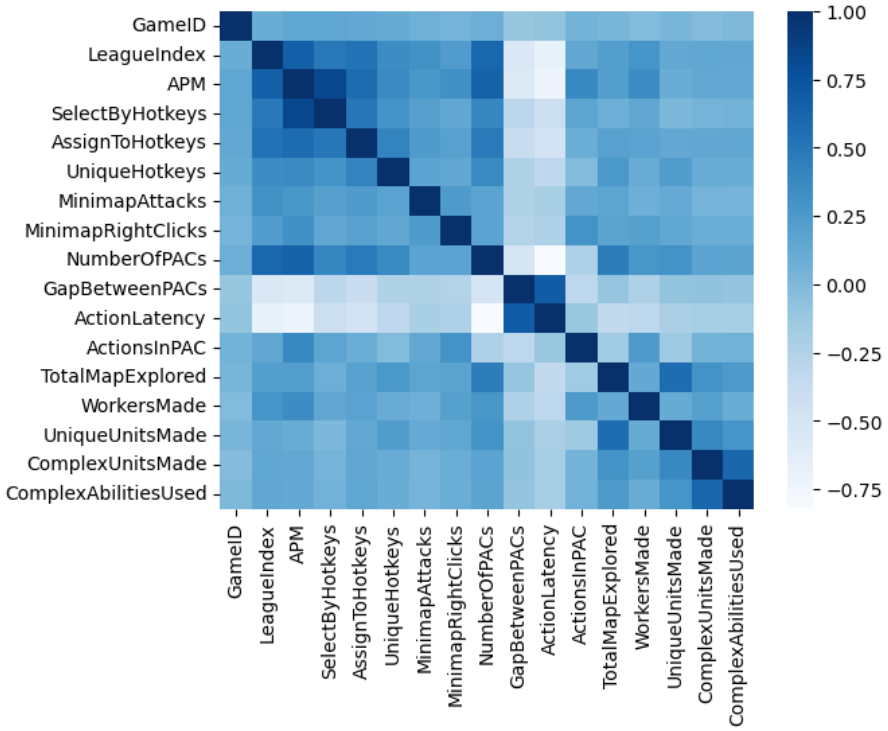
Evil Geniuses Data Science Intern Assessment

Aidan Admadjaja

May 2023

1 Exploring the Data

Before beginning to construct any machine learning models, I first examined the data in pandas and looked at the data dictionary to see what each feature entailed. After that, I made a heat map showing the correlation between the different features of the data. This heat map was made primarily to get a sense of while features were correlated, in particular while features were correlated with LeagueIndex. This is because LeagueIndex is the feature that we are trying to predict.



Looking at the heatmap above, we can see that different features have different

correlations with LeagueIndex. For example, APM and NumberofPACs are strongly correlated with the LeagueIndex. The stronger the players are, the more actions per minute that they should be able to make. This is most likely due to a combination of better game knowledge as well as high mechanical skill. Also, the higher strength players will have also have more attention shifts and actions done. This is because there is always something to do to improve your starcraft army and one must be able to micromanage different units to attack and mine for economy. Conversely, a stronger player should have less action latency since they will know what to do when their attention shifts to a new PAC. The same would be true for ActionsInPAC, where stronger level players would have less actions in each PAC before shifting their attention to micromanage other units.

2 Data Cleaning

I started by removing all NaN values, but there were none. However, I noticed that there were ?'s in the data, so I decided to remove them and then try training and testing some models. I then realized that there were only 7 different classes of LeagueIndex, therefore, I went back to check the data and see that all the players with LeagueIndex 8 were removed. Pulling up only the data points with LeagueIndex 8, shows that all of them are missing the Age, HoursPerWeek and TotalHours columns. So, I decided to just remove those 3 specific columns from the entire dataset.

3 Models

This problem is a classification problem, meaning that I am given data, and using that data, I am trying to classify this person into a group. Specifically, it is a multinomial classification problem since there are 8 different possible classes to identify players as. The two models that I tested for this problem were logistic regression and neural networks. For both, I did grid search to try and find the best hyperparameters for the models. I tested logistic regression and neural networks in 2 different scenarios. The first scenario was with all the features trying to predict LeagueIndex, and the second scenario was just using APM, NumberOfPACs, ActionLatency and ActionsInPAC (most correlated features to LeagueIndex) to predict LeagueIndex.

3.1 Tuning Logistic Regression

The hyperparameters that I tuned in the logistic regression models were C, penalty and max_iter. C is used to help prevent overfitting as it is the inverse of the regularization strength. Regularization adds a penalty term to the loss function which penalizes high weights. Finding the correct C is important to find the balance between the complexity of the model and preventing it from overfitting. Penalty is just the type of regularization that is used in the model. l1

is Lasso and l2 is Ridge. Picking the correct type of regularization is important for model performance as each type of regularization produces different results. Lastly, I tuned the maximum number of iterations. I included this in the grid search because I was getting warnings regarding the model not convering using the default number of iterations (100). Therefore, I increased the number of possible iterations in case it improved the performance of the model.

3.2 Tuning Neural Network

The hyperparameters that I tuned in the neural networks models were layer size, dropout rate and learning rate. Layer size refers to the number of nodes in the hidden layer; more nodes can lead to more complex representations of relationships between input and output features, but can also lead to overfitting. Finding the optimal layer size is important for not having too complex of a model which can lead to overfitting. Learning Rate tells the model how much the weights of features are updated given the errors in training. I tuned the learning rate to help with the efficiency of model learning - if too low, the model can learn too slow, but if too high, the model can overshoot an optimal solution. Dropout rate is key for reducing overfitting as it will randomly drop out which adds a penalty to the loss function based on the magnitude of the feature weights. Dropout rate was tuned to help balance the complexity of the model to try and reduce overfitting.

4 Evaluation

I used mean squared error as an evaluation metric for my models for multiple reasons. Firstly, it is quite simple to understand mathematically as it is just the squared differences between the ground truth and predicted values averaged out. Secondly, it can be used to compare different models as you can just find the model with the minimum mean squared error. This can be useful for tuning hyperparameter as well as comparing different types of models (in this case, logistic regression and neural networks).

4.1 Output and Findings

Model	Hyperparameters	Mean Squared Error
Logistic Regression All Features	C = 31.6, penalty = "l2",max_iter = 1000	1.408
Logistic Regression 4 Features	C = 10, penalty = "l2", max_iter = 1000	1.224
Neural Network All Features	LS: (128,), DR: 0.3, LR: 0.01	1.5712
Neural Network 4 Features	LS: (32,), DR: 0.2, LR: 0.01	1.7198

In this table, LS is Layer Size, DR is dropout rate, LR is learning rate. As seen here, the logistic regression did better than the neural network. I was slightly surprised by this as one of the strengths of neural networks are the fact that they

can derive complex, non-linear relationships between features well. I thought that with all features in the data, the neural network would do well and find some complex relationships between the features to help predict LeagueIndex. However, this was not the case. I think this is due to the fact that there were not that many data points, only around 3400 or so. Neural networks need many data points to properly train and learn these different relationships.

5 Additional Data Collection

For one, it would be interesting to see the Age, HoursPerWeek and TotalHours columns for LeagueIndex 8 players. Adding this data would allow us to run the models with 3 more features, as well as get a more complete picture in the heat map, seeing whether or not these 3 features are correlated or not with LeagueIndex. In particular, to me, age would be interesting, since it is known that e-sports athletes typically have a prime around their early to mid twenties. So age would be interesting, since there are definitely outliers where some older players can stick around the top, as well as some young prodigies that rise through the ranks quickly. Overall, taking an educated guess, I would say that age would not be well correlated with LeagueIndex. The second thing would just be to collect more data relating to APM, NumberOfPACs, ActionLatency and ActionsInPAC as these are the features that are most strongly correlated (positively and negatively) with LeagueIndex. Having more data points will allow the neural network to better train itself and ultimately be more accurate.