**PUBG Finish Placement Prediction**

(To predict the battle royale finish of PUBG Players)

Thesis submitted in partial fulfillment of the

requirements for

**Post Graduate Diploma in Data Science**

By

(Signature)

**Suhail AK**

17225760070

Under the guidance of

Mr. Vishnu K,

Manipal Hospital,

Bangalore



**MANIPAL ACADEMY OF HIGHER EDUCATION, MANIPAL**

**PUBG Finish Placement Prediction**

(To predict the battle royale finish of PUBG Players)

Thesis submitted in partial fulfillment of the

requirements for

**Post Graduate Diploma in Data Science**

By

**Suhail AK**

17225760070

**Examiner 1**   **Examiner 2**

Signature: Signature:

Name: Name:



**MANIPAL ACADEMY OF HIGHER EDUCATION, MANIPAL**

**CERTIFICATE**

This is to certify that the project work titled

**PUBG Finish Placement Prediction**

(To predict the battle royale finish of PUBG Players)

is a bonafide record of the work done by

**Suhail AK**

17225760070

In partial fulfillment of the requirements for the award of **Post Graduate Diploma in Data Science** under Manipal Academy of Higher Education, Manipal, Manipal and the same has not been submitted elsewhere for any kind of certification/recognition.

(Signature)

Mr. Vishnu K

Manipal Hospital

Bangalore

Table of Contents

Acknowledgments

Abstract

1. Introduction
   1. Background
   2. Project Goal
   3. Business Understnading
2. Project Description and Tools
   1. Datasets understanding
   2. Data Limitations
   3. Benefits of project
   4. Jupyter Notebook
3. Exploratory Data Analysis
   1. Data collection
   2. Variable Identification
   3. Univariate Analysis
   4. Bivariate Analysis
   5. Impact of Outlier on Dataset
   6. Missing value Treatment
4. Feature Engineering
   1. Introduction
   2. Creating New Features
      1. PlayerJoined Feature
      2. Total Items and Total Distance Feature

4.3.3 Mean Match and Median Feature

4.3.4 List of New features added

1. Model Building
   1. Introduction to Light GBM
   2. Understanding of Light GBM Algorithm
   3. Implementation of Light GBM
   4. Parameters of Light GBM Algorithm
2. Conclusion
   1. Summary of the Project Outcome
3. Future Scope

References

**ACKNOWLEDGMENT**

I feel a sense of satisfaction and immense pleasure for completing this project work. I would not have completed this task without the assistance of my supervisor Mr. Vishnu K, Manipal Pro Learn.

So, I take this opportunity to express our gratitude for him who has guided us in each and every step. The project was born through the encouragement of my guide and lived through her guidelines and supervision. We thank him for all the time he gave me in spite of his busy schedule. I would like to express our gratitude to Dr. Ramesh Babu, Director, Manipal Academy of Data Science for providing a very good motivation

and environment for this research work. I also extend my thanks to the faculty

members of Manipal for the invaluable knowledge they have imparted on me.

Last but not at least I am grateful to almighty God to give us strength and blessings to complete the project successfully. Also, sincere thanks to my parents for supporting me in every work.

**ABSTRACT**

Battle Royale-style video games have taken the world by storm. 100 players are dropped onto an island empty-handed and must explore, scavenge, and eliminate other players until only one is left standing, all while the play zone continues to shrink.

PlayerUnknown's BattleGround s (PUBG) has enjoyed massive popularity. With over 50 million copies sold, it's the fifth best selling game of all time, and has millions of active monthly players.

The team at PUBG has made official game data available for the public to explore and scavenge outside of "The Blue Circle." This competition is not an official or affiliated PUBG site - Kaggle collected data made possible through the PUBG Developer API.

You are given over 65,000 games' worth of anonymous player data, split into training and testing sets, and asked to predict final placement from final in-game stats and initial player ratings.

What's the best strategy to win in PUBG? Should you sit in one spot and hide your way into victory, or do you need to be the top shot? Let's let the data do the talking!

Chapter 1

Introduction

* 1. Background:

The gaming industry is no longer a niche arena for a certain age group or consumer segment. With the advent of mobile gaming and improvements to hardware used in playing these games, gaming has become a viable form of entertainment for players from all backgrounds and ages. This switch to mainstream has also meant an increase in revenues generated by the industry with about US $9.5 billion generated in the United States in 2007, 11.7 billion on 2008 and 25.1 billion in 2010.

The improvements to hardware such as sound cards, graphics and faster processors have meant a related growth and development of the gaming industry as well. As a result, modern games, especially those that are PC based, have become very demanding as applications and serious gamers are among those who purchase high-powered personal computers to keep up with the newest games.

PlayerUnknown's Battlegrounds (PUBG) is an online multiplayer battle royale game developed and published by PUBG Corporation, a subsidiary of South Korean video game company Bluehole. The game is based on previous mods that were created by Brendan "PlayerUnknown" Greene for other games using the film Battle Royale for inspiration, and expanded into a standalone game under Greene's creative direction. In the game, up to one hundred players parachute onto an island and scavenge for weapons and equipment to kill others while avoiding getting killed themselves. The available safe area of the game's map decreases in size over time, directing surviving players into tighter areas to force encounters. The last player or team standing wins the round.

The game was first released for Microsoft Windows via Steam's early access beta program in March 2017, with a full release on December 20, 2017. That same month, the game was released by Microsoft Studios for the Xbox One via its Xbox Game Preview program, and officially released in September 2018. The same year, two different mobile versions based on the game for Android and iOS were release as well as plans for a port for the PlayStation 4. The game is one of the best selling of all time, with over fifty million sold across all platforms by June 2018. In addition, the Windows version holds a peak concurrent player count of over three million on Steam, which is an all-time high on the platform.

* 1. Project Goal:

The Gaming industry is the key industry in terms of generating revenue Idea here is to predict the placement of the one of the most popular game in the market right now i.e. PlayerUnknowns Battle Grounds. We have been given a huge dataset for training with 4 million+ rows and 20+ columns which contains the post game data of PUBG players with over 65,000 games worth of data and using these data we have to do placement percentage for those 65000 games for each player. This is a classic Regression problem with Mean Absolute Error(MAE) as its measuring metrics.

* 1. Business Understanding:
* Capturing and retention of customers(gamers)
* 87 million daily active users
* 50million units sold online for PC where as around 400 million unit for free mobile version
* Improving InApp purchasing for potential customers

Chapter – 2

Dataset and Tools

2.1 Dataset Understanding:

You are given over 65,000 games' worth of anonymous player data, split into training and testing sets, In a PUBG game, up to 100 players start in each match (matchId). Players can be on teams (groupId) which get ranked at the end of the game (winPlacePerc) based on how many other teams are still alive when they are eliminated. In game, players can pick up different munitions, revive downed-but-not-out (knocked) teammates, drive vehicles, swim, run, shoot, and experience all of the consequences -- such as falling too far or running themselves over and eliminating themselves.

You must create a model, which predicts players' finishing placement based on their final stats, on a scale from 1 (first place) to 0 (last place).

2.2 Dataset Limitations:

You are provided with a large number of anonymous PUBG game stats, formatted so that each row contains one player's post-game statistics. The data comes from matches of all types: solos, duos, squads, and custom; there is no guarantee of there being 100 players per match nor at most 4 player per group this is because of the reason that few players AFK(away from keyboard) once the game starts,

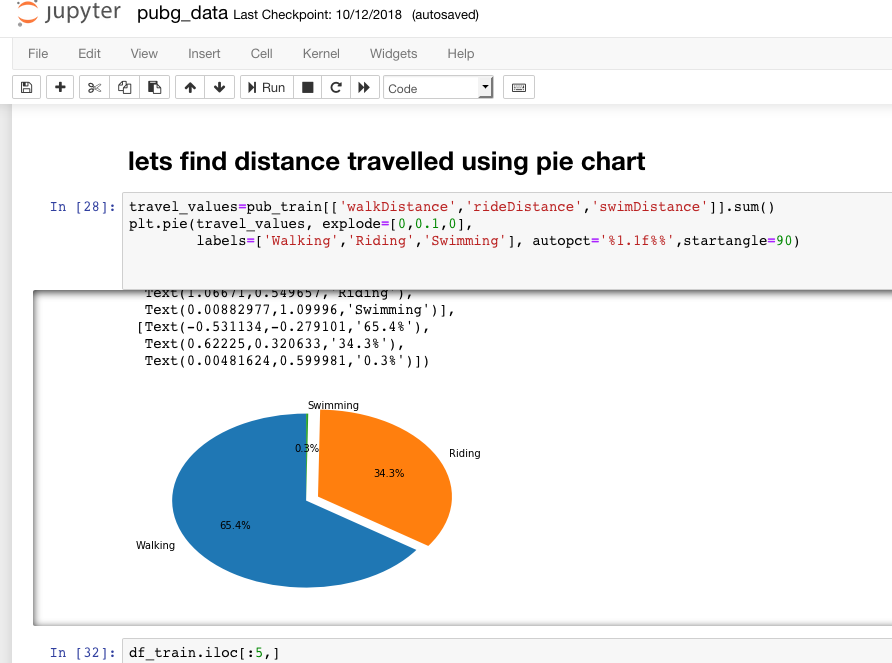
2.3 Benefits of the project:

The main aim of the project is to create a model which predicts players finishing placement based on their final stats, on a scale from 1 (first place) to 0 (last place i.e. 100th place considering 100 players joined that particular game).

From the EDA perspective it is possible to improve InApp purchasing for the potential customers also Capturing ,retaintion of customers .

2.4 Jupter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. The Notebook has support for over 40 programming languages, including Python, R, Julia, and Scala. Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer. Your code can produce rich, interactive output: HTML, images, videos, LaTeX, and custom MIME types. Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, TensorFlow.



Chapter -3

EDA

3.1 Data collection:

Data was taken from kaggle website where you are given over 65,000 games' worth of anonymous player data, split into training and testing sets, and asked to predict final placement from final in-game stats and initial player ratings. Lets understand the data by looking into the variables

**DBNOs** - Number of enemy players knocked.

**assists** - Number of enemy players this player damaged that were killed by teammates.

**boosts** - Number of boost items used.

**damageDealt** - Total damage dealt. Note: Self inflicted damage is subtracted.

**headshotKills** - Number of enemy players killed with headshots.

**heals** - Number of healing items used.

**Id** - Player’s Id

**killPlace** - Ranking in match of number of enemy players killed.

**killPoints** - Kills-based external ranking of player. (Think of this as an Elo ranking where only kills matter.) If there is a value other than -1 in rankPoints, then any 0 in killPoints should be treated as a “None”.

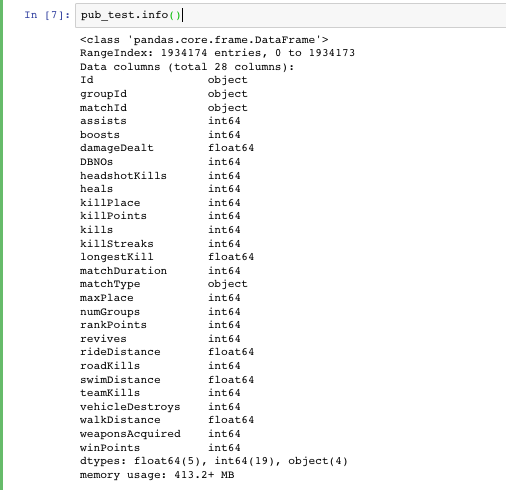
**killStreaks** - Max number of enemy players killed in a short amount of time.

**kills** - Number of enemy players killed.

l**ongestKill** - Longest distance between player and player killed at time of death. This may be misleading, as downing a player and driving away may lead to a large longestKill stat.

**matchDuration** - Duration of match in seconds.

**matchId** - ID to identify match. There are no matches that are in both the training and testing set.



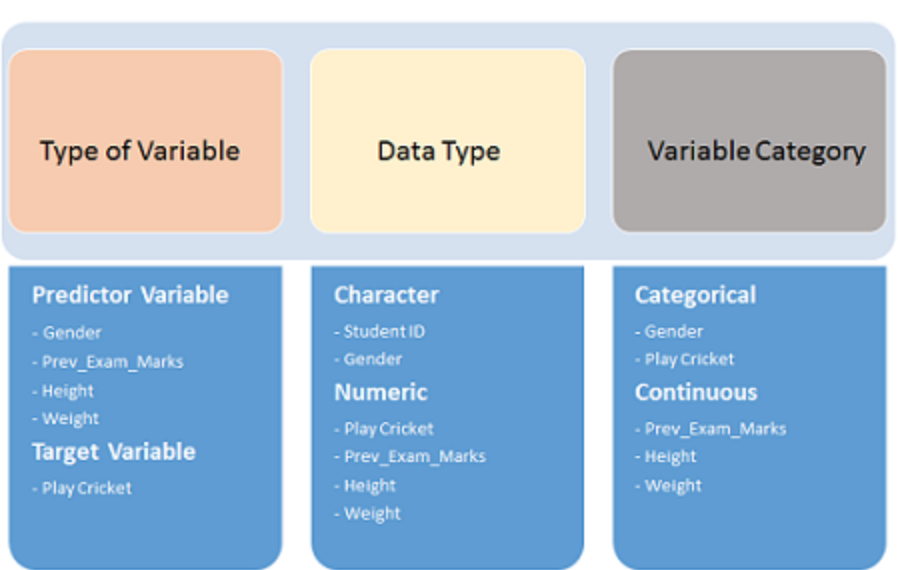
3.2 Variable identification:

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

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

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

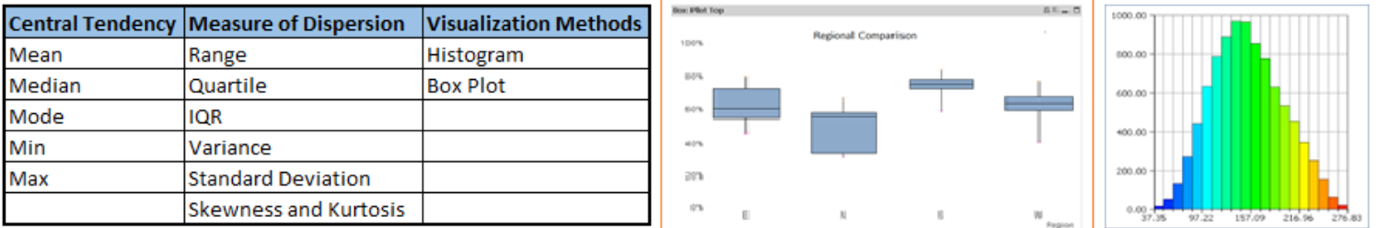
Below, the variables have been defined in different category:



3.3 Univariate analysis:

univariate data, which is a fancy way of saying **samples of one variable**—the kind of data that goes into a single list in python. Analysis of univariate data isn't concerned with the why questions—causes, relationships, or anything like that; the purpose of univariate analysis is simply to describe.

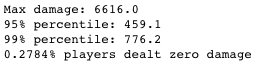
**Continuous Variables:-** In case of continuous variables, we need to understand the central tendency and spread of the variable. These are measured using various statistical metrics visualization methods as shown below:

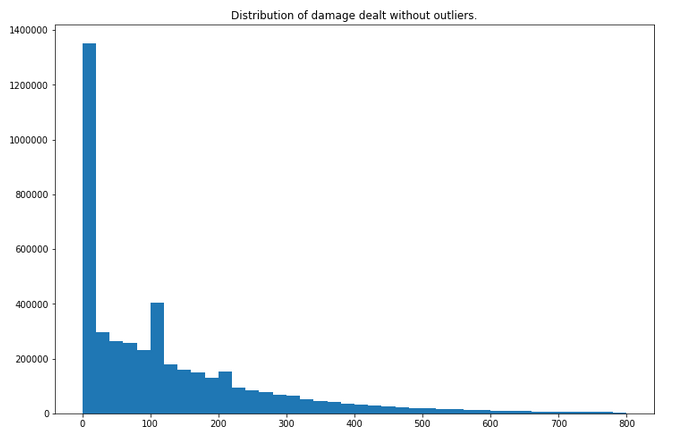


**Note:** Univariate analysis is also used to highlight missing and outlier values. In the upcoming part of this series, we will look at methods to handle missing and outlier values. To know more about these methods, you can refer course [descriptive statistics from Udacity](https://www.udacity.com/course/ud827" \t "_blank).

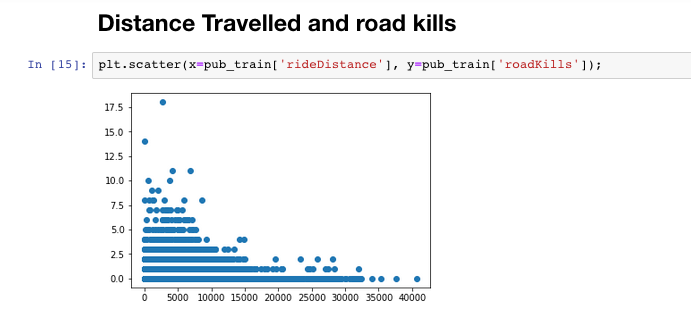
**Categorical Variables:-**For categorical variables, we’ll use frequency table to understand distribution of each category. We can also read as percentage of values under each category. It can be be measured using two metrics, **Count** and **Count%** against each category. Bar chart can be used as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

3.3.1 damageDealt column:

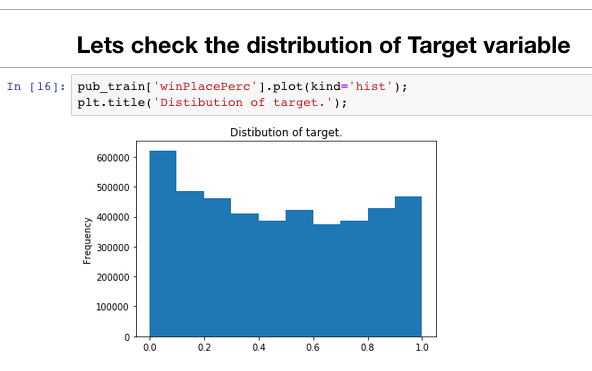




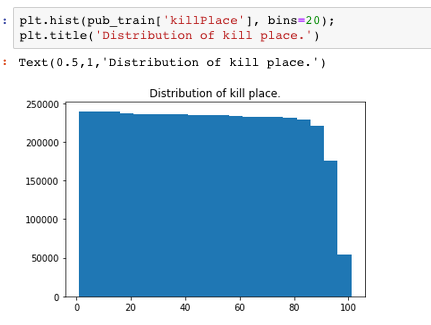
* + 1. Distance travelled and road kills



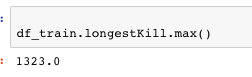
* + 1. Distribution Check on Target Variable



* + 1. Distribution of killPlace Column



* + 1. Univariate Analysis on kills
* Longestkill : In Meters

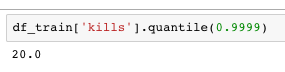




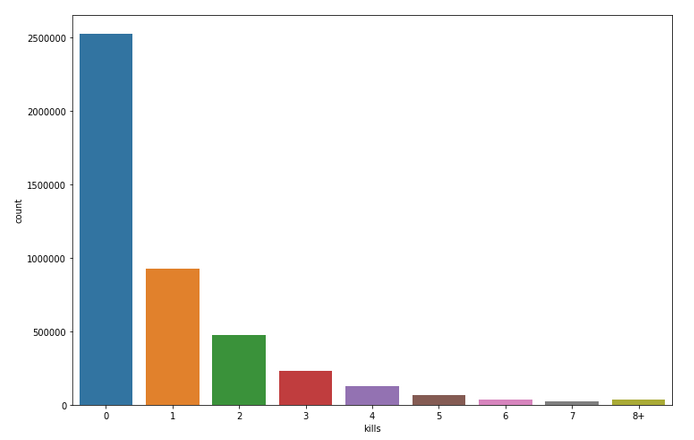
* To check percentile of kills i.e.

How many kills does 99th percent of players have ?

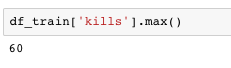
From below result its 20 or less.



* Kills count



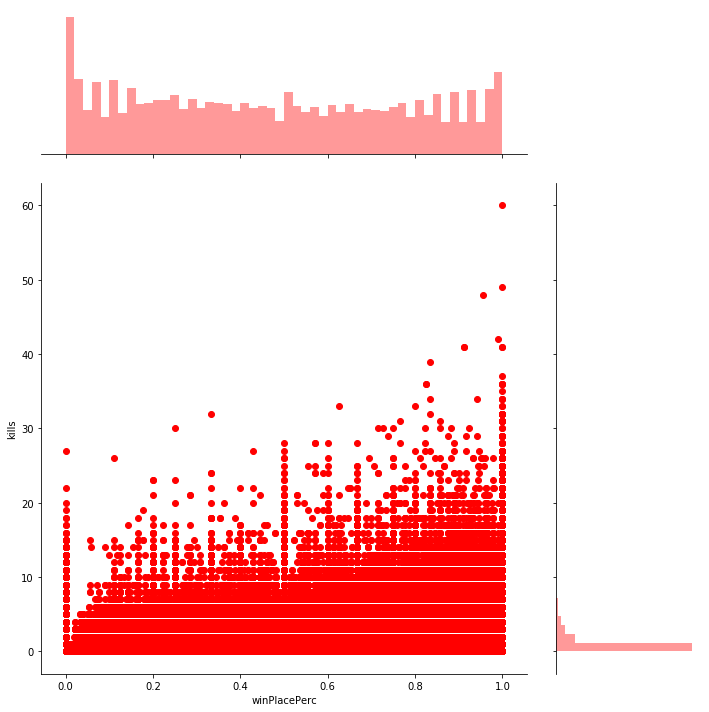
* Highest kill a Player secured in a Game



3.4 Bivariate analysis:

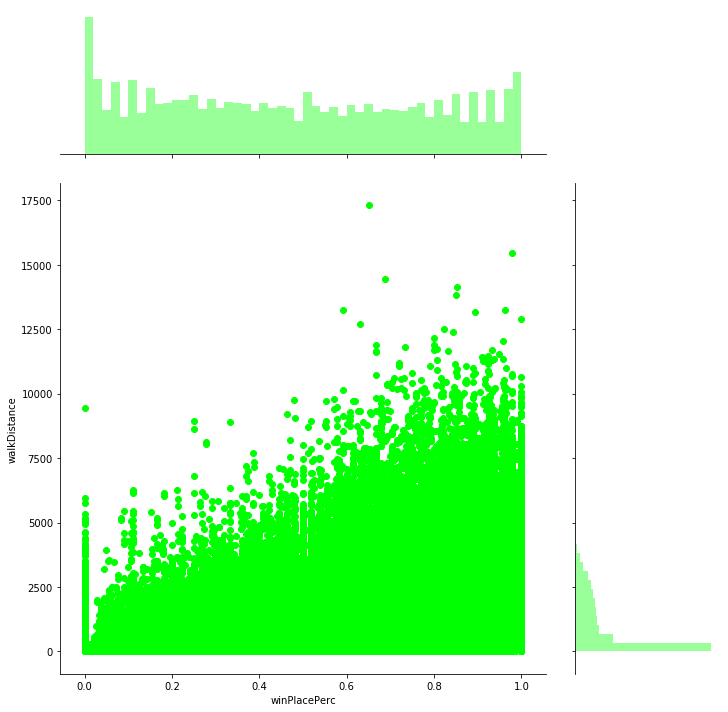
Bi-variate Analysis finds out the relationship between two variables. Here, we look for association and disassociation between variables at a pre-defined significance level. We can perform bi-variate analysis for any combination of categorical and continuous variables. The combination can be: Categorical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

* + 1. Correlation between kills and winPlacePerc column :



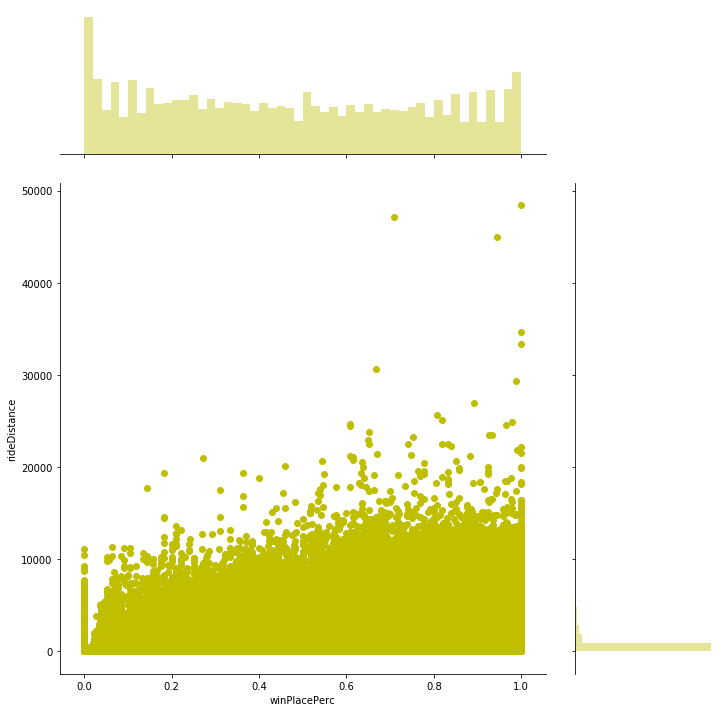
As we can see from above Kills have good correlation between Target variable i.e. As the kills increase the winPlacePerc column also increases.

* + 1. Correlation between walkDistance and winPlacePerc:



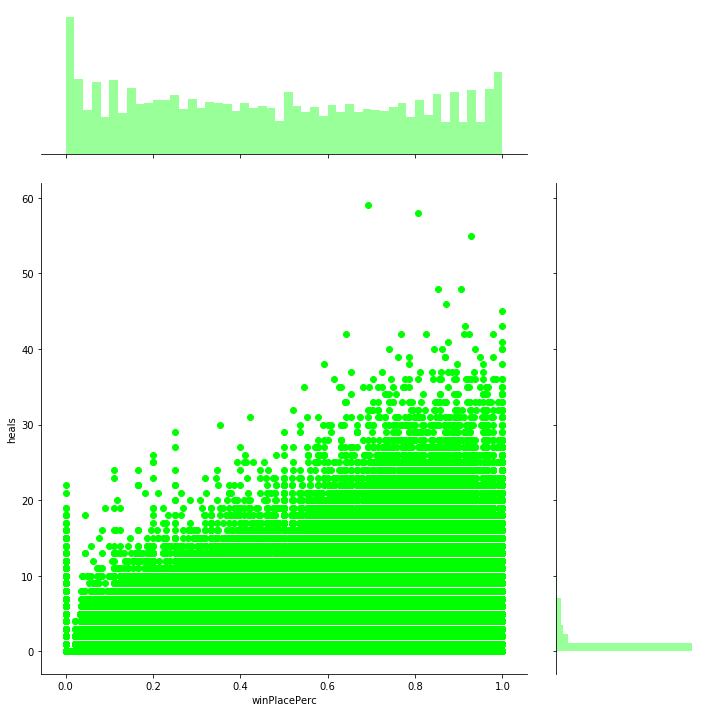
1. The average person walks for 1055.1m, 99% of people have walked 4138.0m or less, while the highest walked distance is for 17300.0m.
2. 94306 players (2.0581%) walked 0 meters. This means that they die before even taking a step or they are afk (more possible).
3. Walking has a high correlation with winPlacePerc.

3.4.3 Correlation between rideDistance and winPlacePercent

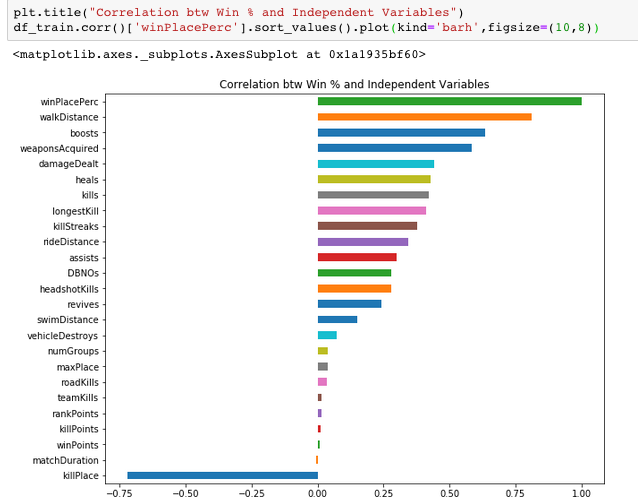


1. The average person drives for 423.9m, 99% of people have drove 6133.0m or less, while the formula 1 champion drove for 48390.0m.
2. 3439985 players (22.7940%) drove for 0 meters.
3. There is a small correlation between rideDistance and winPlacePerc

3.4.4 Correlation between heals and winPlacePerc



1. The average person uses 1.2 heal items, 99% of people use 11.0 or less, while the doctor used 59.
2. The average person uses 1.0 boost items, 99% of people use 7.0 or less, while the doctor used 18.
3. Healing and boosting, definitely are correlated with winPlacePerc. Boosting is more.
   * 1. Similarly Correlation between winPlacePerc and Independent variables



3.5 Impact of Outliers on Dataset

Outlier is an observation that appears far away and diverges from an overall pattern in a sample. Outliers can drastically change the results of the data analysis and statistical modeling. There are numerous unfavorable impacts of outliers in the data set:

• It increases the error variance and reduces the power of statistical tests

• If the outliers are non-randomly distributed, they can decrease normality

• They can bias or influence estimates that may be of substantive interest

• They can also impact the basic assumption of Regression, ANOVA and other statistical model assumptions.

## Missing Value Treatment

## Why missing values treatment is required?

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

3.6.1 My Dataset:

No NA values found in this dataset.

Chapter- 4

Feature Engineering

4.1 Introduction:

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process. Feature Engineering is an art.

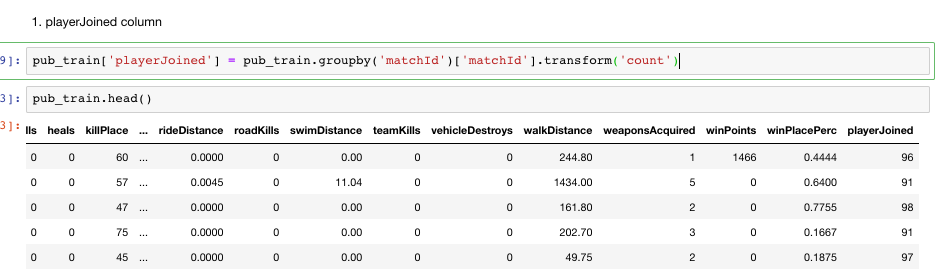
Steps which are involved while solving any problem in machine learning are as follows:

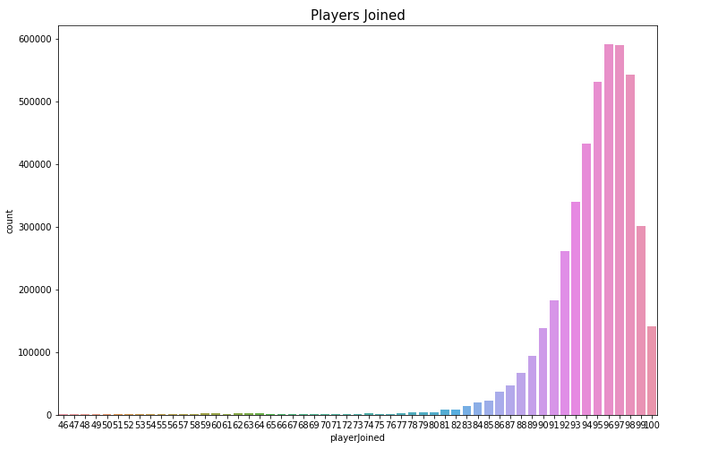
* Gathering data.
* Cleaning data.
* Feature engineering.
* Defining model.
* Training, testing model and predicting the output.

Feature engineering is the most important art in machine learning which creates the huge difference between a good model and a bad model. Let’s see what feature engineering covers.

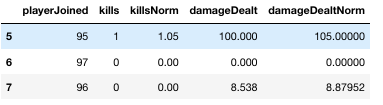
4.2 Creating new features:

4.2.1 playerJoined feature

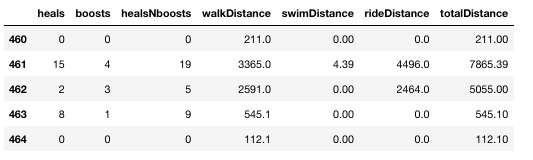




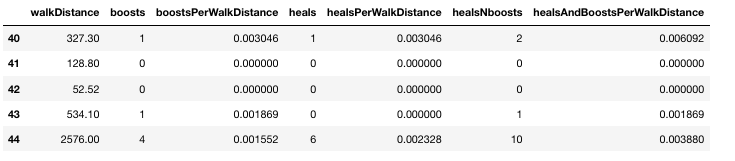
Based on the "playersJoined" feature we can create (or change) a lot of others to normalize their values. For example i will create the "killsNorm" and "damageDealtNorm" features. When there are 100 players in the game it might be easier to find and kill someone, than when there are 90 players. So i will normalize the kills in a way that a kill in 100 players will score 1 (as it is) and in 90 players it will score (100-90)/100 + 1 = 1.1. This is just an assumption. You can use different scales.



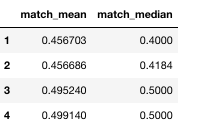
4.2.2 To sum up heals and boosts and make 'totalDistance' as one feature.



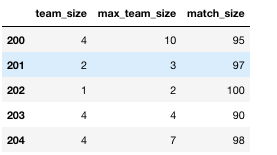
When using boosting items you run faster. They also help staying out of the zone (PUBG term) and loot more (meaning walking more). So lets create a feature boosts per walking distance. Heals don't make you run faster, but they also help staying out of the zone and loot more. So lets create the same feature for heals also.



4.2.3 To create match\_mean and match\_median feature



Team size and match sizes are not given metrics but they can be easily derived from the groupid and matchid.



4.2.4 List of new features added

* Team\_indicator
* Game\_mode
* Max\_possible\_kills
* Total\_items\_acquired
* Items\_per\_distance
* Kills\_per\_distance
* Knocked\_per\_distance
* Damage\_per\_distance
* Headshot\_kill\_rate
* Max\_kills\_by\_team
* Total\_team\_damage
* Total\_team\_kills
* Total\_team\_items
* Perc\_killed
* Perc\_knocked
* Perc\_team\_killed
* Team\_kill\_points
* Team\_kill\_rank
* Max\_kills\_match
* Total\_kills\_match
* Total\_distance\_match
* Map\_has\_sea

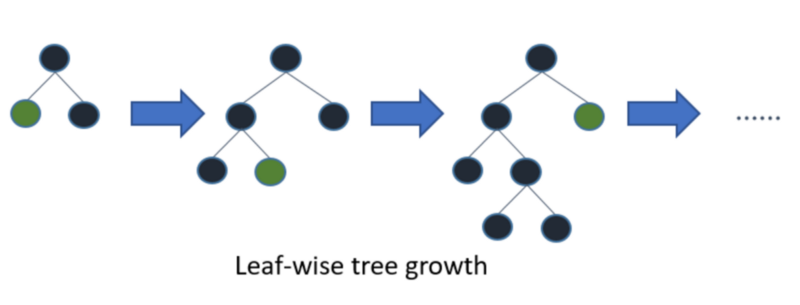
Chapter- 5

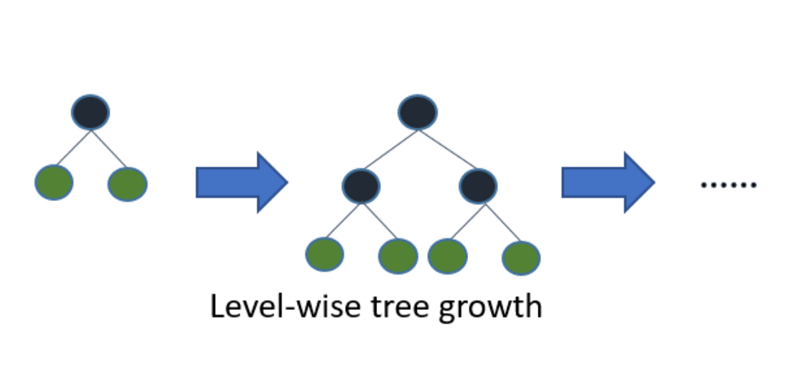
Model Building

* 1. Introduction to Light GBM

Light GBM is a gradient boosting framework that uses tree based learning algorithm. Light GBM grows tree vertically while other algorithm grows trees horizontally meaning that Light GBM grows tree leaf-wise while other algorithm grows level-wise. It will choose the leaf with max delta loss to grow. When growing the same leaf, Leaf-wise algorithm can reduce more loss than a level-wise algorithm.

5.2 Understanding of Light GBM Algorithm





When do we use Light GBM ?

The size of data is increasing day by day and it is becoming difficult for traditional data science algorithms to give faster results. Light GBM is prefixed as ‘Light’ because of its high speed. Light GBM can handle the large size of data and takes lower memory to run. Another reason of why Light GBM is popular is because it focuses on accuracy of results. LGBM also supports GPU learning and thus data scientists are widely using LGBM for data science application development.

When do we not use Light GBM ?

It is not advisable to use LGBM on small datasets. Light GBM is sensitive to overfitting and can easily overfit small data. Their is no threshold on the number of rows but my experience suggests me to use it only for data with 10,000+ rows.

5.3 Implementation of Light GBM

Implementation of Light GBM is easy, the only complicated thing is parameter tuning. Light GBM covers more than 100 parameters

It is very important for an implementer to know atleast some basic parameters of Light GBM. If you carefully go through following parameters of LGBM, I bet you will find this powerful algorithm a piece of cake.

5.4 Parameters of Light GBM Algorithm

Control Parameters

max\_depth: It describes the maximum depth of tree. This parameter is used to handle model overfitting. Any time you feel that your model is overfitted, my first advice will be to lower max\_depth.

min\_data\_in\_leaf: It is the minimum number of the records a leaf may have. The default value is 20, optimum value. It is also used to deal over fitting

feature\_fraction: Used when your boosting(discussed later) is random forest. 0.8 feature fraction means LightGBM will select 80% of parameters randomly in each iteration for building trees.

bagging\_fraction: specifies the fraction of data to be used for each iteration and is generally used to speed up the training and avoid overfitting.

early\_stopping\_round: This parameter can help you speed up your analysis. Model will stop training if one metric of one validation data doesn’t improve in last early\_stopping\_round rounds. This will reduce excessive iterations.

lambda: lambda specifies regularization. Typical value ranges from 0 to 1.

min\_gain\_to\_split: This parameter will describe the minimum gain to make a split. It can used to control number of useful splits in tree.

max\_cat\_group: When the number of category is large, finding the split point on it is easily over-fitting. So LightGBM merges them into ‘max\_cat\_group’ groups, and finds the split points on the group boundaries, default:64

Core Parameters

Task: It specifies the task you want to perform on data. It may be either train or predict.

application: This is the most important parameter and specifies the application of your model, whether it is a regression problem or classification problem. LightGBM will by default consider model as a regression model.

* regression: for regression
* binary: for binary classification
* multiclass: for multiclass classification problem

boosting: defines the type of algorithm you want to run, default=gdbt

* gbdt: traditional Gradient Boosting Decision Tree
* rf: random forest
* dart: Dropouts meet Multiple Additive Regression Trees
* goss: Gradient-based One-Side Sampling

num\_boost\_round: Number of boosting iterations, typically 100+

learning\_rate: This determines the impact of each tree on the final outcome. GBM works by starting with an initial estimate which is updated using the output of each tree. The learning parameter controls the magnitude of this change in the estimates. Typical values: 0.1, 0.001, 0.003…

num\_leaves: number of leaves in full tree, default: 31

device: default: cpu, can also pass gpu

Metric parameter

metric: again one of the important parameter as it specifies loss for model building. Below are few general losses for regression and classification.

* mae: mean absolute error
* mse: mean squared error
* binary\_logloss: loss for binary classification
* multi\_logloss: loss for multi classification

IO parameter

max\_bin: it denotes the maximum number of bin that feature value will bucket in.

categorical\_feature: It denotes the index of categorical features. If categorical\_features=0,1,2 then column 0, column 1 and column 2 are categorical variables.

ignore\_column: same as categorical\_features just instead of considering specific columns as categorical, it will completely ignore them.

save\_binary: If you are really dealing with the memory size of your data file then specify this parameter as ‘True’. Specifying parameter true will save the dataset to binary file, this binary file will speed your data reading time for the next time.

Chapter- 6

Conclusion

6.1 Summary of the project outcome:

Using exploratory data analysis, we mainly did variable identification in which we identify predictor (Input) and target (output) variables. Next step was to identify the data type and category of the variables. Then we moved to univariate analysis. At this stage, we explored variables one by one. Method to perform uni-variate analysis will depend on whether the variable type is categorical or continuous. After this we did bi-variate analysis. Bi-variate Analysis finds out the relationship between two variables. Here, we look for association and disassociation between variables at a pre-defined significance level. We can perform bi-variate analysis for any combination of categorical and continuous variables. The combination can be: Categorical & Categorical, Categorical & Continuous and Continuous & Continuous. Different methods were used to tackle these combinations during analysis process. Then we did missing value treatment, fortunately this training set did not have any missing value.so basically missing data in the training data set can reduce the power / fit of a model or can lead to a biased model because we have not analysed the behaviour and relationship with other variables correctly. It can lead to wrong prediction or classification. And lastly, we did outlier detection. Basically, outlier is an observation that appears far away and diverges from an overall pattern in a sample. In feature engineering we came up with around 25+ new features from which we found many of them as the key feature for this dataset. Finally for Modelling purpose we used Light GBM since our data is huge with 4million rows and 55+ columns and was a regression problem with Mean Absolute Error (mae) as its measuring metrics. Finally prediction for Winning Placement Percentage is done using LGBMRegressor Algorithm from sklearn.

Chapter – 7

Future Scope

7.1 Future Scope:

The next stage of this project is to apply different machine learning algorithm to build up a model which can predict with lesser Mean Absolute Error(mae). I will be using Multi-layer Perceptron as a demo to check for better prediction. I will also be testing with hyperparameter in LGBM so that my prediction will be better than what I obtained so far.

References

* <https://www.kaggle.com/c/pubg-finish-placement-prediction>
* <https://medium.com/mindorks/what-is-feature-engineering-for-machine-learning-d8ba3158d97a>
* <https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc>
* <https://en.wikipedia.org/wiki/PlayerUnknown%27s_Battlegrounds>