# VISUALIZING ITEM DIVERSITY THROUGH DIMENSIONALITY REDUCTION

CONOR DOWLING AND ALEX BRIGHAM

## 1. Introduction

## 2. Data Pipeline

### 2.1. RiotWatcher.

The first step in the data pipeline is collecting the match data for the matches provided for the challenge. We used a Python module called RiotWatcher (github.com/pseudonym117/Riot-Watcher) to interface with the Riot API and systematically request the data for each game.

### 2.2. MongoDB.

We chose to store this match data in a MongoDB database, creating a table for each set of matches provided. MongoDB seemed like an obvious choice because it was easy to setup and could store the match data in raw JSON format with no editing required.

### 2.3. MapReduce.

Once we had gathered all of the data, we used MongoDB's built-in map-reduce functionality to extract the data that we wanted from each game and aggregate that data within regions and patches. We performed three separate map-reduce operations to obtains statistics on how many times champions were played, which items they built, and which builds were most popular.

### 2.4. Sci-Kit Learn.

## 3. Visualization

### 3.1. Javascript.

### 3.2. D3.

## 4. Data Sets

### 4.1. Build Diversity Across Champions.

### 4.2. Build Diversity Within Champions.

### 4.3. Champion Build Order Statistics.

## 5. Results

## 6. Build Diversity Across Champions

## 7. Item Diversity and Build Order Diversity Within Champions

## 8. Notable Trends

## 9. Conclusion

## 10. Future Work

### 10.1. Confidence Intervals.