**DOCUMENTATION**

**FINDINGS DOCUMENTED FOR STAKEHOLDERS:**

The aim of the project is to utilize the 'Starcraft' player performance dataset, analyze the dataset that is perform Exploratory data analysis, model it is using various machine learning models, evaluate the results. To study the dataset, I have first looked at its shape that is how many rows and columns the dataset is having, the dataset initially was having 3395 rows and 20 columns, further I have checked the structure of the dataset that is the data type of each column, summary of the dataset. I have also looked at the first 10 values of the dataset to check how the data has been classified. I wanted to see the minimum, maximum, mean, median etc., statistics of each column, so I have generated the description of the dataset.

I have started performing exploratory data analysis on the dataset, this involved examining the dataset, looking for correlations, looking if there are any null values in the dataset, visualize the dataset. First step, I have checked if there are any null values in the dataset, the result was that there were no null values, then I have checked for the correlation among the target variable and feature variables. All the features are having descent correlation with each other, but to compare 'GameId', 'UniqueUnitsMade', 'ComplexAbilitiesUsed' and 'ComplexUnitsMade'. But it is not advised to remove anything related to Units as it gives us the data regarding the player's performance, considering this 'GameId', 'ComplexAbilitiesUsed' variables can be removed safely. This means that the variables are either positively correlated or negatively correlated.

I have also performed visualization on the dataset, when plotted a bar plot for 'LeagueIndex' and 'UniqueHotkeys', every index of league index has all types of unique hot keys. I have also plotted a scatter plot for 'APM' and 'LeagueIndex', when the league index is the highest, the action per minute is the highest, we can observe a positive correlation between two variables.

After performing EDA, next step is to identify any categorical variables and encode them. We can see there are no categorical variables, so no need to encode the variables. But we need to scale the variables as all the variables are not in equal range, so I have applied 'z-score normalization' to the dataset where the mean of each column is equal to zero, and the standard deviation is equal to one. After scaling, all the values are in equal range and the dataset is ready to be applied on machine learning models.

We have to predict the player's rank using the above given information, that means the target variable is 'LeagueIndex' which is ordinal in nature that means it has many more than two unique values.To predict the target variables which are ordinal in nature, I have used first method ‘**SVM’** which is support vector machine.

Based on the results after performing SVM model on the dataset to predict the player's rank, svm model is not performing significantly better than random chance. The train accuracy of 41% indicates that the model can predict the ordinal values correctly for 41% of the training instances, while the test accuracy of 37.2% suggests that the model's performance on unseen data is slightly worse.

I have then chosen Random Forest Classifier model to apply on the dataset as the dataset we have is having many columns and is complex to capture the relationships among the variables, in such cases Random Forest model can outperform SVM.

Applying Random Forest classifier to our dataset, the Random Forest model is overfitting the training data, resulting in a high training accuracy but a significantly lower test accuracy. The large discrepancy between the training accuracy (99.9% to 100%) and test accuracy (36.3% to 34.9%) suggests that the model is memorizing the training data rather than generalizing well to unseen data. I have increased the number of estimators (from 25 to 100) but it did not improve the test accuracy significantly. This indicates that adding more trees alone was not sufficient to address the overfitting issue. Before setting the maximum depth to 10, I have adjusted the dataset by removing 'SelectByHotkeys' and 'ActionLatency' as it might perform better by having less relations, then I have set the maximum depth to 10 which resulted in a slight improvement in the test accuracy (34.9% from 36.3%). By limiting the depth of the individual trees, I am trying to effectively control the complexity of the model and prevent it from overfitting the training data.

Next step I followed was to use XGBoost model on the dataset, as the random forest model continued to struggle with the overfitting problem.

The XGBoost model also has given the same results as Random Forest where the model has overfitted, the high training accuracy (99.9% to 62%) and low-test accuracy (35% to 38%) indicate that the XGBoost model is overfitting the training data. I have reduced the maximum depth to 3, By reducing the max depth to 3, I have constrained the complexity of the individual trees in the XGBoost model. This regularization technique helps control overfitting and can lead to better generalization on unseen data, but it did not give the best results.

I believe the problem of overfitting in the dataset is because there is insufficient training data, when the size of the training dataset is small relative to the complexity of the problem, the model may learn to memorize the training examples instead of understanding the underlying patterns. As a result, this is failing to generalize the new data. There could be another reason that the distribution of the classes is imbalanced.

**HYPOTHETICAL QUESTION ANSWERED:**

If the stakeholders want to collect more data, and seeking for advises, the following points must be considered:

* The stakeholders must consider the data they collect is not having null or duplicate values.
* The main issue in the existing dataset is that there is improper distribution of classes, so the stakeholders must keep in their mind that the new data collected must represent different classes of distribution to help the model to learn and generalize.
* The stakeholders must consider the importance of features to train the models.
* Collect the data such that it is cost-effective and feasible.

The above points are crucial for the stakeholders to gather more data.