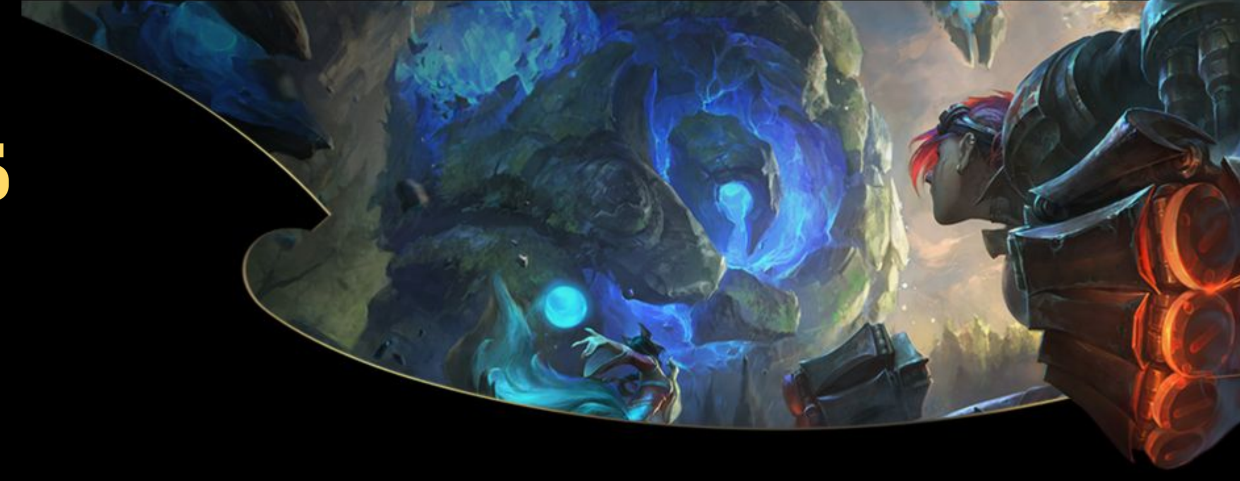
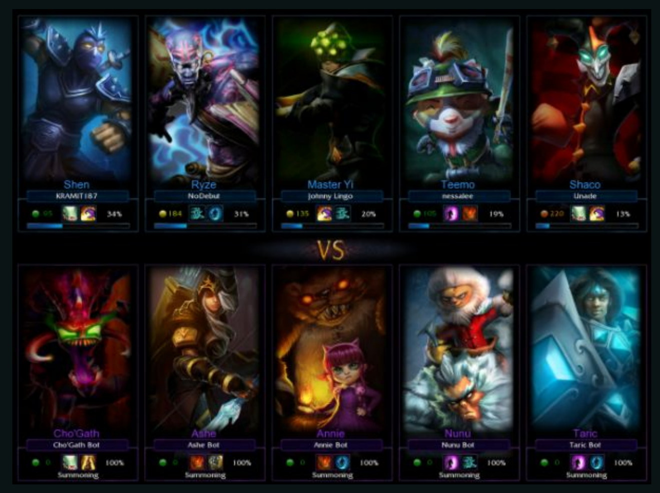


Predicting Seasonal Rank Changes in League of Legends

Se Won Jang / swjang@stanford.edu
Stanford University, CS 229 Machine Learning

What is LOL?

- LOL(League of Legends) is an online multiplayer game where 10 players are matched up in 2 teams of 5 to choose game characters and fight against each other.
- Each player's level is represented by their Rank (25 in total)
- I wanted to use the supervised learning techniques that I learned in CS229 to ***predict a player's rank at the end of the season***. Just because we all want to know how high we can get!

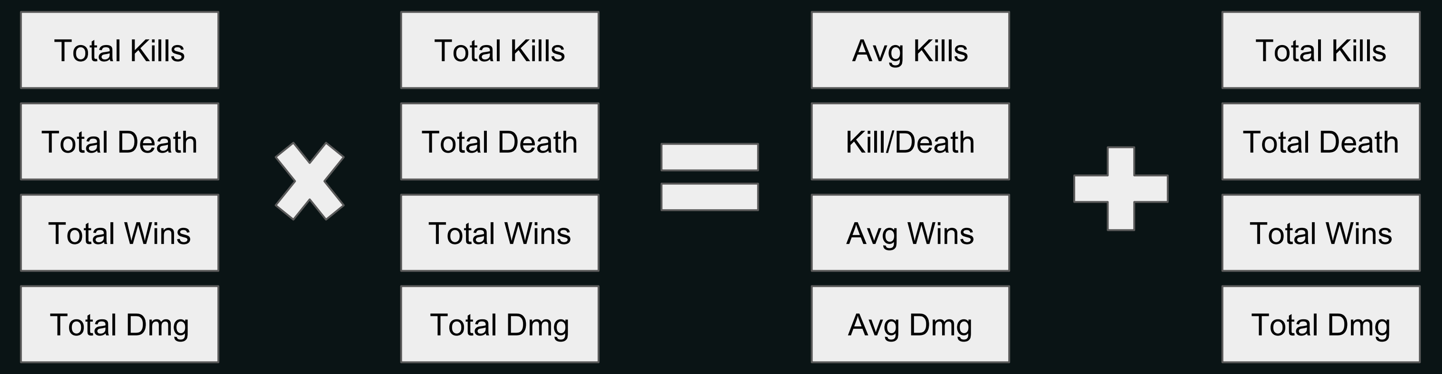


What is the Training Data?

- Riot (The developer of LOL) provides a rich API allowing us to query its internal gaming statistics of a given player.
- I used the Stats API which returns a JSON object per player :
{"totalPhysicalDamageDealt": 6562106, "totalTurretsKilled": 124, "curRank": 25, "totalSessionsPlayed": 236, "totalAssists": 2935, "totalDamageDealt": 11818577, "mostChampionKillsPerSession": 16, ...)

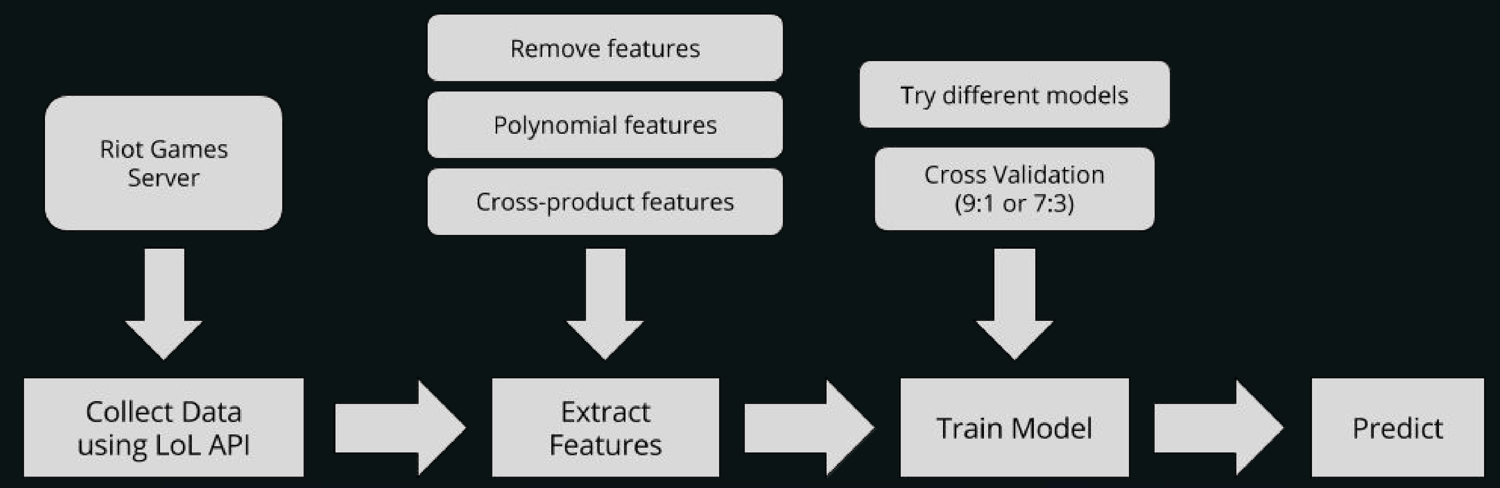
Selecting Features

- Transformed each JSON object into a python dictionary
- Manipulate the dictionary to represent meaningful information (ex. Death -> 1/Death, Kills -> Kills/Death). Total 57 Features!
- I used SVM with RBF Kernel for one of the learners, so did not add experimental cross terms. (only added those with human-level meaning)



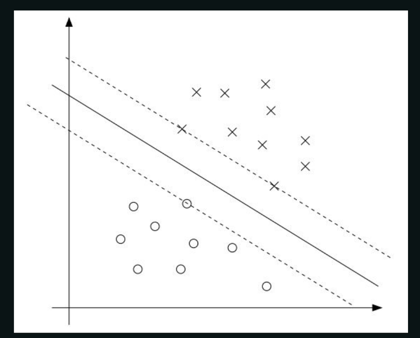
Learning and Prediction Process

- Once through with data collection / feature selection, I chose to try two different output classes :
  1. Predict the Exact End-season rank (25 classes total)
  2. Predict Change in rank (Down, Constant, Up)
- In order to compare 2 algorithms (Ada Boosting, SVM) for the 2 classification problems,I chose to use 10-fold cross-validation.



Choice of Learning Algorithms

- I wanted to answer "how harder would it be to classify out of a much larger class pool?".I chose Ada Boosting, and SVM. My hypothesis was that since I have 57 independent features that all contribute to the rank change independently, AdaBoosting would perform great for this problem.
- **Support Vector Machine (SVM) with RBF Kernel**
  - Linear classifier that uses hinge loss for its objective. Given labeled training data, outputs an optimal hyperplane which categorizes new examples.



- **AdaBoosting (Adaptive Boosting)**
  - Combines weak learners to a strong one

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \exp(-y^{(i)}\theta^T\phi(x^{(i)})).$$

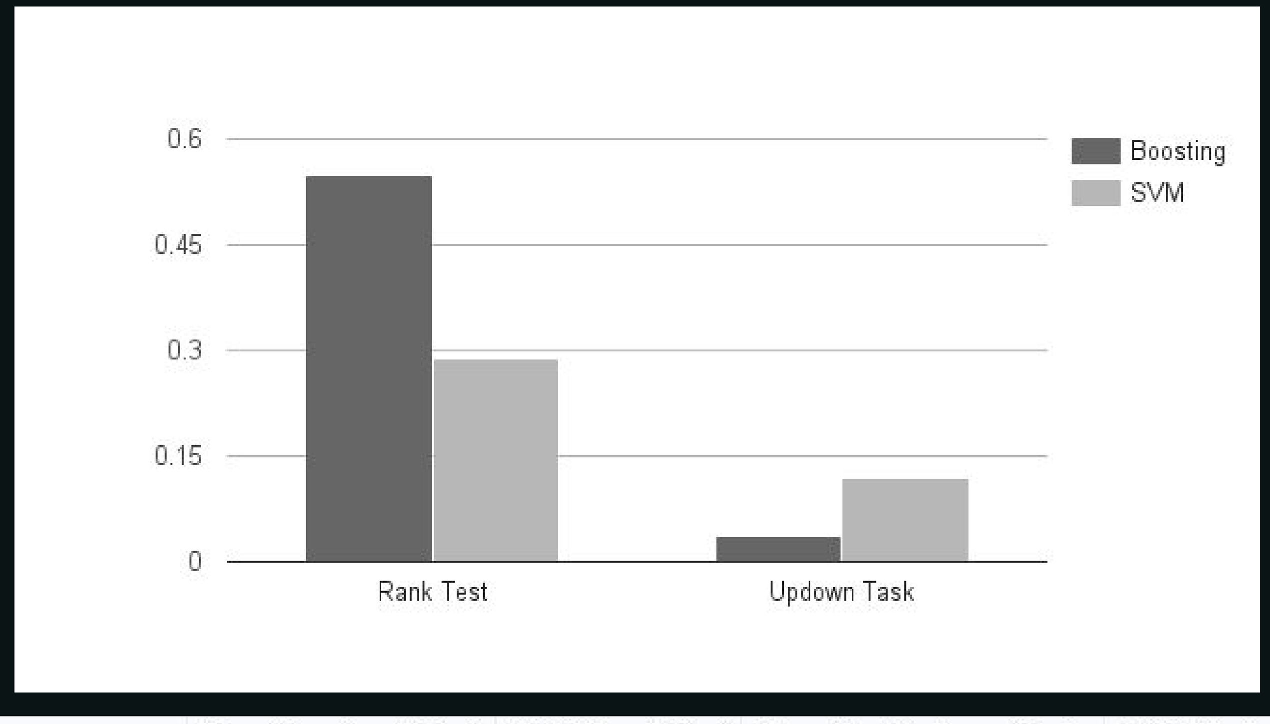
(i) **Input:** A distribution $p^{(1)}, \ldots, p^{(m)}$ and training set $\{(x^{(i)}, y^{(i)})\}_{i=1}^m$ with $\sum_{i=1}^m p^{(i)} = 1$ and $p^{(i)} \geq 0$

(ii) **Return:** A weak classifier $\phi_j : \mathbb{R}^n \to \{-1, 1\}$ such that

$$\sum_{i=1}^{m} p^{(i)} 1\left\{y^{(i)} \neq \phi_j(x^{(i)})\right\} \leq \frac{1}{2} - \gamma.$$

Prediction Results

- The two models were trained with 600 LOL player season data. The data were representative of the rank distribution of the LOL population.



	Boosting Rank Test	SVM Rank Test	Boosting Updown Task	SVM Updown Task
Before Feature Extraction	0.5873	0.3037	0.0661	0.1473
After Feature Extraction	0.5484	0.2874	0.0381	0.1202

Understanding the Results

- Interesting Results!
- Take 1 : Feature Extraction didn't make a lot of difference (1 ~ 3%)
  - Explanation : My hypothesis is that since the features are independent, adding their cross-terms don't have much effect
- Take 2 : Predicting out of 25 classes is extremely hard
  - Explanation : Both SVM and Boosting performed very poorly on this, although SVM still performed much better than Boosting. I think in order for boosting to work well with multi-class problems, we need a lot of dataset to cover many types of outcomes.
- Take 3 : Ada Boosting worked much better than SVM (Up Down)
  - Explanation : boosting algorithm performed very well with less than 5% error rate, compared to the SVM model. I think this is because there might not be an inherent relationship between the features that makes a player go up or down in rank. An outcome of a game may only be affected by one of the features, for instance.

Github Repository
https://github.com/sjang92/LOLRankPredictor

Reference
Riot API: https://developer.riotgames.com/api/methods MMR: https://na.op.gg/, https://na.whatismymmr.com Scikit-learn: http://scikit-learn.org
SVM: http://cs229.stanford.edu/extra-notes/cs229-notes3.pdf Boosting: http://cs229.stanford.edu/extra-notes/boosting.pdf

NOTE : Overlaps with my CS221 Project