KEDAR HARDIKAR:21070126030

DEVANSH TIWARI: 21070126025

DHRUBO Bhattacharjee: 21070126026

PROJECT REPORT

*Sports Star Net Worth Predictor*

Table of Contents

**Introduction**

Context3 Motivation……………………………………………………………………………………………3

Idea……………………………………………………………………………………………………3

Sources………………………………………………………………………………………………. 3

**The Project**

Data4

Data Acquisition ……………………………………………………………………………………5

Data Cleaning 5

Train Test Split …………………………………………………………………………………….5

Separating Features and labels …………………………………………………………………5

Exploratory Data Analysis ………………………………………………………………………… 6

Machine Learning ………………………………………………………………………………….10

Sampling …………………………………………………………………………………………10

Creating a pipeline …………………………………………………………………………….11

Standardization ………………………………………………………………………………….13

Learning Algorithm …………………………………………………………………………….14

Introduction …………………………………………………………………………………… 14

Pros …………………………………………………………………………………………….. 17

Cons ……………………………………………………………………………………………. 18

Where do we use random forest regressor ……………………………………………… 19

Math behind the algorithm …………………………………………………………………. 19

Conclusion ……………………………………………………………………………………. 21

1. **INTRODUCTION:**
   1. CONTEXT:

This project has been done as part of my course for the B. Tech Artificial Intelligence and Machine Learning at Symbiosis Institute of Technology, Pune. Supervised by Dr Preeti Mulay, Prof Kalyani Kadam, Dr Anupkumar Bongale. Myself (Kedar Hardikar) along with my teammates Devansh Tiwari and Dhrubo Bhattacharjee had two months to fulfill the requirements in order to succeed the module. Regular follow up was conducted by the teachers in order to mark our progress.

* 1. MOTIVATION:

Being extremely interested in everything having a relation with AI and Machine Learning, such a project was a great occasion to give me the time to learn and confirm my interest for this field. The fact that we can make estimations, predictions and give the ability for machines to learn by themselves is both powerful and limitless in term of application possibilities. Machine Learning can be used in a wide range of fields like Finance, Medicine, Entrepreneurship, Businesses, almost everywhere.

* 1. IDEA:

As this was our first AI/ML project our main aim was to explore the vastness of this field and try to learn the basic structure of a ML model. We chose to take ‘Real Time Athlete’s market value’ as our topic. Even though this is a very simple idea, it gives a perfect balance between learning how a model works while also aligning it with our interests. As the name suggests, the goal was to find the approximate market value of an athlete. In this project we have used a football players’ s dataset in order to find their market value.

* 1. SOURCES:

Because I truly think that sharing sources and knowledges allow to help others but

also ourselves, the sources of the project are available at the following link:

<https://github.com/kedarhardikar/Predict-Players-value-ML-Project>

Feel free to give me your point of view or ideas for anything you want.

1. **THE PROJECT:**
   1. DATA:

Data is the most important part of all Data Analytics, Machine Learning, Artificial Intelligence. Without data, we can't train any model and all modern research and automation will go in vain. Big Enterprises are spending lots of money just to gather as much certain data as possible.

Many sub steps are taken to get, clean and transform the data. I am going to explain each one of them to show how they have been applied on my project why they are useful for the machine learning part.

Know the data:

1. Overall: It indicates how good a player is. The highest overall in the used dataset is 91 for Lewendowski. For example: a very good player will have a very high overall whereas an average player will have lower overall.
2. Potential: The potential feature tells us how much potential a player has to become world class. The highest potential in the used dataset is 95 for Kylian Mbappe.
3. wage\_eur: The weekly wage of the player
4. League level: The quality of the league the particular player plays. 1 meaning the highest level and 5 meaning the lowest level.
5. Club Contract valid until: The year till which the player has signed for the club.
6. Weak foot: Higher the value, better is the quality of the players weaker foot.
7. Dribbling: The ability of a player to go past other players
8. Attacking: The mean of a player’s attacking traits
9. Skill: The mean of a players skill traits
10. Movement: The mean of a player’s movement on and off the ball
11. Power: The mean of a player’s physical traits
12. Mentality: The mean of a player’s mental strength traits
13. Defending: The mean of a player’s defending traits
    * 1. DATA ACCUSITION:

The first problem was where can we get the data to build a large enough dataset so that our ML model works with a very low error rate. After going through lots of datasets on GitHub, Kaggle, we chose *https://www.kaggle.com/datasets/bryanb/fifa-player-stats-database* as our dataset. Our course required us to acquire this data using web scraping which is a technique of extracting information from websites.

Web scraping is an automated method used to extract large amounts of data from websites. The data on the websites are unstructured. Web scraping helps collect these unstructured data and store it in a structured form. There are different ways to scrape websites such as online Services, APIs or writing your own code.

As the dataset couldn’t be scraped, we scraped another website having some information about football transfer market. *https://www.kaggle.com/datasets/bryanb/fifa-player-stats-database*

* + 1. DATA CLEANING:

First, we removed all the columns which were not required or showed very less correlation with the label i.e. the Value of the football player.

Then we converted the categorical data into numerical data. This step is important as a ML model doesn’t work with categorical data. This was done using the pandas.get\_dummies() function which creates new columns which are numerical. Then dropping the original categorical column.

Null values needed to be removed as a ML model needs complete data. As the data was vast we decided to drop the rows which had any null values instead of using different imputing techniques like simple imputer or imputing the null values with mean/ median.

* + 1. TRAIN TEST SPLIT:

Every ML model requires a training set and a testing set. The training set is the data which is used to train the model to make predictions. We compare the predictions of our model with the actual label and find out how of our model works using different methods. The test data is used to find out how the model works.

We used 70% of our data as our training data and the remaining as test data.

So, our training data included 11,214 rows and the test dataset included 4,806 rows.

* + 1. SEPERATING FEATURES AND LABELS

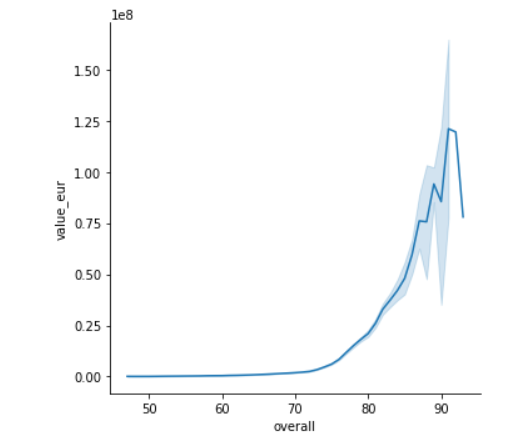
We separated the features and labels of the training dataset. This is necessary as the steps required in training a model requires features and labels as two separate parameters

* 1. EXPLORATORY DATA ANALYSIS:

Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

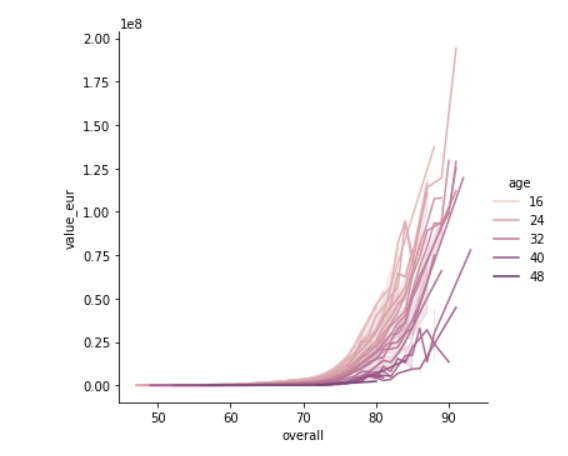
As a person who understands football, I wanted to check the reality of football transfer market with the acquired data.

1. **As the overall of a player increases the value of the player should increase.**

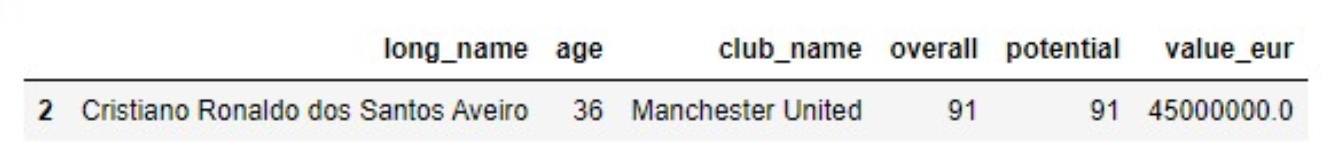


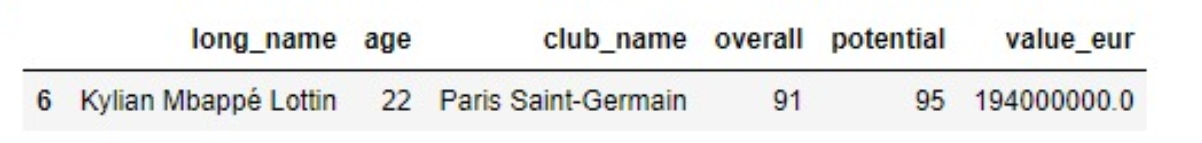
We can see that this line is accurate till the overall of around 90 but after that , the value starts dropping. This is because the players with overall>90 are fairly old, this leads to decrease in their value. We will confirm this assumption in the following graph:

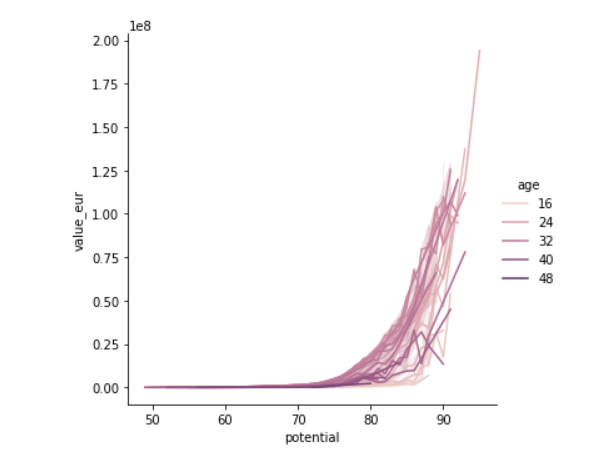
1. **As the overall of the player increases, with decrease in age, the players value must increase.**



This plot confirms the reality. We can also observe the same from the below examples of the overall and values of Cristiano Ronaldo and Mbappe. Even though they both have the same overall, as Mbappe is much younger than Cristiano, he is valued higher.

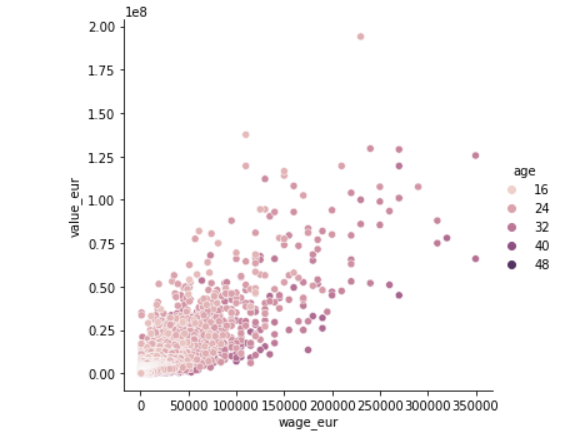




1. **As the potential of a player increases, the value of the player must also increase. But if the player is young and has high potential then the value of the player might be even higher.** 

This plot confirms the reality. We can see the younger players having a potential around 90 are higher valued than the older players having the same potential.

1. **There Is a very mixed observation between the wage of the player and their value and age.**



We can see this peculiar behavior in the above graph

After cleaning, processing and understanding the data, next step is machine learning.

* 1. MACHINE LEARNING

The Machine Learning part is about trying to find the best learning algorithm for a given problem even if it is highly conditioned by how well the data has been processed and tune some parameters to improve it. Depending on the problem, if it is supervised (meaning we build a model from labeled training set, the value of the dependent variable is known) or if it unsupervised (the model is built on unstructured and unlabeled data), if it is a regression or classification problem, many learning algorithm exist each with their benefits and drawbacks.

* + 1. SAMPLING:

Given my dataset, I applied a sampling technique in order to divide it into different subset having each its own utility. It is commonly assumed that more we have data to build a model more it will have tend to give good results. Usually, the dataset is divided into 3 parts: Training set, Cross set, Test Set. But we decided to divide our dataset into just the Training set and Testing set to make things easier for us.

A picture containing text, screenshot, businesscard

Description automatically generated

The training set as its name indicates it is used to train the learning algorithm.

The test set is used only to see how well the learning algorithm is generalized,

meaning how it performs with unknown data.

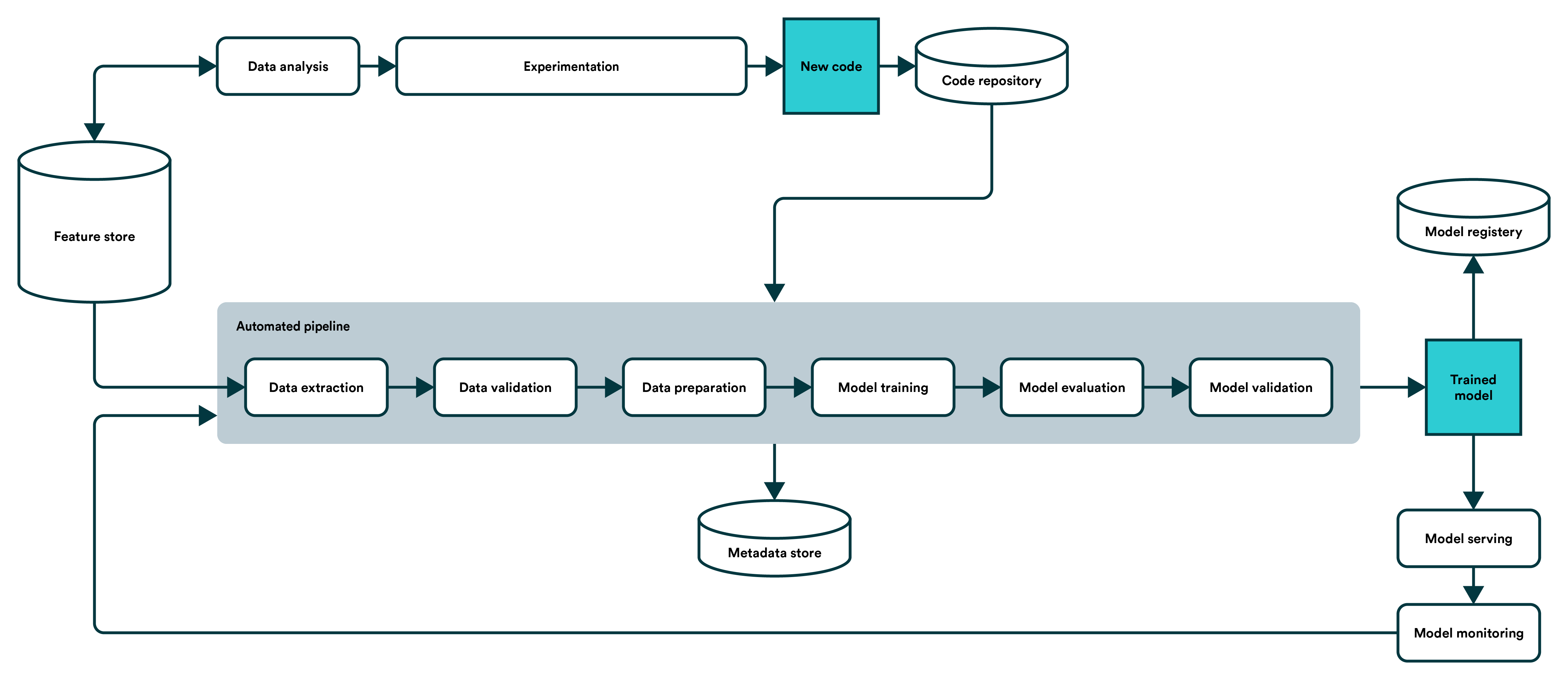
* + 1. Creating a Pipeline:

A machine learning pipeline is a way to codify and automate the workflow it takes to produce a machine learning model. Machine learning pipelines consist of multiple sequential steps that do everything from data extraction and preprocessing to model training and deployment.

For data science teams, the production pipeline should be the central product. It encapsulates all the learned best practices of producing a machine learning model for the organization’s use-case and allows the team to execute at scale. Whether you are maintaining multiple models in production or supporting a single model that needs to be updated frequently, an end-to-end machine learning pipeline is a must.

It is beneficial to look at the stages which many data science teams go through to understand the benefits of a machine learning pipeline. Implementing the first machine learning models tends to be very problem-oriented, and data scientists focus on producing a model to solve a single business problem, for example, classifying images.

Once teams move from a stage where they are occasionally updating a single model to having multiple frequently updating models in production, a pipeline approach becomes paramount. In this workflow, you don’t build and maintain a model. You develop and maintain a pipeline. The pipeline is the product.



An automated pipeline consists of components and a blueprint for how those are coupled to produce and update the most crucial component – the model.

In the automated workflow, solid engineering principles become more into play. The code is split into more manageable components, such as data validation, model training, model evaluation, and re-training triggering.

The system offers the ability to execute, iterate, and monitor a single component in the context of the entire pipeline with the same ease and rapid iteration as running a local notebook cell on a laptop. It also lets you define the required inputs and outputs, library dependencies, and monitored metrics.

This ability to split the problem solving into reproducible, predefined, and executable components forces the team to adhere to a joined process. A joined process, in turn, creates a well-defined language between the data scientists and the engineers and also eventually leads to an automated setup that is the ML equivalent of continuous integration (CI) – a product capable of auto-updating itself.

Characteristics of an automated ML pipeline:

• The pipeline is the product

• Fully automated process

• Co-operation between the data scientist and the engineer

• Fast iteration cycle

• Automated testing and performance monitoring

• Version-controlled

Building a pipeline:

1. Build every step into reusable components.

Consider all the steps that go into producing your machine learning model. Start with how the data is collected and preprocessed and work your way from there. It’s generally encouraged to limit each component’s scope to make it easier to understand and iterate.

2. Don’t forget to codify tests into components.

Testing should be considered an inherent part of the pipeline. If you, in a manual process, do some sanity checks on how the input data and the model predictions should look like, you should codify this into a pipeline. A pipeline gives opportunities to be much, much more thorough with testing as you will not have to perform them manually each time.

3. Tie your steps together.

There are many ways to handle the orchestration of a machine learning pipeline, but the principles remain the same. You define the order in which the components are executed and how inputs and outputs run through the pipeline.

4. Automate when needed.

While building a pipeline already introduces automation as it handles the running of subsequent steps without human intervention, for many, the ultimate goal is also to automatically run the machine learning pipeline when specific criteria are met. For example, you may monitor model drift in production to trigger a re-training run or – simply do it more periodically, like daily.

Depending on your specific use case, your final machine learning pipeline might look different. For example, you might train, evaluate and deploy multiple models in the same pipeline. There are common components that are similar in most machine learning pipelines.

As this was a very new concept to us, we just decided to create a pipeline with the standard scalar function.

* + 1. Standardization:

In the pipeline we fitted the standard scalar function.

Scaling of Features is an essential step in modeling the algorithms with the datasets. The data that is usually used for the purpose of modeling is derived through various means such as:

• Questionnaire

• Surveys

• Research

• Scraping, etc.

So, the data obtained contains features of various dimensions and scales altogether. Different scales of the data features affect the modeling of a dataset adversely.

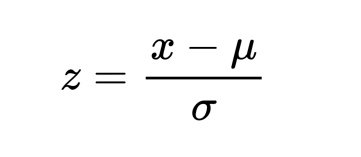
It leads to a biased oux`tcome of predictions in terms of misclassification error and accuracy rates. Thus, it is necessary to Scale the data prior to modeling.

This is when standardization comes into picture.

Standardization is a scaling technique wherein it makes the data scale-free by converting the statistical distribution of the data into the below format:

• mean - 0 (zero)

• standard deviation - 1



By this, the entire data set scales with a zero mean and unit variance, altogether.

Let us now try to implement the concept of Standardization in the upcoming sections.

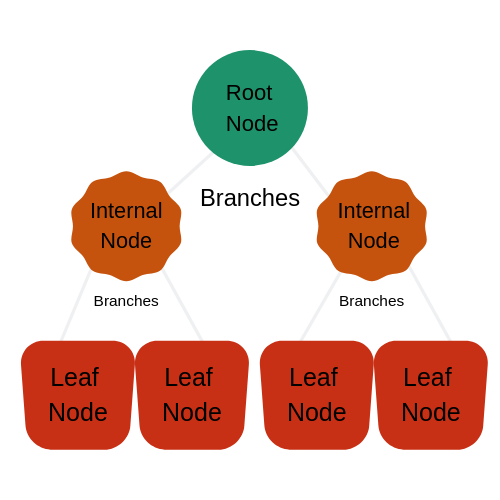
* + 1. Learning Algorithm used:

Among the large choice of learning algorithms, I chose to use Random Forest Regression because my dependent variable is continuous. A dependent variable is a variable that we try to predict, and it depends on other variables.

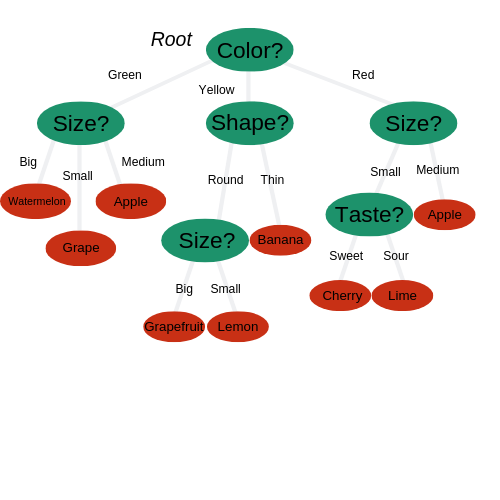
1. **Introduction**

Have you ever asked yourself a series of questions in order to help make a final decision on something? Maybe it was a simple decision like what you wanted to eat for dinner. You might have asked yourself if you wanted to cook or pick food up or get delivery. If you decided to cook, then you would have needed to figure out what type of cuisine you were in the mood for. And lastly, you probably needed to figure out if you had all of the ingredients in your fridge or needed to make a run to the store. Finding the answer to these questions would have helped you come to a final decision on dinner that night.

We all have to use this decision-making process multiple times, every single day. In the machine learning world this process is called a*decision tree*. You start with a node which then branches to another node, repeating this process until you reach a leaf. A node asks a question in order to help classify the data. A branch represents the different possibilities that this node could lead to. A leaf is the end of a decision tree, or a node that no longer has any branches.



The Random Forest Algorithm is composed of different decision trees, each with the same nodes, but using different data that leads to different leaves. It merges the decisions of multiple decision trees in order to find an answer, which represents the average of all these decision trees.



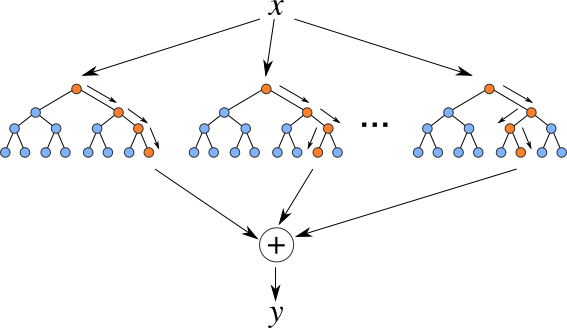
The random forest algorithm is a [supervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning) model; it uses labelled data to “learn” how to classify unlabelled data. This is the opposite of the K-means Cluster algorithm, which we learned in a past article was an unsupervised learning model. The Random Forest Algorithm is used to solve both regression and classification problems, making it a diverse model that is widely used by engineers.

**B) Pros:**

1. Used for regression and classification problems, making it a diverse model.
2. Prevents overfitting of data.
3. Fast to train with test data.

**C) Cons:**

1. Slow in creating predictions once model is made.
2. Must beware of outliers and holes in the data.



In the above example, we have three individual decision trees which together make up a Random Forest. Random Forest is considered ensemble learning, meaning it helps to create more accurate results by using multiple models to come to its conclusion. The algorithm uses the leaves, or final decisions, of each node to come to a conclusion of its own. This increases the accuracy of the model since it’s looking at the results of many different decision trees and finding an average.

**C) Where do we use Random Forest Regressor:**

Let’s say you want to estimate the average household income in your town. You could easily find an estimate using the Random Forest Algorithm. You would start off by distributing surveys asking people to answer a number of different questions. Depending on how they answered these questions, an estimated household income would be generated for each person.

After you’ve found the decision trees of multiple people you can apply the Random Forest Algorithm to this data. You would look at the results of each decision tree and use random forest to find an average income between all of the decision trees. Applying this algorithm would provide you with an accurate estimate of the average household income of the people you surveyed.

**E) Error metrics:**

An error metric is a way to quantify the performance of a model and provides a way for the forecaster to quantitatively compare different models1. They give us a way to more objectively gauge how well the model executes its tasks. There are three error metrics that are commonly used for evaluating and reporting the performance of a regression model; they are: Mean Squared Error (MSE). Root Mean Squared Error (RMSE). Mean Absolute Error (MAE).

We have used MSE and RMSE in this project.

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Root mean square error is commonly used in climatology, forecasting, and regression analysis to verify experimental results.

The formula is

Text

Description automatically generated with low confidence

Where: f = forecasts (expected values or unknown results),

o = observed values (known results).

To find the rmse value, firstly we find the mean square value by simply calling the mean\_squared\_error function and put actual labels and the predicted labels as the parameters, then we take the square root of this to find the rmse.

Model.score is a method used to determine how well a model works. It simply takes the input features and the real labels as parameters.

**F) Result:**

The rmse value of our model after training is 443927.23661913007 and its score is 0.9967240682337638. On testing our model the rmse value is 835332.201402942 and its score is 0.9889204872218451.

**G) Conclusion:**

It is essential to understand a single decision tree before you can fully understand the random forest algorithm. You must understand the difference between a node, branch and leaf, and how the different formulas are applied in order to come to a final decision.

When used correctly, the random forest algorithm can be extremely useful with all different types of data sets, whether regression or classification data. It is easy to use, fast to train, and finds an accurate representation of the decision trees it is using.