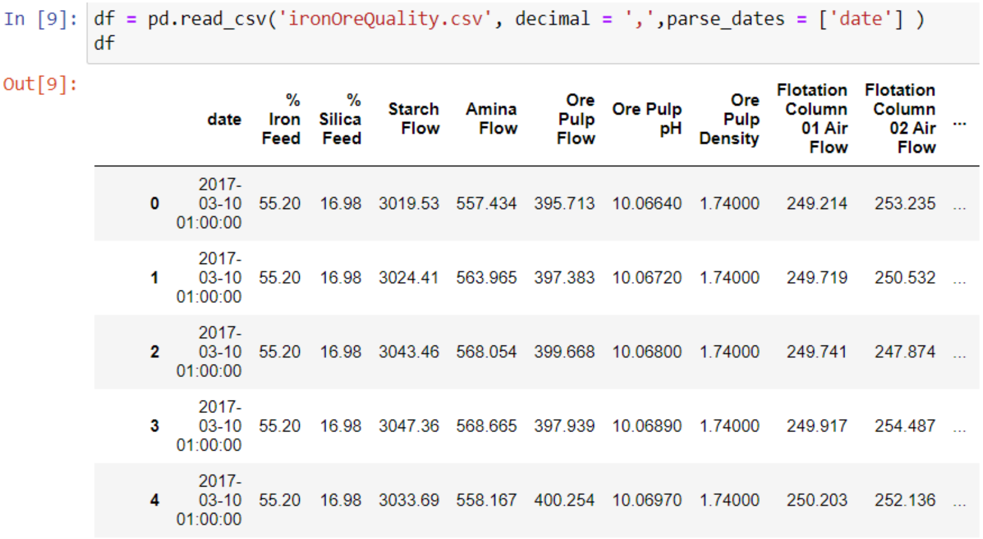


```
In [11]: data = pd.read_csv('steel_energy_data.csv')
         data.head()
```

Out[11]:

	date	Usage_kWh	Lagging_Current_Reactive.Power_kVarh	Leading_Current_Reactive_Power_kVarh	CO2(tCO2)	Lagging_Current_Power_Factor	Leading_(
0	01/01/2018 00:15	3.17	2.95	0.0	0.0	73.21	
1	01/01/2018 00:30	4.00	4.46	0.0	0.0	66.77	
2	01/01/2018 00:45	3.24	3.28	0.0	0.0	70.28	
3	01/01/2018 01:00	3.31	3.56	0.0	0.0	68.09	
4	01/01/2018 01:15	3.82	4.50	0.0	0.0	64.72	

Fig 5.2.2



```
In [9]: df = pd.read_csv('ironOreQuality.csv', decimal = ',',parse_dates = ['date'] )
        df
```

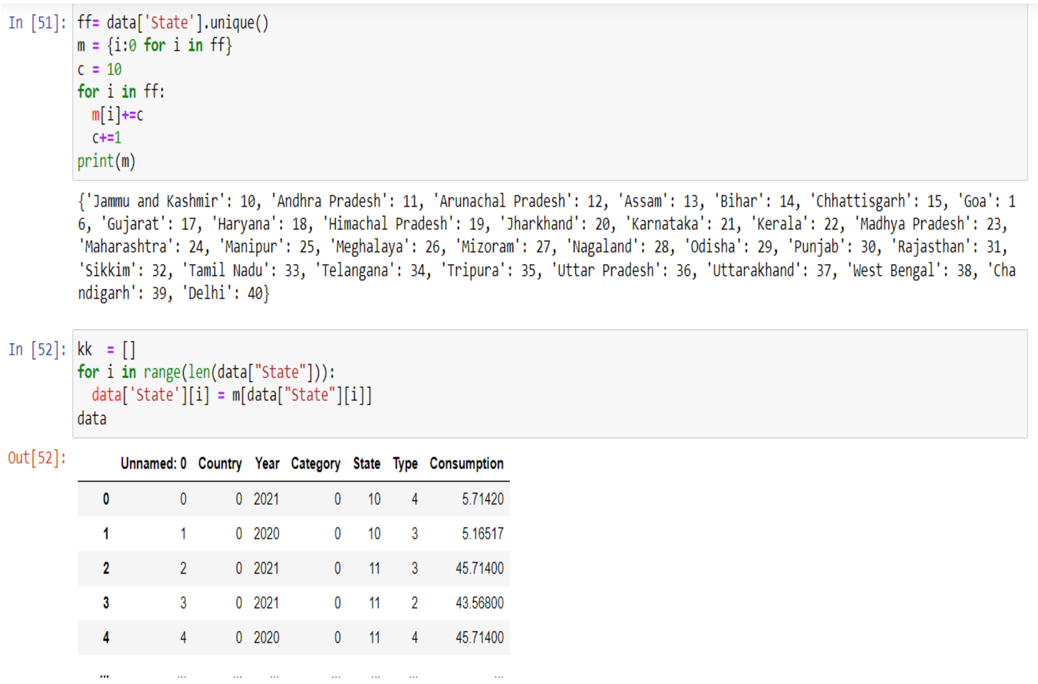
Out[9]:

	date	% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow	Flotation Column 02 Air Flow	...
0	2017-03-10 01:00:00	55.20	16.98	3019.53	557.434	395.713	10.06640	1.74000	249.214	253.235	...
1	2017-03-10 01:00:00	55.20	16.98	3024.41	563.965	397.383	10.06720	1.74000	249.719	250.532	...
2	2017-03-10 01:00:00	55.20	16.98	3043.46	568.054	399.668	10.06800	1.74000	249.741	247.874	...
3	2017-03-10 01:00:00	55.20	16.98	3047.36	568.665	397.939	10.06890	1.74000	249.917	254.487	...
4	2017-03-10 01:00:00	55.20	16.98	3033.69	558.167	400.254	10.06970	1.74000	250.203	252.136	...

Fig 5.2.3

5.3 Data preprocessing

For applying KNN and multiple regresssion, I had to change the “string data” to “integer data”. For doing so, first I changed the States by mapping them to number starting from 10.

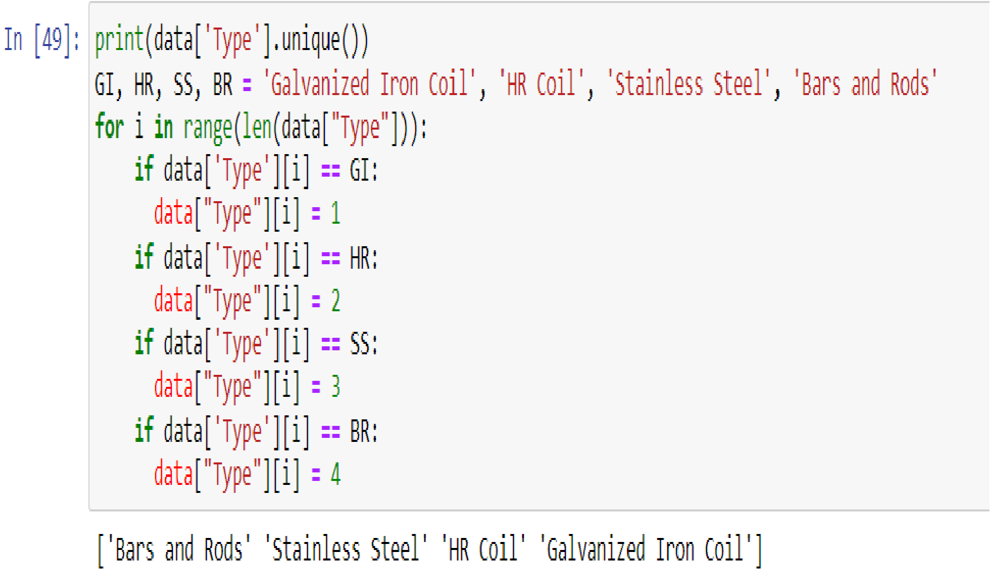


```
In [51]: ff= data['State'].unique()
         m = {i:0 for i in ff}
         c = 10
         for i in ff:
           m[i]+=c
           c+=1
         print(m)

         {'Jammu and Kashmir': 10, 'Andhra Pradesh': 11, 'Arunachal Pradesh': 12, 'Assam': 13, 'Bihar': 14, 'Chhattisgarh': 15, 'Goa': 1
         6, 'Gujarat': 17, 'Haryana': 18, 'Himachal Pradesh': 19, 'Jharkhand': 20, 'Karnataka': 21, 'Kerala': 22, 'Madhya Pradesh': 23,
         'Maharashtra': 24, 'Manipur': 25, 'Meghalaya': 26, 'Mizoram': 27, 'Nagaland': 28, 'Odisha': 29, 'Punjab': 30, 'Rajasthan': 31,
         'Sikkim': 32, 'Tamil Nadu': 33, 'Telangana': 34, 'Tripura': 35, 'Uttar Pradesh': 36, 'Uttarakhand': 37, 'West Bengal': 38, 'Cha
         ndigarh': 39, 'Delhi': 40}

In [52]: kk  = []
         for i in range(len(data["State"])):
           data['State'][i] = m[data["State"][i]]
         data

Out[52]:
```

	Unnamed: 0	Country	Year	Category	State	Type	Consumption
0	0	0	2021	0	10	4	5.71420
1	1	0	2020	0	10	3	5.16517
2	2	0	2021	0	11	3	45.71400
3	3	0	2021	0	11	2	43.56800
4	4	0	2020	0	11	4	45.71400
...	...	...	...	...	...	...	...

Fig 5.3.1

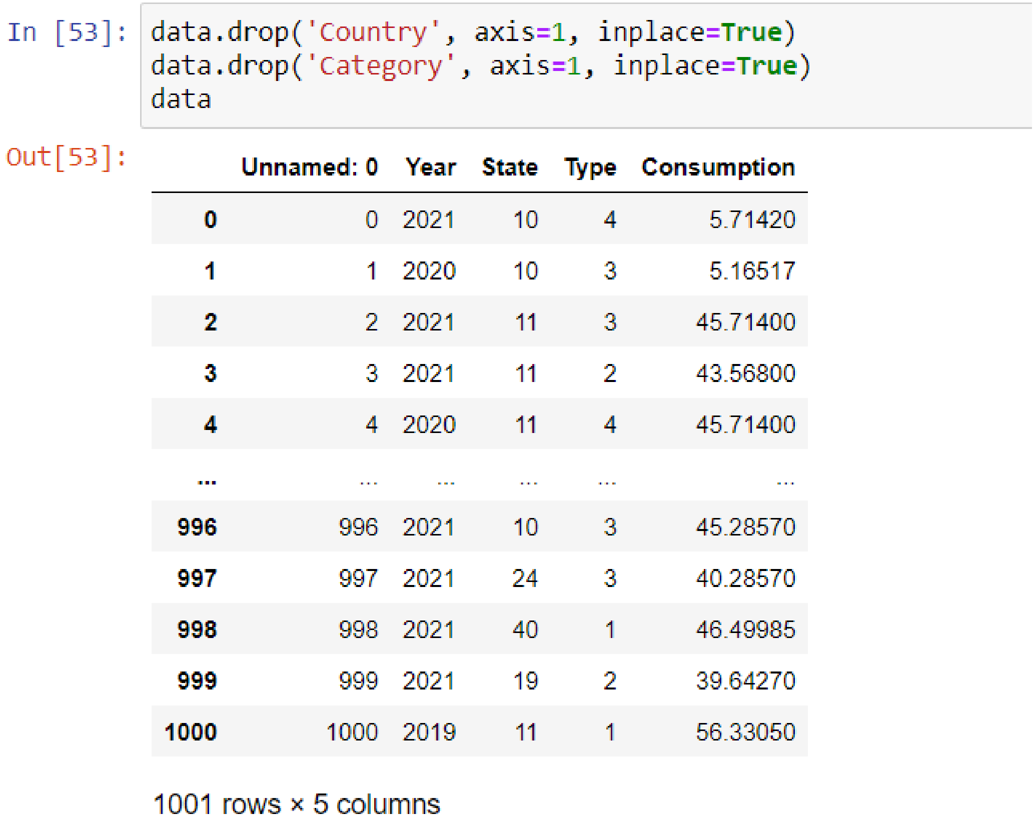
Then I changed the “Types of Steel products” to integer data by mapping them to number starting from 1.



```
In [49]: print(data['Type'].unique())
         GI, HR, SS, BR = 'Galvanized Iron Coil', 'HR Coil', 'Stainless Steel', 'Bars and Rods'
         for i in range(len(data["Type"])):
             if data['Type'][i] == GI:
               data["Type"][i] = 1
             if data['Type'][i] == HR:
               data["Type"][i] = 2
             if data['Type'][i] == SS:
               data["Type"][i] = 3
             if data['Type'][i] == BR:
               data["Type"][i] = 4

         ['Bars and Rods' 'Stainless Steel' 'HR Coil' 'Galvanized Iron Coil']
```

Fig 5.3.2

I also deleted the unnecessary columns which was not used in my analysis.



```
In [53]: data.drop('Country', axis=1, inplace=True)
         data.drop('Category', axis=1, inplace=True)
         data
```

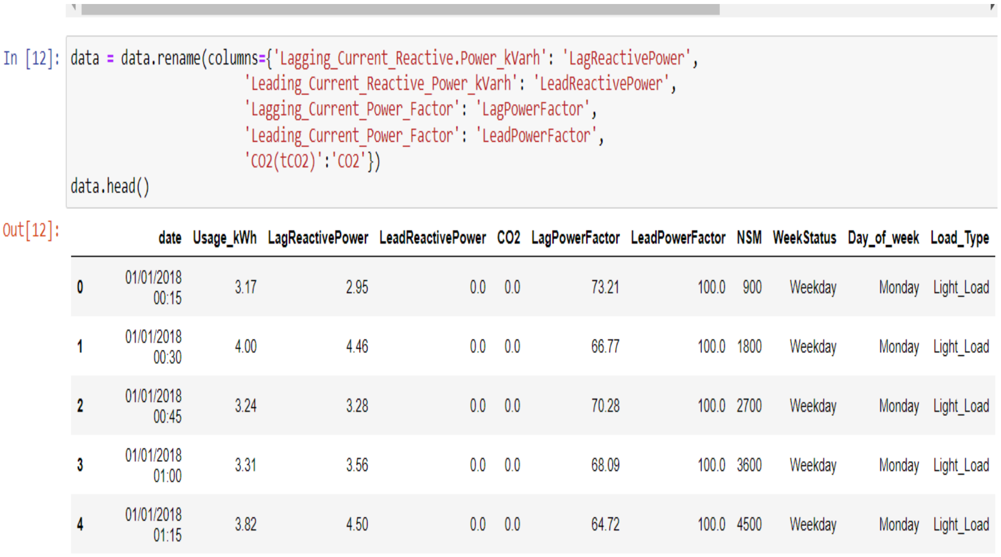
Out[53]:

	Unnamed: 0	Year	State	Type	Consumption
0	0	2021	10	4	5.71420
1	1	2020	10	3	5.16517
2	2	2021	11	3	45.71400
3	3	2021	11	2	43.56800
4	4	2020	11	4	45.71400
...	...	...	...	...	...
996	996	2021	10	3	45.28570
997	997	2021	24	3	40.28570
998	998	2021	40	1	46.49985
999	999	2021	19	2	39.64270
1000	1000	2019	11	1	56.33050

1001 rows × 5 columns

Fig 5.3.3

Now for applying Decision Tree classifier I had a different dataset, for which I renamed a few columns:



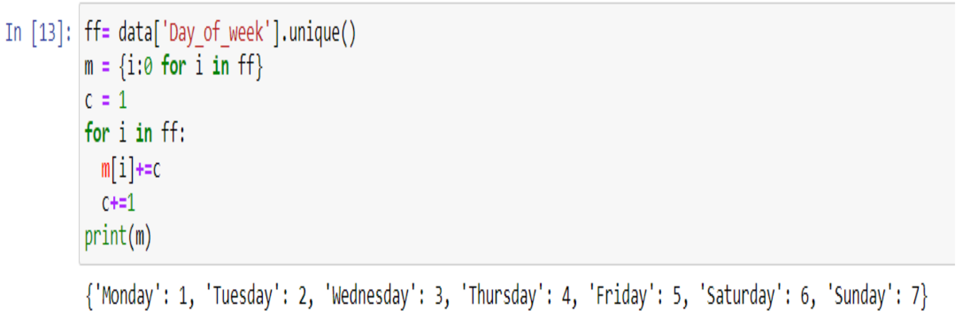
```
In [12]: data = data.rename(columns={'Lagging_Current_Reactive.Power_kVarh': 'LagReactivePower',
                         'Leading_Current_Reactive_Power_kVarh': 'LeadReactivePower',
                         'Lagging_Current_Power_Factor': 'LagPowerFactor',
                         'Leading_Current_Power_Factor': 'LeadPowerFactor',
                         'CO2(tCO2)':'CO2'})
         data.head()
```

Out[12]:

	date	Usage_kWh	LagReactivePower	LeadReactivePower	CO2	LagPowerFactor	LeadPowerFactor	NSM	WeekStatus	Day_of_week	Load_Type
0	01/01/2018 00:15	3.17	2.95	0.0	0.0	73.21	100.0	900	Weekday	Monday	Light_Load
1	01/01/2018 00:30	4.00	4.46	0.0	0.0	66.77	100.0	1800	Weekday	Monday	Light_Load
2	01/01/2018 00:45	3.24	3.28	0.0	0.0	70.28	100.0	2700	Weekday	Monday	Light_Load
3	01/01/2018 01:00	3.31	3.56	0.0	0.0	68.09	100.0	3600	Weekday	Monday	Light_Load
4	01/01/2018 01:15	3.82	4.50	0.0	0.0	64.72	100.0	4500	Weekday	Monday	Light_Load

Fig 5.3.4

I also mapped the “Day_of_week” to integer datatype starting from 1.

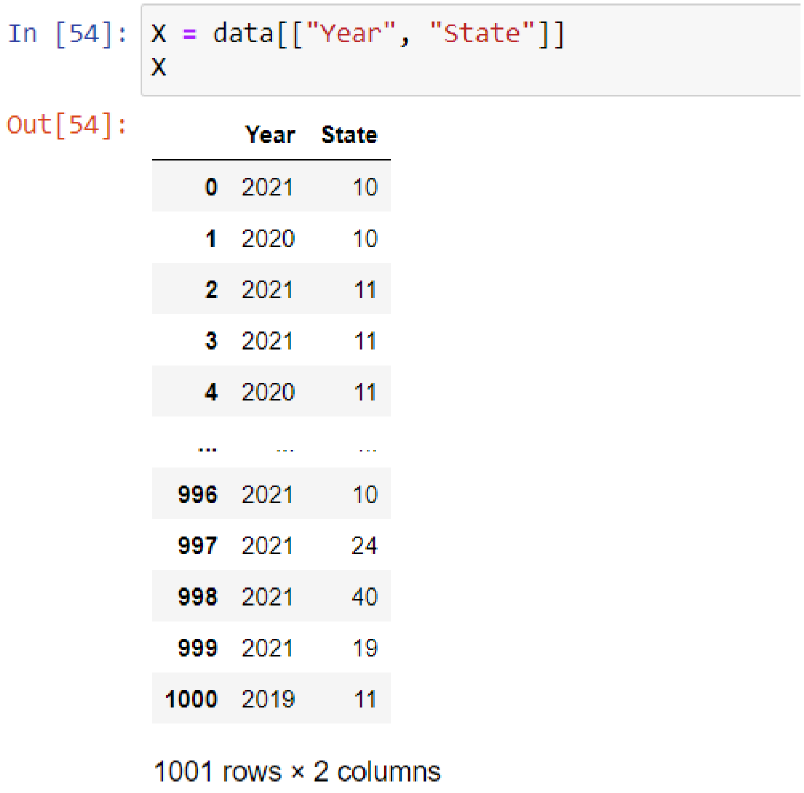


```
In [13]: ff= data['Day_of_week'].unique()
         m = {i:0 for i in ff}
         c = 1
         for i in ff:
           m[i]+=c
           c+=1
         print(m)

         {'Monday': 1, 'Tuesday': 2, 'Wednesday': 3, 'Thursday': 4, 'Friday': 5, 'Saturday': 6, 'Sunday': 7}
```

Fig 5.3.5

5.4 Apply Multiple Linear Regression

For applying Multiple Linear Regression, I defined variables X & Y:



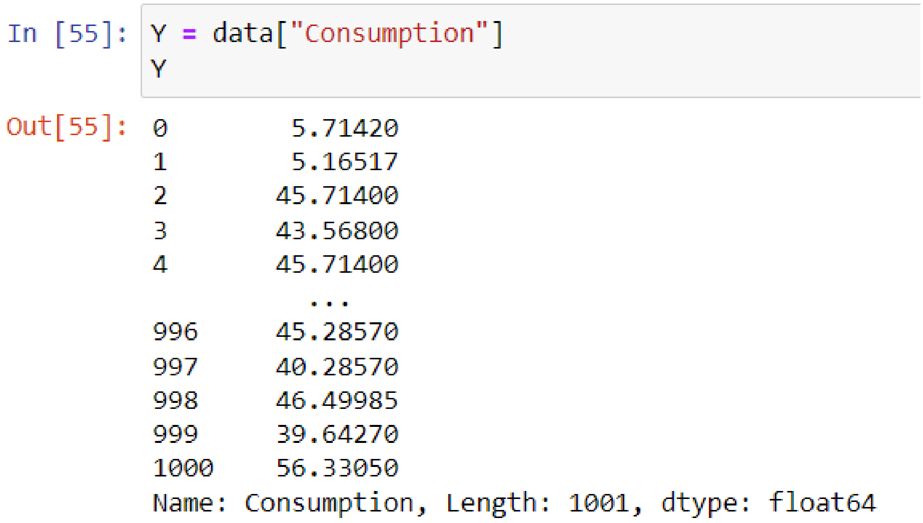
```
In [54]: X = data[["Year", "State"]]
         X
```

Out[54]:

	Year	State
0	2021	10
1	2020	10
2	2021	11
3	2021	11
4	2020	11
...	...	...
996	2021	10
997	2021	24
998	2021	40
999	2021	19
1000	2019	11

1001 rows × 2 columns

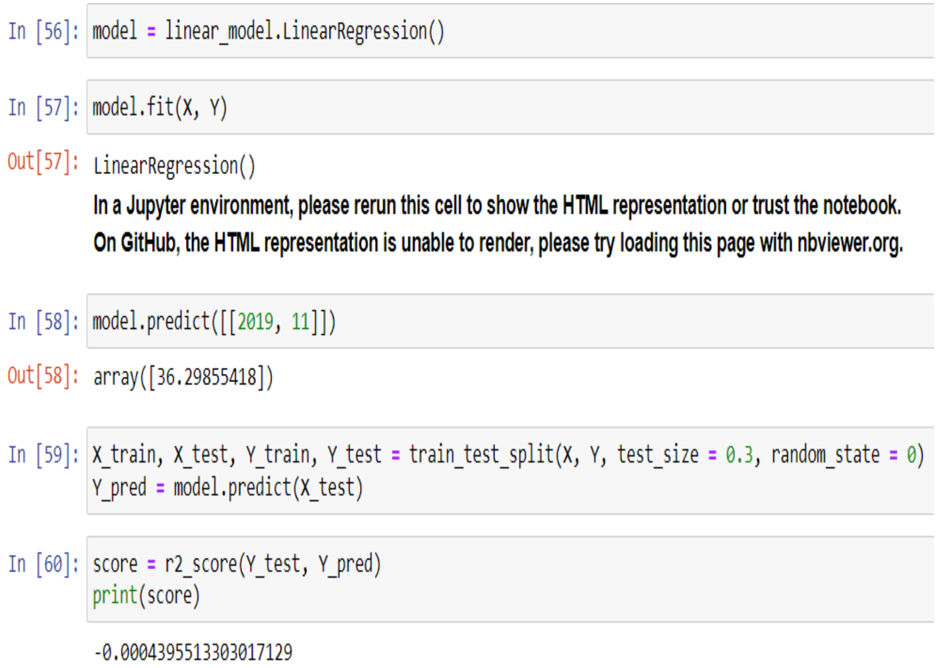
Fig 5.4.1



```
In [55]: Y = data["Consumption"]
         Y

Out[55]: 0        5.71420
          1        5.16517
          2       45.71400
          3       43.56800
          4       45.71400
                    ...
          996     45.28570
          997     40.28570
          998     46.49985
          999     39.64270
          1000    56.33050
          Name: Consumption, Length: 1001, dtype: float64
```

Fig 5.4.2

After defining my X & Y, I finally applied the Multiple Linear Regression.



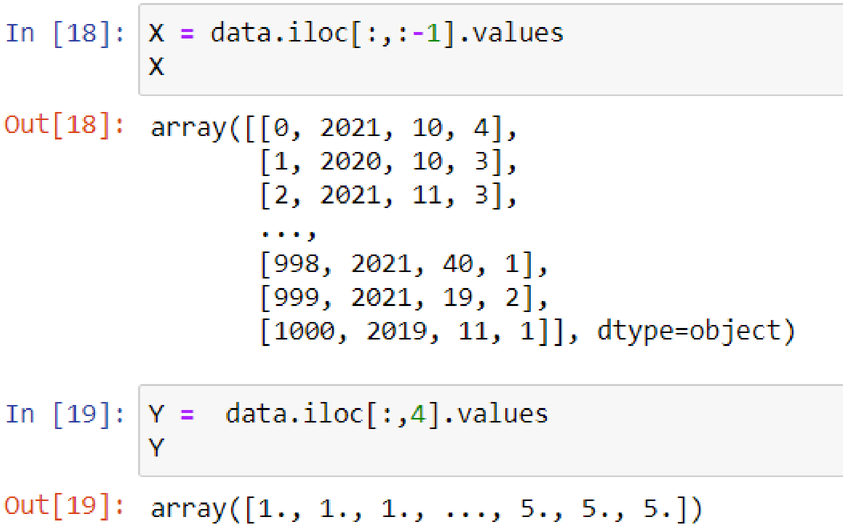
```
In [56]: model = linear_model.LinearRegression()

In [57]: model.fit(X, Y)

Out[57]: LinearRegression()
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [58]: model.predict([[2019, 11]])

Out[58]: array([36.29855418])

In [59]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3, random_state = 0)
         Y_pred = model.predict(X_test)

In [60]: score = r2_score(Y_test, Y_pred)
         print(score)

         -0.0004395513303017129
```

Fig 5.4.3

I got a score ~ 0.00043 which is very good as the score should be between 0 and 1. I also predicted a value just to check if my algorithm works fine or not.

5.5 Applying KNN

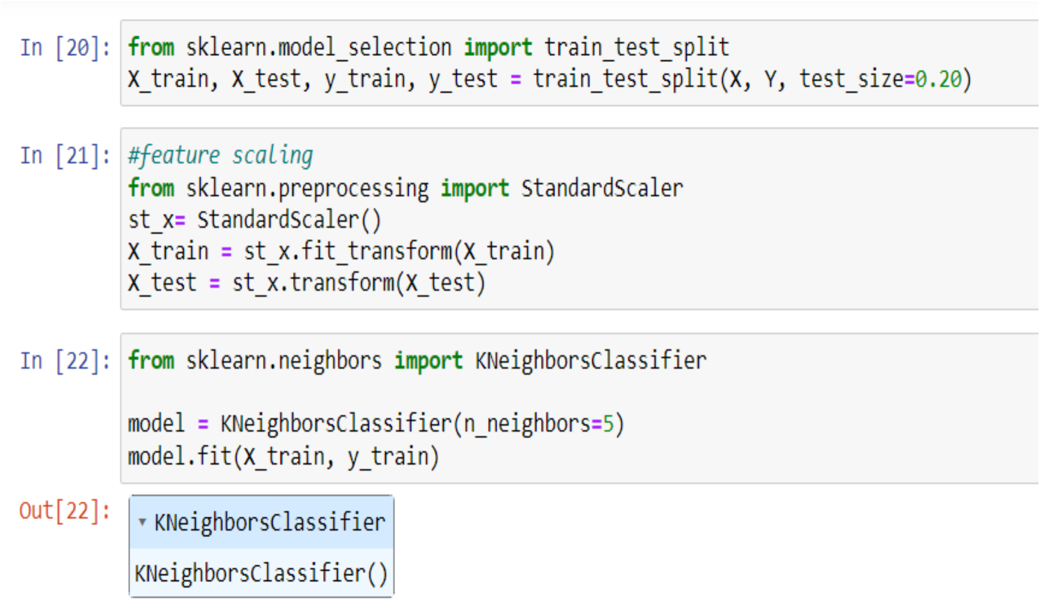
For applying KNN, I defined variables X & Y:



```
In [18]: X = data.iloc[:,:-1].values
         X

Out[18]: array([[0, 2021, 10, 4],
                [1, 2020, 10, 3],
                [2, 2021, 11, 3],
                ...,
                [998, 2021, 40, 1],
                [999, 2021, 19, 2],
                [1000, 2019, 11, 1]], dtype=object)

In [19]: Y =  data.iloc[:,4].values
         Y

Out[19]: array([1., 1., 1., ..., 5., 5., 5.])
```

Fig 5.5.1

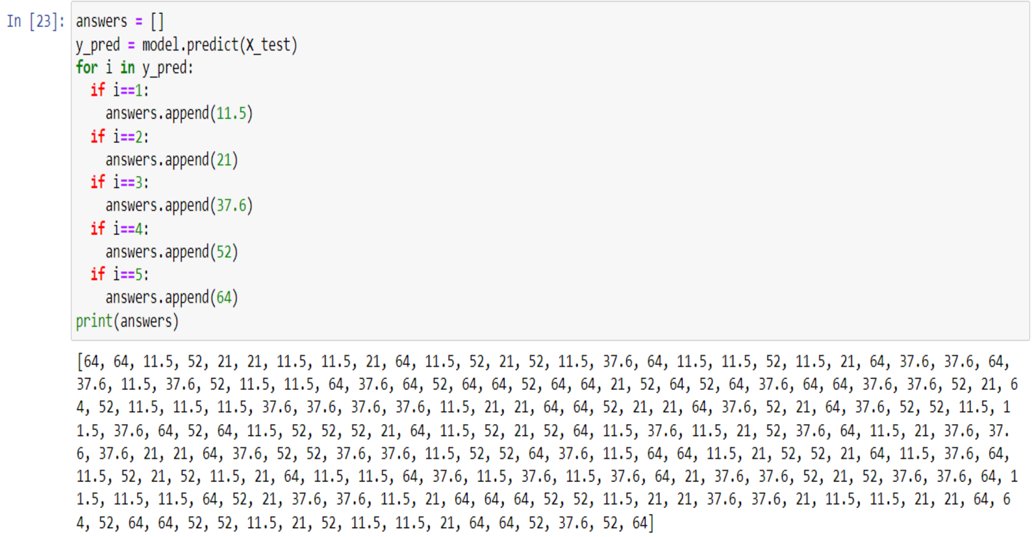
After this, I applied KNN.



```
In [20]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.20)

In [21]: #feature scaling
         from sklearn.preprocessing import StandardScaler
         st_x= StandardScaler()
         X_train = st_x.fit_transform(X_train)
         X_test = st_x.transform(X_test)

In [22]: from sklearn.neighbors import KNeighborsClassifier

         model = KNeighborsClassifier(n_neighbors=5)
         model.fit(X_train, y_train)

Out[22]: ▾ KNeighborsClassifier
         KNeighborsClassifier()
```

Fig 5.5.2

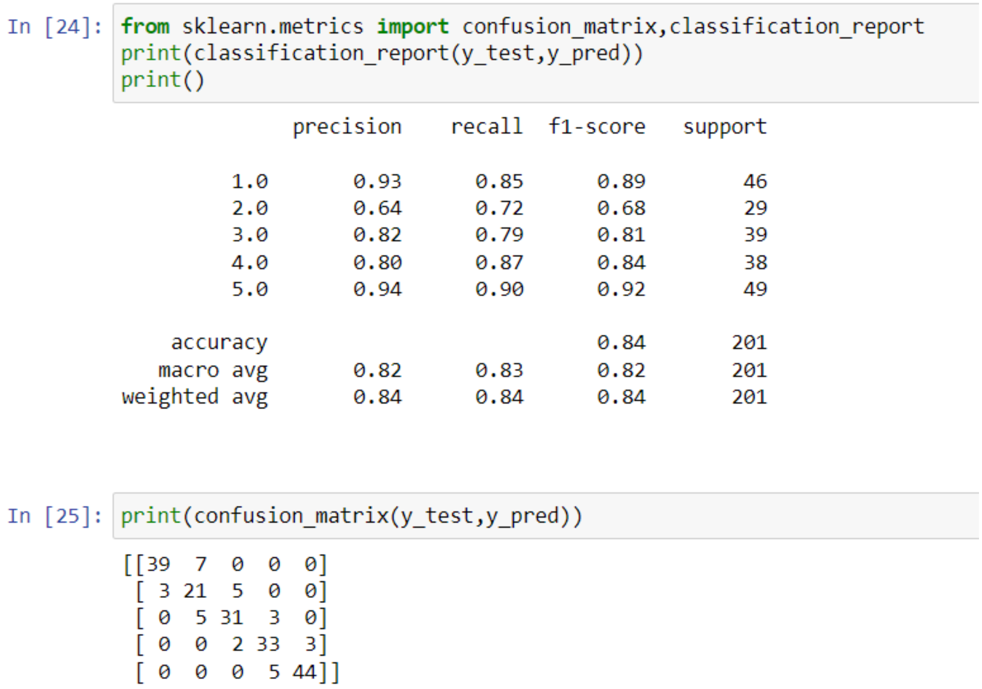
According to the KNN, my data was divided into 5 neighbours, so I tested for each neighbour.



```
In [23]: answers = []
         y_pred = model.predict(X_test)
         for i in y_pred:
           if i==1:
             answers.append(11.5)
           if i==2:
             answers.append(21)
           if i==3:
             answers.append(37.6)
           if i==4:
             answers.append(52)
           if i==5:
             answers.append(64)
         print(answers)

         [64, 64, 11.5, 52, 21, 21, 11.5, 11.5, 21, 64, 11.5, 52, 21, 52, 11.5, 37.6, 64, 11.5, 11.5, 52, 11.5, 21, 64, 37.6, 37.6, 64,
         37.6, 11.5, 37.6, 52, 11.5, 11.5, 64, 37.6, 64, 52, 64, 64, 52, 64, 64, 21, 52, 64, 52, 64, 37.6, 64, 64, 37.6, 37.6, 52, 21, 6
         4, 52, 11.5, 11.5, 11.5, 37.6, 37.6, 37.6, 37.6, 11.5, 21, 21, 64, 64, 52, 21, 21, 64, 37.6, 52, 21, 64, 37.6, 52, 52, 11.5, 1
         1.5, 37.6, 64, 52, 64, 11.5, 52, 52, 52, 21, 64, 11.5, 52, 21, 52, 64, 11.5, 37.6, 11.5, 21, 52, 37.6, 64, 11.5, 21, 37.6, 37.
         6, 37.6, 21, 21, 64, 37.6, 52, 52, 37.6, 37.6, 11.5, 52, 52, 64, 37.6, 11.5, 64, 64, 11.5, 21, 52, 52, 21, 64, 11.5, 37.6, 64,
         11.5, 52, 21, 52, 11.5, 21, 64, 11.5, 11.5, 64, 37.6, 11.5, 37.6, 11.5, 37.6, 64, 21, 37.6, 37.6, 52, 21, 52, 37.6, 37.6, 64, 1
         1.5, 11.5, 11.5, 64, 52, 21, 37.6, 37.6, 11.5, 21, 64, 64, 64, 52, 52, 11.5, 21, 21, 37.6, 37.6, 21, 11.5, 11.5, 21, 21, 64, 6
         4, 52, 64, 64, 52, 52, 11.5, 21, 52, 11.5, 11.5, 21, 64, 64, 52, 37.6, 52, 64]
```

Fig 5.5.3

Now I found out the score of this algorithm:



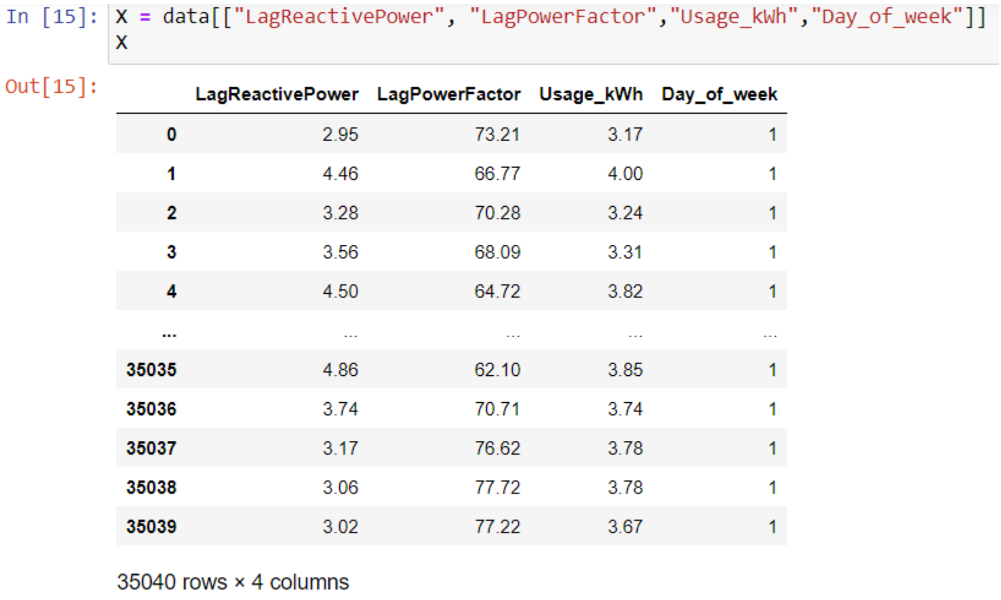
```
In [24]: from sklearn.metrics import confusion_matrix,classification_report
         print(classification_report(y_test,y_pred))
         print()

                       precision    recall  f1-score   support

                  1.0       0.93      0.85      0.89        46
                  2.0       0.64      0.72      0.68        29
                  3.0       0.82      0.79      0.81        39
                  4.0       0.80      0.87      0.84        38
                  5.0       0.94      0.90      0.92        49

             accuracy                           0.84       201
            macro avg       0.82      0.83      0.82       201
         weighted avg       0.84      0.84      0.84       201
```

```
In [25]: print(confusion_matrix(y_test,y_pred))

         [[39  7  0  0  0]
          [ 3 21  5  0  0]
          [ 0  5 31  3  0]
          [ 0  0  2 33  3]
          [ 0  0  0  5 44]]
```

Fig 5.5.4

5.6 Apply Decision Tree Algorithm

For applying Decision Tree algorithm, first I defined X & Y:



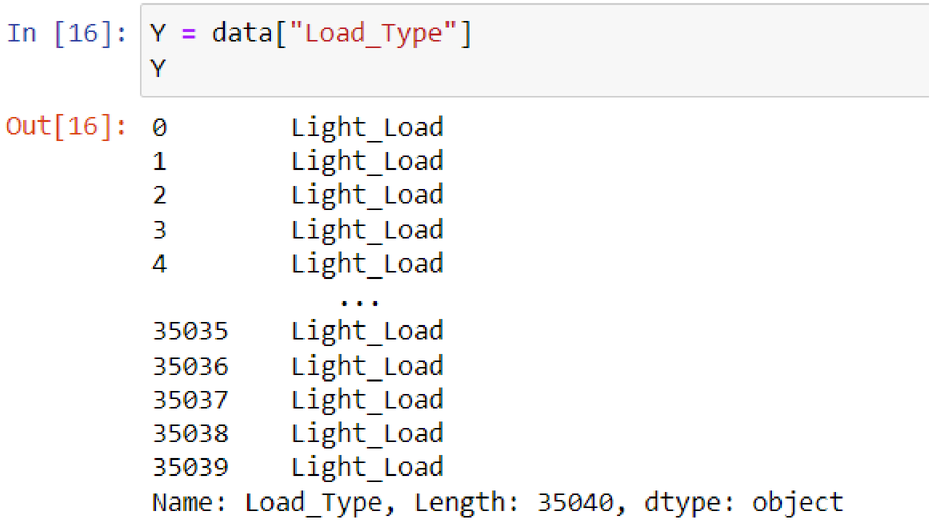
```
In [15]: X = data[["LagReactivePower", "LagPowerFactor","Usage_kWh","Day_of_week"]]
         X
```

Out[15]:

	LagReactivePower	LagPowerFactor	Usage_kWh	Day_of_week
0	2.95	73.21	3.17	1
1	4.46	66.77	4.00	1
2	3.28	70.28	3.24	1
3	3.56	68.09	3.31	1
4	4.50	64.72	3.82	1
...	...	...	...	...
35035	4.86	62.10	3.85	1
35036	3.74	70.71	3.74	1
35037	3.17	76.62	3.78	1
35038	3.06	77.72	3.78	1
35039	3.02	77.22	3.67	1

35040 rows × 4 columns

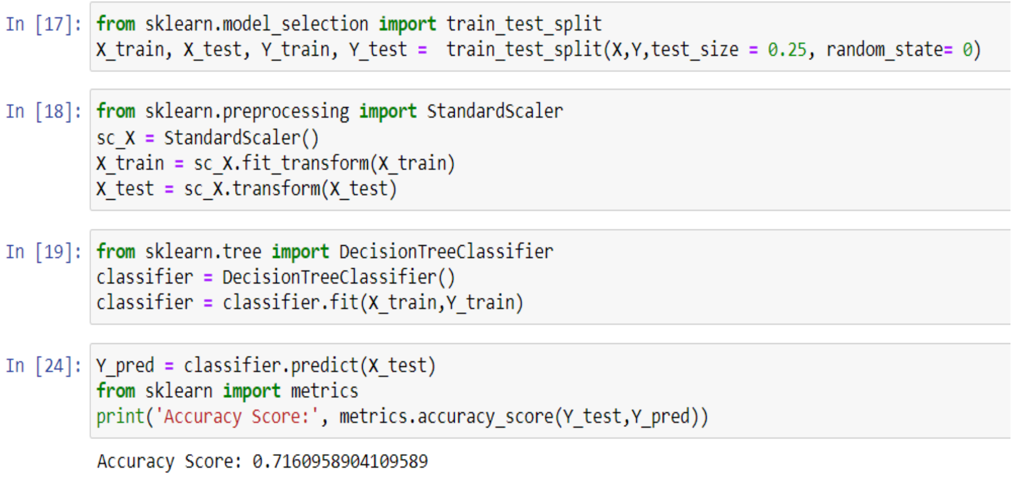
Fig 5.6.1



```
In [16]: Y = data["Load_Type"]
         Y

Out[16]: 0        Light_Load
         1        Light_Load
         2        Light_Load
         3        Light_Load
         4        Light_Load
                     ...
         35035    Light_Load
         35036    Light_Load
         35037    Light_Load
         35038    Light_Load
         35039    Light_Load
         Name: Load_Type, Length: 35040, dtype: object
```

Fig 5.6.2

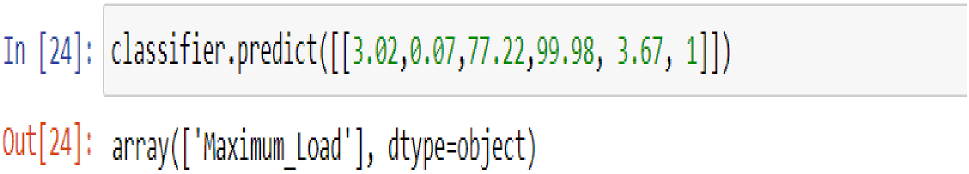
After defining my X&Y, I then applied decision tree algorithm:



```
In [17]: from sklearn.model_selection import train_test_split
         X_train, X_test, Y_train, Y_test =  train_test_split(X,Y,test_size = 0.25, random_state= 0)

In [18]: from sklearn.preprocessing import StandardScaler
         sc_X = StandardScaler()
         X_train = sc_X.fit_transform(X_train)
         X_test = sc_X.transform(X_test)

In [19]: from sklearn.tree import DecisionTreeClassifier
         classifier = DecisionTreeClassifier()
         classifier = classifier.fit(X_train,Y_train)

In [24]: Y_pred = classifier.predict(X_test)
         from sklearn import metrics
         print('Accuracy Score:', metrics.accuracy_score(Y_test,Y_pred))

         Accuracy Score: 0.7160958904109589
```

Fig 5.6.3

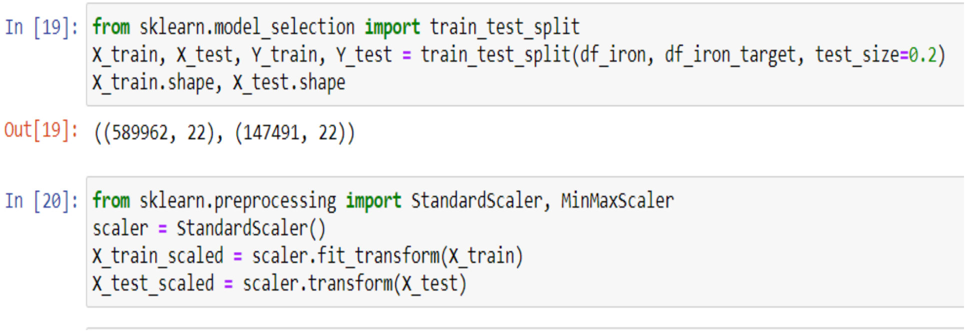
Then I tested for a dummy data:



```
In [24]: classifier.predict([[3.02,0.07,77.22,99.98, 3.67, 1]])

Out[24]: array(['Maximum_Load'], dtype=object)
```

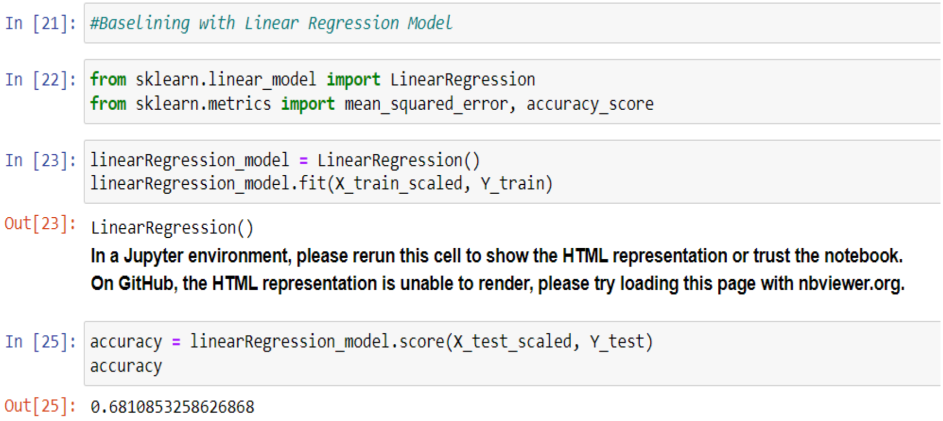
Fig 5.6.4

5.7 APPLY RANDOM FOREST



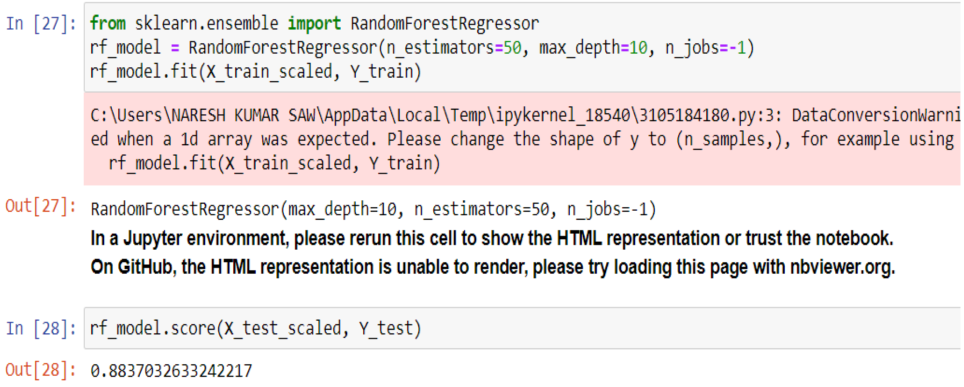
```
In [19]: from sklearn.model_selection import train_test_split
         X_train, X_test, Y_train, Y_test = train_test_split(df_iron, df_iron_target, test_size=0.2)
         X_train.shape, X_test.shape

Out[19]: ((589962, 22), (147491, 22))

In [20]: from sklearn.preprocessing import StandardScaler, MinMaxScaler
         scaler = StandardScaler()
         X_train_scaled = scaler.fit_transform(X_train)
         X_test_scaled = scaler.transform(X_test)
```

Fig 5.7.1



```
In [21]: #Baselining with Linear Regression Model

In [22]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import mean_squared_error, accuracy_score

In [23]: linearRegression_model = LinearRegression()
         linearRegression_model.fit(X_train_scaled, Y_train)

Out[23]: LinearRegression()
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [25]: accuracy = linearRegression_model.score(X_test_scaled, Y_test)
         accuracy

Out[25]: 0.6810853258626868
```

Fig 5.7.2



```
In [27]: from sklearn.ensemble import RandomForestRegressor
         rf_model = RandomForestRegressor(n_estimators=50, max_depth=10, n_jobs=-1)
         rf_model.fit(X_train_scaled, Y_train)

         C:\Users\NARESH KUMAR SAW\AppData\Local\Temp\ipykernel_18540\3105184180.py:3: DataConversionWarni
         ed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using
           rf_model.fit(X_train_scaled, Y_train)

Out[27]: RandomForestRegressor(max_depth=10, n_estimators=50, n_jobs=-1)
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [28]: rf_model.score(X_test_scaled, Y_test)

Out[28]: 0.8837032633242217
```

Fig 5.7.3

We successfully implemented Random Forest.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 INTRODUCTION

This chapter will discuss the model's results and outcomes, as well as an appraisal of the proof of theory, future work to develop and design Steel consumption & Energy consumption predictors using Machine Learning Algorithms.

6.2 ALGORITHMS USED

For predicting the Steel consumption different algorithms were used like:

- KNN
- Multiple Linear Regression

For predicting the steel energy algorithms used:

- Decision Tree Classifier

For predicting the quality of mining of iron ore

- Random Forest

6.3 CHALLENGES

- Steel data is very sensitive and not easily available. Even the reputed constitutions do not release their data publicly.
- Procuring data from the years before 2018 for the consumption prediction was problematic because of its sensitive nature.
- Categorizing a mass data like steel consumption is difficult.
- Choosing the correct algorithm for a mass data like this is very challenging.
- Iron ore deposits can vary significantly in their composition, including the presence of different minerals, impurities, and ore grades. This fluctuation

can affect the nature of the separated metal, as well as the effectiveness of the mining system. Precisely portraying the fluctuation in metal organization and understanding its effect on mineral quality can be testing, and may require broad examining, examination, and demonstrating.

6.4 FUTURE WORKS

With a plant with a production history of more than 50 yrs. where machines and operating devices have been systematically modernized over the time. Some machines can be easily connected to the system while others would require a chip for connection. Automation is always a step wise process and there is no shortcut to it. Our current goal is to make data and AI driven organizations. Our goal is always to be resilient to continuous changing scenarios and markets as these digital technologies are fast changing. Peer industry has already been using these digital technologies to increase productivity, reducing manpower and streamlining the overall process. Being capital, and labor intensive is a blessing in disguise as new entrants are very low in this sector. Good performing systems need to be replicated which have high ROI, while investment towards future goals need to be continued. It’s never too late to accommodate change but faster it is done, healthier it is for the industry. Change is feared by many but it is the ultimate truth of survival. If we don’t start changing today there are high chances that we may perish. If we can’t be leaders then we must be fast followers and it can be a game changer. Roadmap to digital transformation should be done in a clear goal setup. It should come in a centralized way in corporate office directives.

Now in the project, many data model was successfully created that correctly predicted the consumption status for various years by analyzing the previous year’s dataset.

In the future, there is potential for the application of machine learning techniques to further enhance the determination of mining quality of iron ore. Machine learning algorithms can be utilized to develop predictive models that can analyze large datasets related to ore composition, mining parameters, and environmental factors, to identify patterns and correlations that may impact ore quality. These models can aid in optimizing mining operations by providing real-time feedback on ore quality, identifying optimal mining strategies, and predicting potential quality issues. Additionally, machine learning can be utilized for automated image analysis and sensor-based monitoring of mining operations, allowing for continuous assessment of ore quality throughout the mining process. Machine learning can also be employed for decision support systems that aid in the selection of mining sites, equipment allocation, and resource allocation for optimizing ore quality. However, future research and development efforts would be required to address challenges such as data quality, model interpretability, and regulatory compliance, to ensure the responsible and effective use of machine learning for determining the mining quality of iron ore. This is just a prototype/idea that can be implemented in a larger perspective if there is data availability for the previous years.

APPENDIX

1. STEEL CONSUMPTION PREDICTION

Packages installed:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Data loading:

```
data = pd.read_csv('Steeldata.csv')
data
```

Data preprocessing

```
for i in range(len(data["Country"])):
  data["Country"][i] = 0

print(data['Type'].unique())
GI, HR, SS, BR = 'Galvanized Iron Coil', 'HR Coil', 'Stainless Steel', 'Bars and Rods'
for i in range(len(data["Type"])):
    if data['Type'][i] == GI:
      data["Type"][i] = 1
    if data['Type'][i] == HR:
      data["Type"][i] = 2
    if data['Type'][i] == SS:
      data["Type"][i] = 3
    if data['Type'][i] == BR:
      data["Type"][i] = 4

ff= data['State'].unique()
m = {i:0 for i in ff}
c = 10
for i in ff:
  m[i]+=c
  c+=1
print(m)
```

Applying Algorithm:

```
model1 = KNeighborsClassifier(n_neighbors=5)
model1.fit(X_train, Y_train)

model2 = linear_model.LinearRegression()
model2.fit(X_train, Y_train)
```

Accuracy:

```
print(classification_report(y_test,y_pred))

score = r2_score(Y_test, Y_pred)
print(score)
```

2. STEEL ENERGY CONSUMPTION

Package loading:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Data loading:

```
data = pd.read_csv('steel_energy_data.csv')
data.head()
```

Data preprocessing:

```
data = data.rename(columns={'Lagging_Current_Reactive.Power_kVarh': 'LagReactivePower',
                        'Leading_Current_Reactive_Power_kVarh': 'LeadReactivePower',
                        'Lagging_Current_Power_Factor': 'LagPowerFactor',
                        'Leading_Current_Power_Factor': 'LeadPowerFactor',
                 'CO2(tCO2)':'CO2'})

ff= data['Day_of_week'].unique()
m = {i:0 for i in ff}
c = 1
for i in ff:
  m[i]+=c
  c+=1
print(m)
```

Applying algorithm:

```
model3 = DecisionTreeClassifier()
model3 = model3.fit(X_train,Y_train)
```

Accuracy:

```
print('AccuracyScore:',metrics.accuracy_score(Y_test,Y_pred))
```

3. MINING QUALITY OF IRON ORE

Import package:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

Data loading:

```
data = pd.read_csv('ironOreQuality.csv', decimal = ',',parse_dates = ['date'] )
```

Applying Algorithm:

```
model4 = RandomForestRegressor(n_estimators=50, max_depth=10, n_jobs=-1)
model4.fit(X_train_scaled, Y_train)
```

Accuracy:

```
model4.score(X_test_scaled, Y_test)
```

REFERENCES

[1] BCG. (2015). How Will Technology Change the Industrial Workforce Through 2015? Man and Machine in Industry 4.0.

[2] Fox, W. & Bayat, M.S. (2007). “A Guide to Managing Research”. Juta Publications, p.45

[3] I. Bojanova, "The Digital Revolution: What's on the Horizon?". IT Professional, vol. 16, no. 1, pp. 8-12, Jan.-Feb. 2014.

[4] Urbas, L. (2017). Digital transformation in industrial processes – Challenges and first solutions. ATP EDITION, (12), 54-65.

[5] Wikipedia. (2020). Steel Making. Retrieved March 11, 2020, from https://en.wikipedia.org/wiki/Steelmaking

[6] World Steel Association. (2019). World Steel in Figures 2019
https://aircconline.com/ijcses/V8N2/8217ijcses01.pdf
http://article.nadiapub.com/JSEM/vol1_no2/2.pdf

[7] Dianmin Zhou & Feng Gao (2012). Daily electricity consumption forecast for a steel corporation based on NNLS with feature selection, pp.1-4, 2012.

[8] A. G. Malanichev & P. V. Vorobyev (2011). Forecast of global steel prices, 304 (2011)

[9] Yanni Xuan & Qiang Yue (2016). Forecast of steel demand and the availability of depreciated steel scrap in China.

[10] Dr. Shrabanti Pal (2013). A study on performance and prospect of Indian steel industry from national perspective under globalization.

[11] Dr. Shailendra Kumar Chaturvedi & Ms. Suruchi Tripathi (2018). Government policy and FDI triggering growth opportunities of iron steel in India.

[12] G.S. Popli & Rupina Popli (2015). Investment Opportunities in the Steel Sector of India on the Concept of Make in India.

[13] Leo S. Carlsson (2021). Applied Machine Learning in Steel Process Engineering.

[14] Giacomo Pellegrini & Matteo Sandri (2019). Successful Use Case Applications of Artificial Intelligence in the Steel Industry.

[15] Rahman A.B.M Salman Rahman & Moonyong Lee (2020).A prediction model for steel factory manufacturing product based on energy consumption using data mining

technique.

[16] Sai Aravind Sarswatula & Tanna Pugh (2022).Modeling Energy Consumption Using Machine Learning.

[17] Zhao J, Han Z, Pedrycz W, Wang W. Granular Model of Long-Term Prediction for Energy System in Steel Industry. IEEE Trans Cybern. 2016 Feb;46(2):388-400. doi: 10.1109/TCYB.2015.2445918. Epub 2015 Jul 7. PMID: 26168454.

[18] Yanyan Zhang, Xiaolei Zhang, Lixin Tang, Energy Consumption Prediction in Ironmaking Process Using Hybrid Algorithm of SVM and PSO. Advances in Neural Networks – ISNN 2012, 2012, Volume 7368 ISBN : 978-3-642-31361-5

[19] Zeng, Bo & Zhou, Meng & Zhang, Jun. (2017). Forecasting the Energy Consumption of China’s Manufacturing Using a Homologous Grey Prediction Model. Sustainability. 9. 1975. 10.3390/su9111975.

[20] Reimann, Jan & Wenzel, Ken & Friedemann, Marko & Putz, Matthias. (2018). Methodology and model for predicting energy consumption in manufacturing at multiple scales. Procedia Manufacturing. 21. 694-701. 10.1016/j.promfg.2018.02.173.

[21] Sen, Parag & Roy, Mousumi & Pal, Parimal. (2016). Application of ARIMA for forecasting energy consumption and GHG emission: A case study of an Indian pig iron manufacturing organization. Energy. 116. 1031–1038. 10.1016/j.energy.2016.10.068.

[22] P. Karthigaikumar. (2021). Industrial Quality Prediction System through Data Mining Algorithm. https://doi.org/10.36548/jei.2021.2.005Journal of Electronics and Informatics