LEAGUE OF LEGENDS POP CULTURE
HOW THE GAME IS AFFECTED BY SUBREDDITTING

Aleksa Kostic 52228237 akostic@uci.edu

Baolong Truong 86117369 baolonlt@uci.edu

GitHub
https://github.com/aleksa-kostic/portfolio/tree/main/Capstone

1 INTRODUCTION AND PROBLEM STATEMENT

League of Legends is a popular, free online game that has been available for well over a decade. Many people who avidly play the online game League of Legends take to Reddit in the event of exciting news such as new champions (which are the playable characters in the game), new champion abilities, innovative game plays by competitive players, and so on. These events tend to be associated with increased champion play rates in the game. For example, after the release of the Netflix original TV series "Arcane", which is a show based on the game centered around champions Jinx and Vi, League players saw increased turnout of Jinx and Vi across all types of games within League of Legends, especially ranked games. The goal of our project is to see if we can, using data, highlight a correlation between Reddit threads from r/leagueoflegends, a subreddit with 5.7 million subscribers, and Riot Games's developer data on champions' play rates in League of Legends games. Given that a correlation exists, we use subreddit and play rate data to create a model that predicts whether a subreddit thread will or will not make a significant change in champion play rate utilizing multiple modes of parameters such as text featurization, comment approval, and play rates around the time of subreddit postings.

2 RELATED WORK

Sentiment analysis with word2vec: Medium - Yelp Sentiment Classification Using Word2Vec (https://medium.com/swlh/sentiment-classification-using-word-embeddings-word2vec-aedf28fbb8ca). This method gave a little bit of insight as to how to get a vector for a whole comment, essentially that a simple, non-weighted average suffices in the grand scheme of multinomial prediction. Otherwise, the word2vec development was of our own research with respect to the Gensim NLP library.

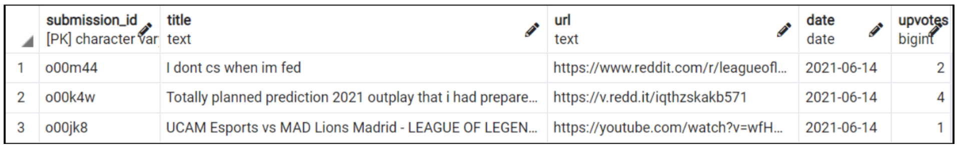
League of Legends Match Crawling: Crawling Matches Using the Riot Games API (https://hextechdocs.dev/crawling-matches-using-the-riot-games-api/). We used "Method 1: Using league-v4 in conjunction with match-v5" in the above article to gather play rate data for each champion. The article above describes a crawling method using only the Riot API to collect the data we need.

Machine Learning Methods: Feature Analysis to League of Legends Victory Prediction on the Picks and Bans Phase (https://ieee-cog.org/2021/assets/papers/paper_292.pdf). This research article explains how feature analysis was used to predict League of Legends victory based on the picks and ban phase. The models that were used in experiments were Logistic Regression, Decision Trees, Naive Bayes, k-Nearest Neighbors, Random Forest, and Support Vector Machines. Since we are also attempting classification based on feature analysis, we would also like to try the algorithms used in this research article.

3 Data Sets

3.1 Reddit Data

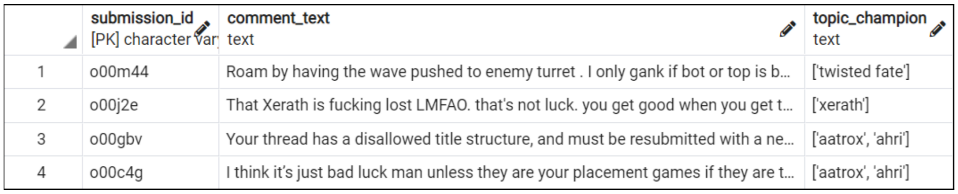
Using the Reddit PushShift API (https://pushshift.io/api-parameters/), which is a Reddit API that is a collective data dump of everything that is on Reddit, we collected threads from the League of Legends subreddit from the start of June 2021 until the end of March 2022. From this, there were a total of 143,567 valid threads, where valid is defined as data entries whose links linked to actual threads, not images or videos, or a domain that was not Reddit. The features pulled for each data entry include the "submission" (thread) id, the title of the thread, the URL leading to the thread on Reddit, the date of creation, and the number of upvotes as in the net total of vote reactions on a thread. Table displayed in Figure 1.



	submission_id [PK] character var	title text	url text	date date	upvotes bigint
1	o00m44	I dont cs when im fed	https://www.reddit.com/r/leagueofl...	2021-06-14	2
2	o00k4w	Totally planned prediction 2021 outplay that i had prepare...	https://v.redd.it/iqthzskakb571	2021-06-14	4
3	o00jk8	UCAM Esports vs MAD Lions Madrid - LEAGUE OF LEGEN...	https://youtube.com/watch?v=wfH...	2021-06-14	1

Figure 1: submissions data table retrieved from SQL

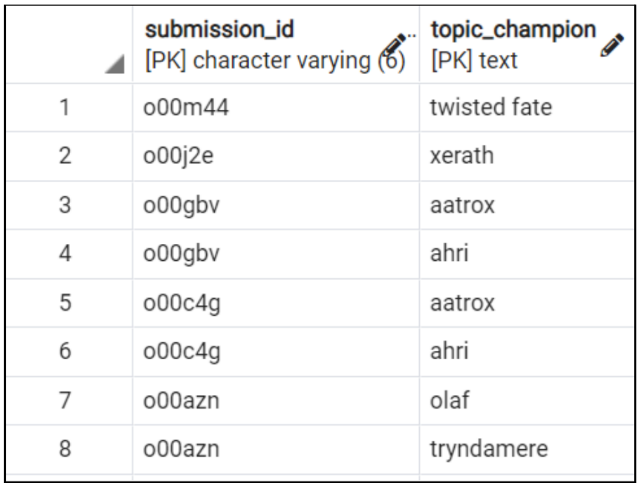
Then, using the PRAW Reddit API wrapper (https://praw.readthedocs.io/en/stable/) we collected all of the comments from each of the 140k+ threads to get 819,322 comments. These comments were then concatenated and paired with their mother submission id, and then labeled with the champion of topic within the text (up to three most frequently mentioned). The table, after grouping comments by submission, has around 88,000 entries. An example of this is shown in Figure 2.



	submission_id [PK] character var	comment_text text	topic_champion text
1	o00m44	Roam by having the wave pushed to enemy turret . I only gank if bot or top is b...	['twisted fate']
2	o00j2e	That Xerath is fucking lost LMFAO. that's not luck. you get good when you get t...	['xerath']
3	o00gbv	Your thread has a disallowed title structure, and must be resubmitted with a ne...	['aatrox', 'ahri']
4	o00c4g	I think it's just bad luck man unless they are your placement games if they are t...	['aatrox', 'ahri']

Figure 2: submissions_champions_arrays data table retrieved from SQL

The table in Figure 2 is then transformed into a table that follows first normal form rules, having 148,662 entries and excluding the comment text. An example of this table is show in Figure 3



	submission_id [PK] character varying (6)	topic_champion [PK] text
1	o00m44	twisted fate
2	o00j2e	xerath
3	o00gbv	aatrox
4	o00gbv	ahri
5	o00c4g	aatrox
6	o00c4g	ahri
7	o00azn	olaf
8	o00azn	tryndamere

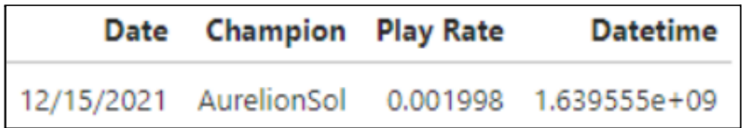
Figure 3: From data table submissions_champions from SQL. Table with submission id and the champion of topic in first normal form (1NF). For example, rows 5 and 6 correspond to one subreddit thread where two champions are equally the most frequently mentioned champion in the thread. Row 1 suggests that the thread with id 'o00m44' had only one champion mentioned the most whose name is 'twisted fate'

After building a Word2Vec model, visualized after principal component analysis on page 4, on the entire corpus of comments, we created average word vectors for each submission and appended it to a new table with 100 columns of feature values coming from the average word vector of 100 features. See Figure 5 below.

3.2 Riot Match Data

Riot Games API Documentation: https://developer.riotgames.com/apis. The second data set that we utilized was from the Riot Games League of Legends API. The API allows us to access historical match data for every game played in League of Legends from June 2021 to the present. With information on every match played, we can then calculate the daily play rate statistics for all 159 champions in the game.

The final dataset includes the following columns: Date: The date of the play rate data. For example, a date of "2022-01-01" indicates that the play rate data was collected for January 1st, 2022; Champion: The champion that the play rate data was collected on. This will be used as a key to join with the topic_champion column from the Reddit data to build our model; Play Rate: The play rate of the champion on the specific date. For example, a play rate of "0.05" indicates that the champion appeared in 5% of games on that specific day; Datetime: A simple unix timestamp of the date. Will be used to join with the datetime column from the Reddit data in order to match play rate data with the date of the Reddit submissions.



Date	Champion	Play Rate	Datetime
12/15/2021	AurelionSol	0.001998	1.639555e+09

Figure 6: Riot Play Rate Data Example Row

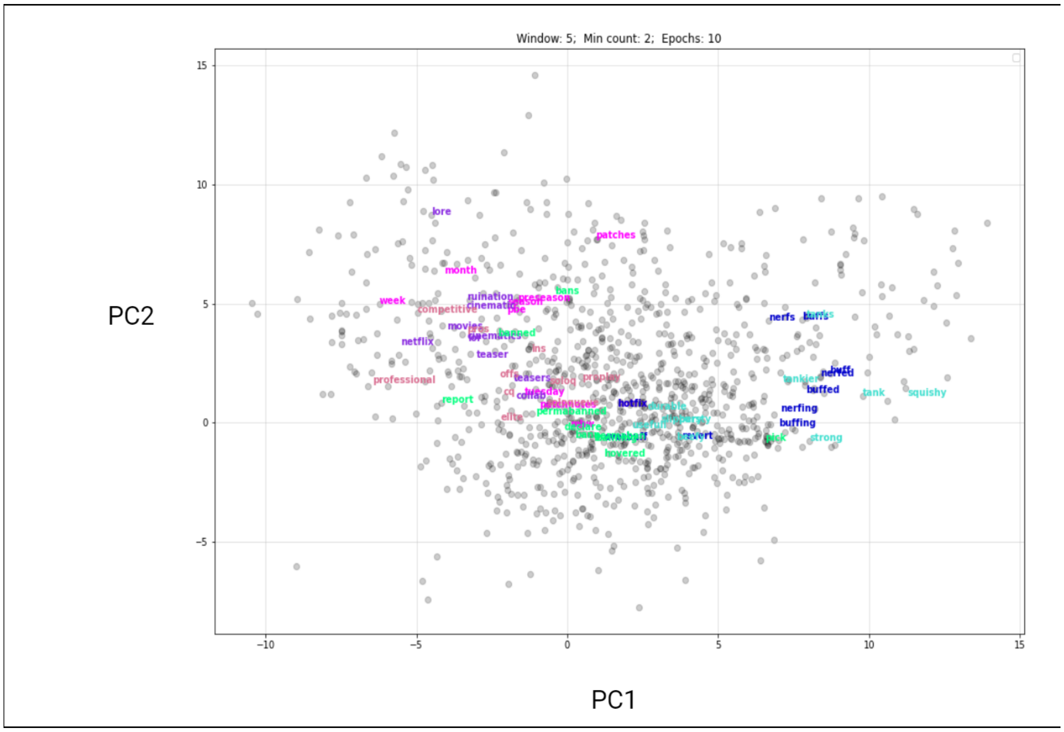
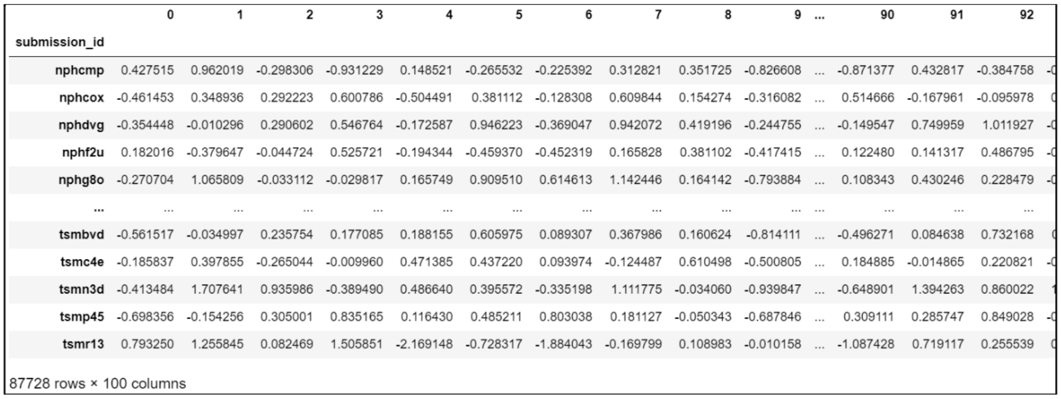


Figure 4: Word2Vec model cast from 100 features down to 2 dimensions via PCA (principal component analysis) for visualization purposes. Works in hot pink are related to the word “patch”, purple represents that of “arcane”, light green to “ban”, dark blue to “nerf”, cyan to “squishy”, and dark pink to “ranked”.



submission_id	0	1	2	3	4	5	6	7	8	9	...	90	91	92
nphcmp	0.427515	0.962019	-0.298306	-0.931229	0.148521	-0.265532	-0.225392	0.312821	0.351725	-0.826608	...	-0.871377	0.432817	-0.384758
nphcox	-0.461453	0.348936	0.292223	0.600786	-0.504491	0.381112	-0.128308	0.609844	0.154274	-0.316082	...	0.514666	-0.167961	-0.095978
nphdvg	-0.354448	-0.010296	0.290602	0.546764	-0.172587	0.946223	-0.369047	0.942072	0.419196	-0.244755	...	-0.149547	0.749959	1.011927
nphf2u	0.182016	-0.379647	-0.044724	0.525721	-0.194344	-0.459370	-0.452319	0.165828	0.381102	-0.417415	...	0.122480	0.141317	0.486795
nphg8o	-0.270704	1.065809	-0.033112	-0.029817	0.165749	0.909510	0.614613	1.142446	0.164142	-0.793884	...	0.108343	0.430246	0.228479
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
tsmbvd	-0.561517	-0.034997	0.235754	0.177085	0.188155	0.605975	0.089307	0.367986	0.160624	-0.814111	...	-0.496271	0.084638	0.732168
tsmc4e	-0.185837	0.397855	-0.265044	-0.009960	0.471385	0.437220	0.093974	-0.124487	0.610498	-0.500805	...	0.184885	-0.014865	0.220821
tsmn3d	-0.413484	1.707641	0.935986	-0.389490	0.486640	0.395572	-0.335198	1.111775	-0.034060	-0.939847	...	-0.648901	1.394263	0.860022
tsmp45	-0.698356	-0.154256	0.305001	0.835165	0.116430	0.485211	0.803038	0.181127	-0.050343	-0.687846	...	0.309111	0.285747	0.849028
tsmr13	0.793250	1.255845	0.082469	1.505851	-2.169148	-0.728317	-1.884043	-0.169799	0.108983	-0.010158	...	-1.087428	0.719117	0.255539

87728 rows × 100 columns

Figure 5: Data from submission_vectors.csv showing a submission with its corresponding average vector split into columns. Table is cut off on right hand side, hiding columns 93 through 99

An example row of our dataset is shown in Figure 6 on page 3. This row indicates that on December 15th, 2021, the champion “Aurelion Sol” appeared in approximately 0.1% of games (a quite unpopular champion).

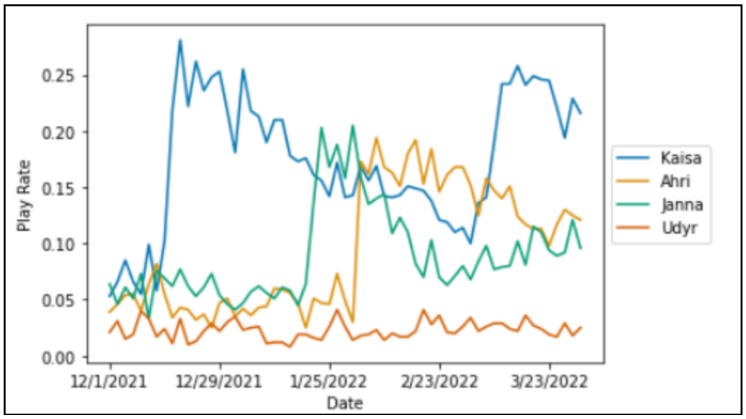


Figure 7: Champion Play Rate Over Time

Figure 7 visualizes how champion play rate over time changes often. Four champion play rates are shown, all with different play rate trends. The champions Kaisa, Ahri, and Janna are all shown to have periodic spikes and drops in play rate over time. However, the champion Udyr has stayed unpopular over this time slot. Our project will attempt to find Reddit threads that correspond to the reasons for these spikes and drops in play rate.

4 Overall Technical Approach

4.1 Riot Match Collection

Riot Games Developers's API does not have readily available data concerning play rates. We used a web-crawling-inspired method to collect daily play rate, starting with a random seed for a specific date, an initial match to start crawling on. Recursively, we used the Riot API to collect which champions were played as well as all of the players present in a specific game. We then chose a random player from that match and repeated the crawl. We repeated this process for each day from July 2021 until March 2022 to collect daily play rate data. After collection, the data was stored in CSV files, cleaned, and uploaded to our PostgreSQL AWS RDS.

4.2 Reddit Data Collection

We first used the PushShift Reddit API, a frequently-updated Reddit data dump API, to collect the threads or "submissions" matching dates from June 2021 to March 2022 from the League of Legends subreddit, a sub directory on Reddit. In this collection, we collected submission IDs, publication dates, titles, and URLs. This data collection, after cleaning, was stored as a CSV file. Then we used the PRAW Reddit wrapper to collect all of the comments associated with the collected threads. Once a table was put together, we saved it into a CSV file. We then used PRAW to append submission upvotes to the submissions table. The CSV tables were updated and then uploaded to our PostgreSQL AWS RDS.

4.3 Word2Vec

Using the corpus of comments from the Reddit data collection. We used the Gensim NLP library to produce Word2Vec models. Using combinations of parameters, we produced 54 models to see which one would featurize our words the best. Our first evaluation method was visualization via PCA. It turned out that the spread and location of certain groups of words did not change much relatively between groups. This told us that the Word2Vec models, independent of parameter combination, produced similar results on the same corpus of text. So, the second metric of evaluation was utilizing "ground truth" pairs of words that were contextually relevant to the game and that we'd expect to be fairly close together in terms of vector distance (for word vectors of 100 features). We ran similarity tests on all of the models, picked the models that produced the shortest distance for each of the pairs of words, and then picked the model that appeared most frequently across all seven pairs' top models. We then calculated the average word vector for all submissions, first by calculating the average word vector for a comment, then averaging the comment vectors per submission ID, and finally creating a table with submission IDs with their corresponding vector split into 100 columns, and uploaded to SQL.

4.4 PostgreSQL AWS RDS

Within our cloud database, we added constraints to enforce relationships between tables as well as primary key constraints to enforce unique entries as appropriate. Additionally, some functions have been developed such as a function that takes a champion name and a date, and then calculates whether there was significant change in play rate in the two weeks after the inputted date; this function uses a simple two-sample t-test to return a Boolean of significant change, though we'll be considering Poisson prediction soon.

4.5 SKLearn Regression Modeling

The models we used from SKLearn were logistic regression, and random forest. We used logistic regression to predict the probability that an observation belongs to one of two possible classes. For our complex problem of predicting change from text data, it was a good place to start. We used 101 numeric features. There were two main metrics for evaluating how well our model function, accuracy and confusion matrices.

The next model that we used was random forest modeling. It does well in classification when given data of imbalanced class sizes. Using an automated cross-validation method in SKLearn, we looked at combinations of 2000 trees. Additionally, we instantiated the random search, and used 3-fold cross validation. We viewed the best parameters from fitting the random search: with 400 trees, 10 minimum samples for splitting, maximum tree depth at 70 layers, and with bootstrapping. Like with logistic regression, the two main metrics for evaluating how well our model function, accuracy and confusion matrices.

5 Software

SKLearn	Logistic regression modeling, random forest modeling, cross validation, optimal model selection, analysis
API Scripting	Using a combination of Pandas, Riot Developer's API, Reddit's PushShift and PRAW APIs, we wrote scripts to collect all of the data used in this project
AWS RDS: PostgreSQL	Not only in the use of storing data in tables, but also creating constraints, indices, and complex functions to ease in the viewing of select data needed for our project's purposes
Gensim Word2Vec	Utilized for featurization of our subreddit's corpus vocabulary. This model gave us 100 features to use for each word and then each subreddit thread to be used in our predictive modeling

6 Experiments and Evaluation

6.1 Logistic Regression Classification

6.1.1 Justification

Binary Classification: Our response variable is binary. We want to predict a class from 101 numeric features, easily translated into a Logistic Regression model. Possibly not the best model, but wanted to start with something simple.

6.1.2 Assumptions

1. Binary response: Whether or not there was a change in play rate

2. Independent observations: Each Reddit submission independent from the previous

3. Non-multicollinear features: A few features from the Word2Vec vectors have V.I.F. scores of over 10, indicating problematic collinearity.

4. Sufficient sample size: $>$30,000 samples with least frequent outcome having $\sim$3000 outcomes.

5. Linearity of independent variables and log-odds: Figure 8 on page 8 shows a plot of the linear relationship between continuous predictor variables and the logit of the outcome. None of the predictors appear to have a linear relationship, violating the assumption and invalidating logistic regression.

Figure 9 on page 8 shows the ROC curve for our logistic regression model. The curve comes close to the diagonal of the plot, indicating that the model does not demonstrate good performance.

6.1.3 Preliminary results

Overall, our logistic regression model was unacceptable. The combination of the classes of data being overwhelmingly unbalanced at a 11:1 ratio, the lack of colinearity in model features, and the less-than-appropriate area under the curve for the ROC curve demonstrate that logistic regression is not the appropriate method for the prediction modeling that we aim to create.

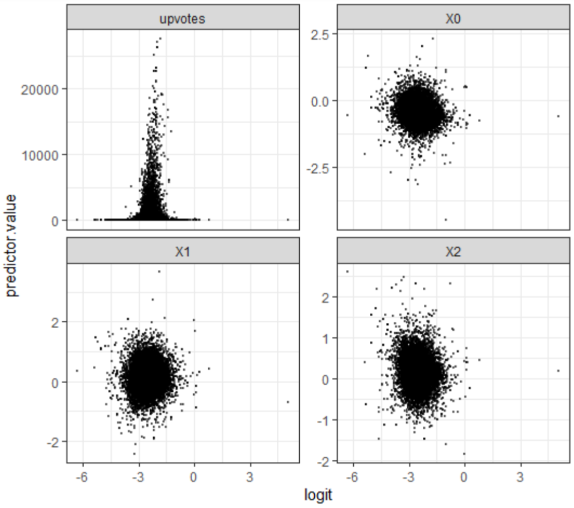


Figure 8: Predictor Values vs Logit Values of 4 Variables

		Predicted Label	
		No Significant Change	Significant Change
True Label	No Significant Change	22589	7
	Significant Change	1955	3

Table 1: Confusion matrix for training data on the logistic regression model.

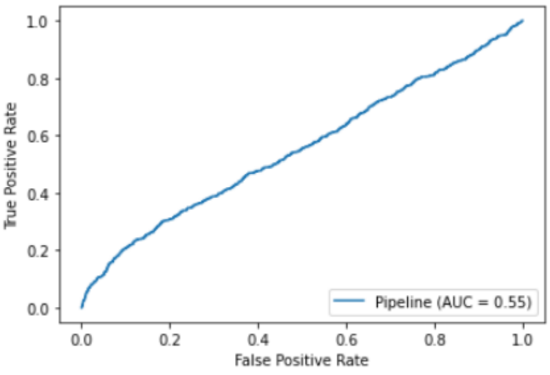


Figure 9: R.O.C. Curve for Logistic Regression

Table 1 shows the confusion matrix for the prediction classes of the training data in our logistic regression model. Where the data does actually have a significant change in play rate, the model predicts incorrectly to an extreme fault. Yet, where the data does not have a significant change in play rate, the model almost always predicts that there is not in fact a significant change in play rate. Table 2 on page 9 showing the confusion matrix for the prediction classes of the testing data follows the same patterns. These errors are easily attributed to the imbalanced data. As such, our next step is in working with random forest modeling which performs better in class separation and categorization despite imbalanced data.

6.2 Random Forest Modeling

6.2.1 Justification

Such as in logistic regression, we want to train a classification model, but random forest modeling can handle non-linear data and imbalanced class sizes.

		Predicted Label	
		No Significant Change	Significant Change
True Label	No Significant Change	7533	2
	Significant Change	649	1

Table 2: Confusion matrix for testing data on the logistic regression model.

6.2.2 Assumptions

Binary response variable: Whether or not there was a change in play rate. Best Split: At each step of building individual tree we find the best split of data. Sampling: While building a tree we use not the whole dataset, but bootstrap sample. Averaging: We aggregate the individual tree outputs by averaging.

6.2.3 Results

As seen in Table 3 on this page and Figure 10 on page 9, random forest modeling didn't do too much better at classification when compared to logistic regression. We still see a bias towards predicting ”no significant change” and also we see that the model does not perform well as a class separator. Sensitivity improved in the random forest modeling and with subsetting the data, but the lack of large improvements can be attributed to our data wrangling and labeling.

		Predicted Label	
		No Significant Change	Significant Change
True Label	No Significant Change	7399	136
	Significant Change	626	24

Table 3: Confusion matrix for testing data on the best random forest model

7 Notebook Description

Our project was spread across multiple notebooks. With at least ten notebooks, each notebook focused on a sub-task, for example, word-to-vec development and then word-to-vec evaluation, etc. We felt that all components of the project were equally important in leading to our ultimate goal of finding a significant prediction model, or at least in highlighting the shortcomings within the process. Parts that were merely testing cells were removed so as to leave behind notebooks that were clear from start to finish.

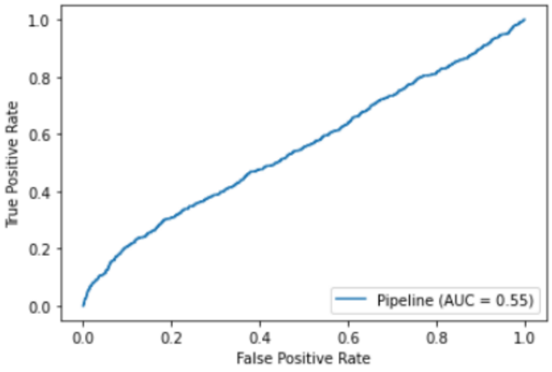


Figure 10: R.O.C. Curve for Random Forest Modeling.

8 Members Participation

Baolong handled crawl script development, Baolong and Aleksa mostly did the data collection for the riot data. As for the Reddit data collection, Aleksa wrote the collection script, and Baolong and Aleksa collected all of the data. Word2Vec modeling, from development, PCA visualization, evaluation, and utilization to convert into vector tables, Aleksa did this primarily independently. Aleksa set up the PostgreSQL AWS RDS and handled index, constraint, and function creation. Baolong and Aleksa both uploaded tables to the server through Python, re-downloading, editing, and re-uploading some tables as necessary. Baolong led the team in predictive-model-building with Aleksa handling coding and debugging. Baolong delegated slides to make for presentations and participated in writing the report with Aleksa.

9 Discussion and Conclusion

Through the use of logistic regression and a random forest modeling, we learned the importance of assumption checking. During model diagnosis, we found that many assumptions were not met. Another technique that we learned was subsetting the data to account for class imbalance. We also learned about the benefits of a random forest model over a logistic regression model. Since our predictors did not have a linear relationship with the log-odds from the logistic regression model, we found that a random forest was a more suitable model because it is a non-linear classifier.

We also learned the importance of robust method planning. We did not foresee that our champion labeling function would lead to incorrect assumptions until late into the project. We did not account for the situation where our algorithm would mark a post as causing a significant change in play rate even though it was a different post that contributed to the change. Furthermore, we would've benefited from planning ahead on how we were going to combine data from the two datasets in a more meaningful way. Primarily, we did not know the most optimal way to label each posts' champion of topic. There were many situations where a champion appeared in a post the same amount of times as another champion or the comments mentioned a champion but the champion was not the primary focus of the thread.

With more time, we'd go back and use better methods for data collection. For example, we could have rebuilt the W2Vec model for 300 rather than 100 features, which could have given us better results. Additionally, we could've re-wrangled our data with more constraints, i.e. requiring the thread to be about ranked or professional games rather than considering every thread on the subreddit. We would also spend more time looking at different approaches for significance evaluation - something more sophisticated than a t-test of average play rates between two time blocks. Lastly, we could have attempted forecasting play rates from historical data rather than just using a classification model.