Introduction

As the world becomes more digitalized, and interests move further towards expression over technology, this trend is seen especially in the growing field of Esports. First established in the early 2000's, the competitive interests in video games has seen a steady growth, however recently with the development of Twitch, Eleagues, and corporate sponsors, the viewership and participants have seen an exponential growth in the past few years.

With that said, the tools that the players can use in their respective fields is still a largely untapped market. Where you still massive databases in professional sports industries, which help create tools for spectators to view the game differently, and for the participants to learn from their faults, these data analysis tools are absent in the world of Esports.

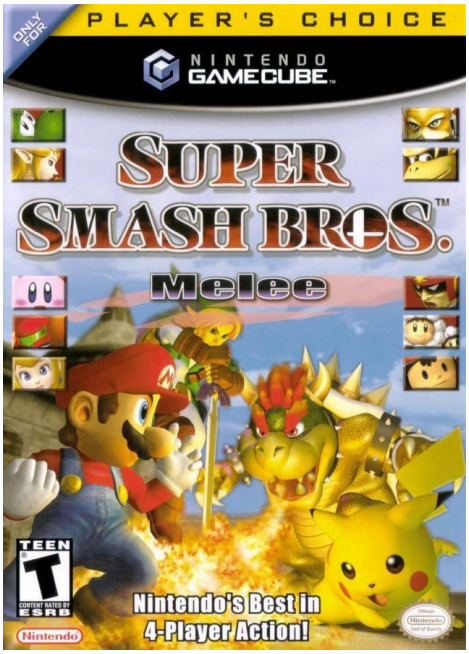
This absence is the result of multiple issues. First, simply in the environment that players are competing. When spectating a tennis match, it is clear to see body movement, or shot placement percentages, and clear to recognize the small victories with each successive point. In the eSports industry, the competitive measures the lead to success are mostly mental, and do not rely as heavily on physical movements. What is physical, however, is a collection of very quick, sharp, calculated movements that is not physically viewed by the spectator. These are movements like mouse clicks, controller stick movements, and well-timed keyboard touches. When going about calculating these data, this presents a massive challenge for people interested in data analysis.

The next issue comes in defining a clear success in most of the games. While competing, it is true that one team will result in a win, and the other a loss. However, the thousands of interactions that lead up to the result do not have the same clear successor. Often, interactions between two players does not result in a clear victory, and results in a simple advantage state, or one player improving a position over another. Defining these interactions can be hard to define in most games, and therefore it is hard to get value out of the data collected.

Lastly, and perhaps the biggest issue, is a lack of understanding from the public of how to approach the field of eSports. This issue is one that is changing for the better, however society is still at a point where many cannot view eSports in the same way that traditional sport culture is viewed. Unfortunately, this concept extends to the developers of the games, who do not include tools for competitive players to learn, grow, and adapt with the current meta. This is a result of the developers not intending to market their games to only competitive players, and these leaves the development of new tools to the communities themselves. To compare to traditional sports, this would be the equivalent of NBA players to create the tools to record shot percentage,

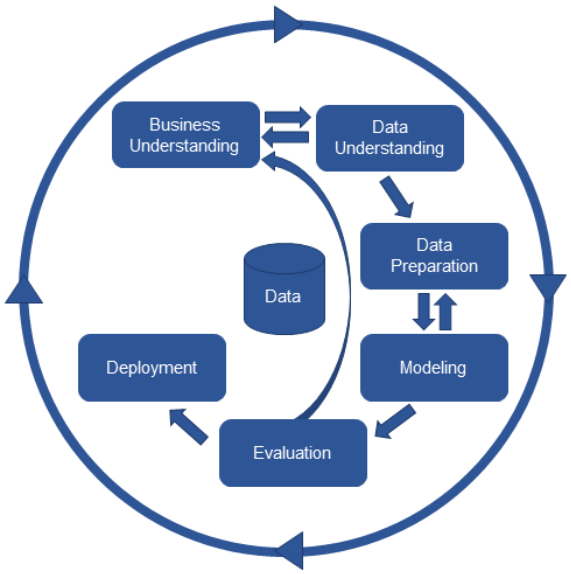
rebounds, assists, and matchup trends rather than an association that would hire specialists in data collection and science to record and maintain this information. Therefore, the process is long, strenuous, and relies on the passion of the community rather than any nominal gain.

With this said, this led to the data project that was conducted, with the hopes to develop a structure of data analysis within the field of Esports. The specific Esport of interest for this study is competitive Super Smash Brothers Melee, which was developed in 2001. Since then, the competitive community has hosted thousands of major tournaments, and has produced a total winnings of around 2.8 million dollars. This study will look at the gameplay details gathered from one tournament and hope to find the factors in an individual's gameplay that leads to overall success.



Methodology

The data project was conducted along a slightly simplified version of CRISP-DM. The study is separated into 5 phases: 1) Data Collection; 2) Data Understanding 3) Data Preparation; 4) Modeling; 5) Evaluation.



Data Collection

This led to the work down by Fizzi in the creation of Project Slippi, the main tool used for data collection in this analysis. Project Slippi is described as a Melee Data Framework, and essentially is a tool that offers every player the tools available to begin data analysis on the matches they play. This framework was implemented at a tournament in Philadelphia under the name, The Game Steals the Script. The software was installed on 25 consoles, and after each game was played on these marked setups, the .slp was uploaded to slippi.gg, where the data and stats could then be viewed. A week

later, on March 15th, Project Slippi then released a JSON dump of all the .slp files in their database, so data analysis could then be done. For this project, the final data set was constructed from this JSON metadata dump.

Data Understanding

This form of data collection, because it was implemented with code, came out to be extremely objective, as every file was consistent, and could not be tampered with by any individuals. Amongst the data, there were no missing columns, and only had NA values for when ratios were calculated with 0 in the denominator.

Therefore, the quality of the data is not measured in terms of how smoothly gathering the data goes or by any bias, but rather how smoothly the tournament ran, and the players that were in attendance. For example, if the tournament had multiple games where players were often not playing games competitively to completion, this has the ability to skew the data to show results that do not lead to a true depiction of killCount. Another measure of quality comes from the skill level of the players attending. If the tournament has a majority of players who often perform well, this will produce more consistent results with proper depiction of killCount than one that has a majority of players who often do not perform well. Lastly, another relevant measure of quality comes from the character representation throughout the tournament. Although there are 26 characters, only about 10 are often used competitively. However, it is important that these characters all have proper representation, as the features that used for the analysis are highly dependent on the character that is being played.

Having addressed how quality can be measured, this tournament had good data quality. Of the 204 entries in the bracket, about 15 of those were in the top 100 best players for the year, which is good representation for a tournament. In addition, about 100 of the other players fit the tier below top level, and the rest fill the tiers of players who are not often successful in a tournament setting. In regard to the character diversity, this also appeared to be successful. The results show that the top 10 viable characters had representation of between 2000-2800 games each and was evenly distributed. The remaining characters ranged from 99 games to 1500 games, which is a fair representation for how often players are chosen in competitive play. The last measure of quality is in the number of games that were played to a competitive level to its completion, however this is simply a variable that cannot be calculated by the data received from the .slp file. Rather this is a subjective measurement that can only be gathered by in person spectators, who can properly witness multiple the interactions between players at the tournament. Therefore, this study cannot speak on the quality of all games that were

played, however it is safe to assume most were competitive, as a large majority of the games recorded were tournament sets, which are by nature treated with seriousness between both competitors.

In conclusion, the quality of this data is unbiased, consistent, and does not content very many missing data values. By tournament standards, The Gang Steals the Script showed proper player and character representation, with a good mixture of top and mid-level talent, as well as a proper character distribution that would be expected in a competitive melee tournament.

Data Preparation

For this project, the final goal was to get the needed data from the JSON format to a CSV format, that way it would be easier to import into RStudio. Not only did the format need changed, but the data supplied in the JSON dump had to be reformatted and cleaned up. After the first download, the first step was to look at an individual .slp file, which was the metadata for an individual game, and see what was necessary to keep and what was not. Each .slp file was formatted into multiple documents, including root, stocks, conversions, moves, action counts, overall, successful conversions, inputs per minute, openings per kill, damage per opening, neutral win ratio, counter hit ratio, and beneficial trade ratio. For the desired analysis that the project was interested in, much of this data was either unhelpful or redundant. Therefore, after examining the data, it was deemed that the only documents that were worth extracting variables out of were root, action counts, and overall.

From this step, the next choice was in discovering what variables would be needed from those three documents. For this analysis, there wasn't a heavy interest in specific combo data, but rather character, stage, movement, and conversion data were the focus of this analysis. Therefore, below show the following data that was extracted from each JSON document.

Document	**Variables Extracted**
Root	characterId stageId
ActionCount	wavedashCount wavelandCount airDodgeCount dashDanceCount spotDodgeCount rollCount
Overall	inputCount conversionCount totalDamage killCount inputsPerMin openingsPerKill_count openingsPerKill_ratio damagePerOpening_ratio neutralWinRatio_count neutralWinRatio_ratio counterHitRatio_count counterHitRatio_ratio beneficialTradeRatio_count

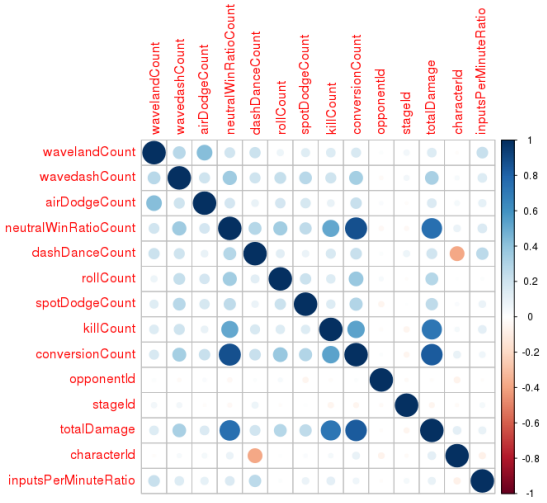
Once this list was created, the data was then extracted into a flat file from a function used in Mongo, and then transferred into a CSV file, which was then prepared to be manipulated. One major issue that had to be accounted for was in the format of the .slp file, that had two players for each row. This is a result of the nature of a game, which is the competition between two individuals. Therefore, when the data is

transferred over, the information for player one and player two are intertwined in the same row. Therefore, the next step in the data manipulation was to eliminate this trend and create unique rows for each player from each game. This process involved writing code in Java to read the current CSV file, extract the data, then create two separate arrays for player one data and player two data.

After the two arrays were constructed, and in the proper order in relation to one another, they were then printed to the file slippiData.csv, which then almost held the final ABT that was used. A note from the code above was in the importance of the construction of the arrays, to be sure that the rows where in the same order as its corresponding character id pair. In other words, the decision to grab the rows in the order I did for each was crucial, and if this step was done unproperly, then some rows would have been with the wrong data, completely invalidating the data set. Another note was that a new data row was constructed from this set, in the form of an opponentId. Essentially, it was important in the analysis to know who the character that the player was fighting against, therefore, the opponentId was added to the set of features.

Feature Selection

At this point, there 22 features for each player, and this is where the feature deletion process really began. To start, there were a few feature selection strategies to consider, with the goal to identify what features were no longer necessary. First, the process began with running multiple correlation plots of the data. The features damagePerOpeningRatio, openingsPerKillRatio, and counterHitRatio had to be removed, as they all included "NA" values that were not suitable for the correlation model to be run. The correlation plots did not show any two variables that were too strongly related to be removed. Among the stronger relationships, there was conversionCount and neutralWinRatioCount, totalDamage and neutralWinRatioCount, totalDamage and conversionCount, and totalDamage and the target, killCount. However, although these variables were decently correlated, each data point is important to the analysis, so they were all kept in for this study.



Next, a random forest model was run to see the attribute importance. The most important far and away was openingsPerKillRatio, and the least important tended to show stageId, opponentId, and spotDodgeCount to be among the least important in distinguishing killCount. However, for the final model, these variables were still considered. Continuing forward, feature selection next focused on entropy values to remove features. However, all features were

somewhere in between an entropy of 8.0-8.7, therefore this process also did not remove any features. After this, prediction tree models, forward greedy with a tree evaluation, and forward greedy with a linear evaluation we're all performed. However, these results were indecisive, and did not lead to the removal of any features.

This concludes the feature selection process, and the last step towards going through the data is simply removing features that intuitively seem unhelpful as a competitive player towards determining a killCount. From here an additional 6 features were removed to create the final ABT. First, inputsPerMinuteTotal was removed, as the values were confusing and misleading, with especially unclear units. Next, id was removed, as the values were not helpful, and they were causing issues with the model creation. Last, the remaining four variables that were deleted were total counts, which proved to be unnecessary due to the existence of their ratios, which is a more helpful unit of measure for this type of information. These four variables were inputCount, openingPerKillCount, inputesPerMinuteCount, and beneficalTradeRatioCount. With these features removed, this resulted in the final ABT to be used for model creation.

Analytics Base Table

The final ABT is 17 columns wide, and 6,131 rows deep. This data contains 14 numeric values, and 3 ids. The information ranges in skill of players, characters, stage choice, and time of match. However, when conducting a tournament, there is a wide level of range for every single game that is played, all unique in their own right. Therefore, this dataset is an accurate description of what competitive Super Smash Brothers Melee looks like, and the many different personalities and skills that are presented at events like these.

```
wavelandCount     wavedashCount     airDodgeCount     neutralWinRatioCount dashDanceCount
Min.   :  0.000   Min.   :  0.00   Min.   : 0.000   Min.   : 0.000       Min.   :  0.00
1st Qu.:  2.000   1st Qu.:  9.00   1st Qu.: 1.000   1st Qu.: 6.000       1st Qu.:  9.00
Median :  5.000   Median : 17.00   Median : 2.000   Median : 9.000       Median : 23.00
Mean   :  7.074   Mean   : 21.15   Mean   : 2.323   Mean   : 9.662       Mean   : 31.64
3rd Qu.: 10.000   3rd Qu.: 28.00   3rd Qu.: 3.000   3rd Qu.:13.000       3rd Qu.: 44.00
Max.   :218.000   Max.   :223.00   Max.   :41.000   Max.   :36.000       Max.   :447.00

openingsPerKillRatio   rollCount      counterHitRatio  spotDodgeCount   damagePerOpeningRatio
Min.   : 0.000        Min.   : 0.000   Min.   :0.0000   Min.   : 0.000   Min.   :  2.00
1st Qu.: 4.500        1st Qu.: 1.000   1st Qu.:0.4167   1st Qu.: 0.000   1st Qu.: 16.60
Median : 5.750        Median : 3.000   Median :0.5000   Median : 1.000   Median : 19.82
Mean   : 6.357        Mean   : 3.687   Mean   :0.5000   Mean   : 1.939   Mean   : 20.68
3rd Qu.: 8.000        3rd Qu.: 5.000   3rd Qu.:0.5833   3rd Qu.: 3.000   3rd Qu.: 23.73
Max.   :29.000        Max.   :33.000   Max.   :1.0000   Max.   :24.000   Max.   :137.29
NA's   :409                            NA's   :402                       NA's   :270
  killCount     conversionCount   opponentId      stageId       totalDamage     characterId
Min.   :0.000   Min.   : 0.00   Min.   : 0.00   Min.   : 2.00   Min.   :  0.0   Min.   : 0.00
1st Qu.:2.000   1st Qu.:13.00   1st Qu.: 2.00   1st Qu.: 5.00   1st Qu.:244.3   1st Qu.: 2.00
Median :3.000   Median :17.00   Median : 9.00   Median :28.00   Median :357.8   Median : 9.00
Mean   :2.874   Mean   :16.56   Mean   :10.67   Mean   :19.76   Mean   :335.6   Mean   :10.67
3rd Qu.:4.000   3rd Qu.:21.00   3rd Qu.:19.00   3rd Qu.:31.00   3rd Qu.:447.1   3rd Qu.:19.00
Max.   :4.000   Max.   :47.00   Max.   :25.00   Max.   :32.00   Max.   :820.7   Max.   :25.00

inputsPerMinuteRatio
Min.   :  0.0
1st Qu.:320.3
Median :376.8
Mean   :374.6
3rd Qu.:429.6
Max.   :923.0
```

Modeling

For the modeling portion of this data project, the use of 6 common algorithms were used to analyze the data: 1) Linear Model; 2) OneR Model; 3) Naïve Bayes Model; 4) Decision Tree Model; 5) Regression Tree Model; 6) Rules Model.

Linear Model

The first model that was run on the dataset was a simple linear model, in hopes to see if there was a simple linear tend within the data. For this model, the results that held "NA" values had to be removed, as these values were not compatible in the creation of a linear model.

```
Call:
lm(formula = gangScriptData$killCount ~ wavelandCount + wavedashCount +
    airDodgeCount + neutralWinRatioCount + dashDanceCount + rollCount +
    spotDodgeCount + conversionCount + opponentId + stageId +
    totalDamage + characterId + inputsPerMinuteRatio, data = gangScriptData[-11])

Coefficients:
        (Intercept)        wavelandCount        wavedashCount         airDodgeCount
          0.9329556            0.0088296           -0.0016557           -0.0066421
neutralWinRatioCount       dashDanceCount            rollCount       spotDodgeCount
          0.0162381            0.0004122           -0.0107416           -0.0140860
     conversionCount           opponentId              stageId          totalDamage
         -0.0440546           -0.0008647           -0.0026862            0.0076155
       characterId  inputsPerMinuteRatio
         -0.0067140            0.0003571
```

This model gave the results shown above, a surprisingly had an accuracy of 43.1% when run on the test data. On interesting note that came from this information if that if the player were to not move or include any inputs, they were expected to get at least one stock a game.

OneR

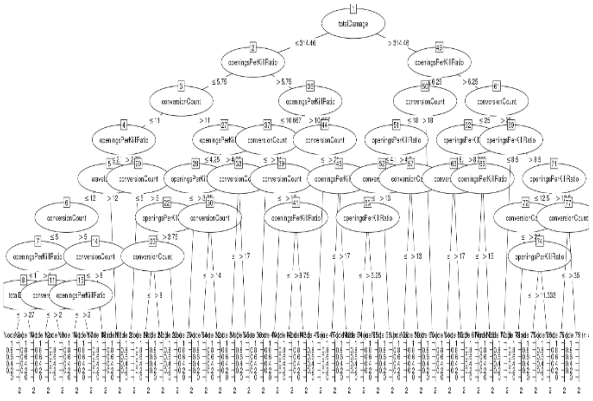
The next model that was rule was a simple OneR analysis, to discover if there is one feature that was best in explaining the data. To begin this step, supervised binning was performed to reduce the risk of overfitting. However, because the model was only looking for one feature of the 17 listed, severe overfitting still occurred. This model showed to be unsuccessful as expected, and simply memorized the feature of totalDamage. This resulted in a 0% overall accuracy and ended up not being very helpful to the study.

Naïve Bayes

After OneR analysis was complete, the next model that was conducted was a Naïve Bayes model. Similar to the OneR model, supervised binning was performed to reduce the risk of overfitting. Unfortunately, this model also resulted in a 0% accuracy, and proved to be unhelpful for the overall research. However, the flaw in this model came with issues done in the binning. With more careful binning, based solely of the features and how they relate rather than the generic algorithm, this model could have been effective.

Decision Tree

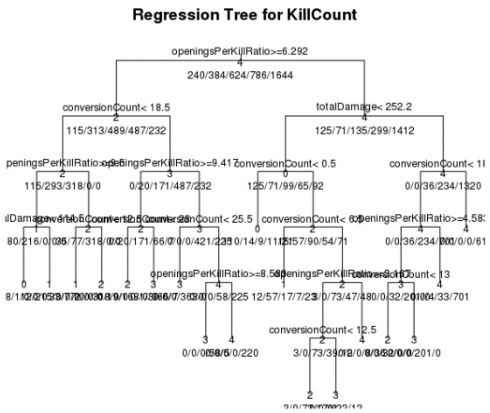
Once those models were complete, the next step in the process was to evaluate the use of tree models, the first of which being a decision tree. The decision tree algorithm that was used in this study was the C4.5, and this model yielded successful results, that appeared to be very reliant on three variables: 1) totalDamage; 2) conversionCount; 3) openingsPerKillRatio. The final model produced an accuracy of 90.4%, with a kappa statistic of 0.8638.



Regression Tree

The regression tree algorithm that was used in this study looked similar to the results shown in the decision tree. Both algorithms were very sensitive to the same three variables, however the regression tree produced much less splits than the decision tree. The summary statistics for the prediction model showed an MSE of .333,

an RSME of .577, and an MAE of .284. The result of the tree is shown below.



Rules Model

Once those models were complete, the last step was to create a rules model. The rules algorithm that was used in this study was with the function ripper, and this model yielded the most successful results, however relied on the same three variables as the decision tree and regression tree models. The overall accuracy for this model was 91.32%, with a kappa statistic of 0.8729. These results are very similar to the Decision Tree, and it could be said that the choice between the two is negligible. The results for this model are shown below.

```
JRIP rules:
===========

(totalDamage <= 260.079994) and (openingsPerKillRatio >= 9) and (conversionCount <= 17) => killCount=1 (215.0/0.0)
(conversionCount <= 8) and (openingsPerKillRatio >= 5) => killCount=1 (65.0/0.0)
(openingsPerKillRatio >= 13) and (conversionCount <= 23) => killCount=1 (49.0/0.0)
(conversionCount <= 4) and (openingsPerKillRatio >= 3) => killCount=1 (12.0/0.0)
(conversionCount <= 11) and (openingsPerKillRatio >= 8) => killCount=1 (5.0/0.0)
(totalDamage <= 48.5) and (dashDanceCount >= 17) => killCount=1 (4.0/1.0)
(totalDamage <= 343.076004) and (openingsPerKillRatio >= 6.5) and (conversionCount <= 19) => killCount=2 (297.0/1.0)
(openingsPerKillRatio >= 8.5) and (conversionCount <= 25) => killCount=2 (161.0/0.0)
(conversionCount <= 12) and (openingsPerKillRatio >= 4.5) => killCount=2 (91.0/0.0)
(conversionCount <= 8) and (openingsPerKillRatio >= 3) => killCount=2 (21.0/0.0)
(openingsPerKillRatio >= 13) => killCount=2 (18.0/3.0)
(conversionCount <= 5) and (openingsPerKillRatio >= 1.5) => killCount=2 (5.0/0.0)
(openingsPerKillRatio >= 6.5) and (conversionCount <= 16) => killCount=2 (17.0/0.0)
(openingsPerKillRatio >= 5.666667) and (conversionCount <= 22) => killCount=3 (337.0/0.0)
(openingsPerKillRatio >= 7.666667) and (conversionCount <= 30) => killCount=3 (233.0/0.0)
(conversionCount <= 16) and (openingsPerKillRatio >= 4.333333) => killCount=3 (123.0/0.0)
(openingsPerKillRatio >= 10.333333) => killCount=3 (32.0/2.0)
(conversionCount <= 12) and (openingsPerKillRatio >= 3.333333) => killCount=3 (32.0/0.0)
(conversionCount <= 9) and (openingsPerKillRatio >= 2.333333) => killCount=3 (11.0/0.0)
(conversionCount <= 6) and (openingsPerKillRatio >= 2) => killCount=3 (2.0/0.0)
 => killCount=4 (1613.0/2.0)

Number of Rules : 21
```

Evaluation

After reviewing the success of all the models that were run, it appeared that for this data set, the best algorithm was in the Rules model that was run. Moving forward, there appeared to be a heavy reliance on only three variables however, so in the future it would be helpful to view the data with these three variables removed.

Regarding the concept of deployment, the best way to contextualize this concept is basically through individuals adjusting their player to fit whatever is most successful in taking stocks. For example, based on the Decision Tree model, it seems apparent that in all matchups, players should prioritize the total damage they output, the number of conversions the have, and the rate at which they kill based on the number of openings. Therefore, when a player is looking to improve, these could be topics of interest.

Unfortunately, these results seem somewhat intuitive, and do not share too much about the current meta game for top players. Essentially the data is suggesting that if you are successful in the punish game and the mental game, then you will win the match, which is a concept that has been known for years. The important information to gather would be how to improve these measures, not to just tell us that they are important.

Future Work

The models were not very successful, simply because the data provided is incredibly complicated and hard to truly

evaluate properly. When playing a game against another individual, there are so many extraneous features that go into the match. Especially in Melee, with each minute played, there are thousands of decisions that need to be processed, small precise movements that need to be made, and all an extremely small frame of time. The moves a player makes in each circumstance is unique to that player alone, and when so many play styles leads to varying levels of success, it is very difficult to truly understand what allows one player to win and one player to lose. For something like this to be successful, many more features need to be collected, and analyzed on a much smaller scale, specially paying close attention to individual performances rather than generalizing by character.

However, this is the first operational version of Project Slippi, and in the future, there will continue to be plenty of developments in the code that can gather more useful data. More information on specific players, more information on character specific matchups, more information on character specific technical traits. Information regarding high pressure situations, or out of game data that could affect the match. These are all valuable pieces of data that can be recorded, and as the platform develops, the work in the field will only become stronger.

Furthermore, this is just one tournament of many. As more tournament organizers adopt the use of Project Slippi, there will be thousands of tournaments to have directly uploaded to the database, each tournament with thousands of games to review and learn from. This is truly just the beginning, and the possibilities for future work truly are endless.

Conclusion

This project could not have been done without the tireless hours of work that was put in by the Project Slippi team. Their effort has allowed for a new way to see a game that has been closely analyze for over a decade, and the information that could be gained from this new perspective has the potential to bring a great source of happiness to plenty of people. Thank you, Project Slippi, for the vision and insight you have brought to the Melee community.

Another thank you to Professor Read and the Applied Data Science Course that this project was created from. Without your instruction and support, this project could not have been completed.