Game Data Analysis of League of Legend

Supervisor: Prof. John C.S. Lui

Team Members: 1155066601 Feng Chen

1155066051 Wu Di

1155066626 Zhao Che

Overview

- Introduction
- Data Crawling
- Champion Matrix and Recommendation
- Champion Ability Ranking
- Champion Clustering and Match Prediction

Outline

- Introduction
- Data Crawling
- Champion Matrix and Recommendation
- Champion Ability Ranking
- Champion Clustering and Match Prediction

Objectives

- MOBA (multiplayer online battle arena): a quite new but popular game genre
- LoL (League of Legends): one of the most popular MOBA in the world: over 27 million daily players

- Data analysis on games
  - Insights on game design: game balance, champion positioning
  - Insights for gamers: how to pick champion, how to find champion suitable for a player

Introduction to League of Legends

- Goal: destroy the opposing team's building (base)
- Two opposing team: 5 v.s. 5, summoner (player) – champion (game character)
- Champion official position: *Fighter*, *Marksman*, *Mage*, *Tank*, *Assassin*, *Support*
- Damage and scores of kill, assist, death
- Earn money and level up by killing champion, building…
- Each match is discrete, kind of like ball games

Outline

- Introduction
- Data Crawling
- Champion Matrix and Recommendation
- Champion Ability Ranking
- Champion Clustering and Match Prediction

API and Target Data

- Riot REST API: free, URL query, return JSON data
- Cassiopeia: open source python wrapper of Riot API, return Python object data
- Target Matches: North American district, Pre-Season 2016 in different levels
- Seven database tables to remodel the crawled data

Crawling Strategy

(1) Randomly choose 3 summoners (players) with tier “Silver”, “Challenger” and “Diamond” separately, add them to an empty seed list;

(2) For each summoner in the seed list, use a processor to crawl all his historical matches into the datasets ;

(3) For each crawled much, obtain ten summoners’ ID. If the summoner is not crawled, add his ID to the seed list;

(4) Stop the corresponding processor when we obtain about 100,000 matches from the Silver seed, 60,000 from the Challenger seeds and 60,000 from the diamond seed;

(5) Repeat step (2);

(6) If all processors stopped, merge three crawled datasets into one.

Crawled Data

SQLite Database size: 2.1 GB

Table Records:

- *Summoner*: 487,484
- *Match*: 222,652
- *Team*: 445,304
- *TeamBan*: 1,330,757
- *Participant*: 2,226,520
- *ParticipantTimeLine*: 8,906,080
- *FrameKillEvent*: 21,692,852

Outline

- Introduction
- Data Crawling
- Champion Matrix and Recommendation
- Champion Ability Ranking
- Champion Clustering and Match Prediction

Preprocessing

- 1. Generate kill/assist matrix
  - Sum the kill/assist events and normalize them by $Total\_pick_i$
- 2. Player level separation
  - Divide players into 7 different levels (tiers) and 5 different versions

Preprocessing

- 1. Generate kill/assist matrix:
- Kill matrix demo

	Aatrox	Ahri	Akali	Alistar	Amumu
Aatrox	0	**232**	141	198	195
Ahri	314	0	761	1601	1032
Akali	151	942	0	352	316
Alistar	82	445	143	0	201
Amumu	177	754	274	601	0

Normalized Kill matrix demo

	Aatrox	Ahri	Akali	Alistar	Amumu
Aatrox	0	0.0478	0.0290	0.0408	0.0402
Ahri	0.0117	0	0.0285	0.0599	0.0386
Akali	0.0249	0.1551	0	0.0579	0.0520
Alistar	0.0034	0.0189	0.0061	0	0.0086
Amumu	0.0125	0.0534	0.0194	0.0425	0

$$\text{avg\_kill}_{ij} = \boldsymbol{K}_{ij}/Total\_pick_i$$

Preprocessing

- 2. Player level separation:

Tiers :	Challenger	Master	Diamond	Platinum	Gold	Silver	Bronze

Versions :	5.21	5.22	5.23	5.24	6.1

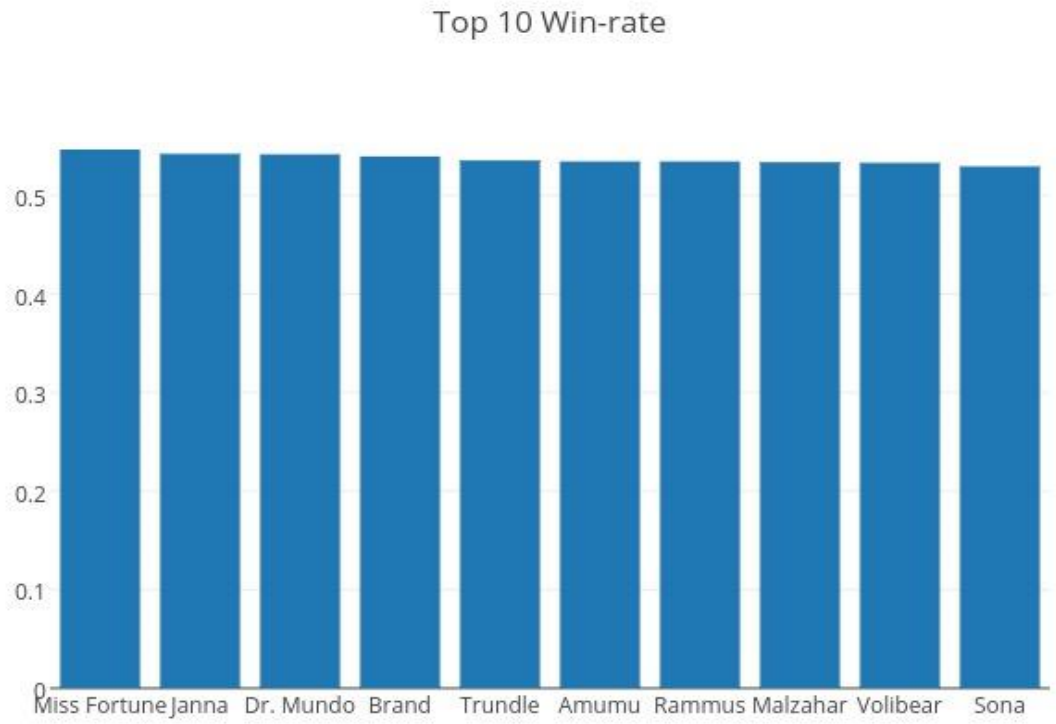
Introduction of Champion Recommendation

Time	Number of Champions
01/01/2016	128
05/01/2016	140

- 1. Recommend according to popularity and win rate
- 2. Recommend according to kill/assist matrices

Introduction of Champion Recommendation

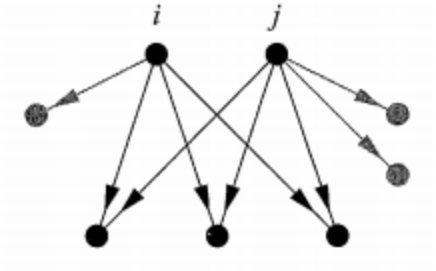
- 1. Recommend according to popularity and win rate:



Champion	win_rate
Miss Fortune	0.546508
Janna	0.542181
Dr. Mundo	0.541674
Brand	0.539527
Trundle	0.535452
Amumu	0.534467
Rammus	0.534403
Malzahar	0.533652
Volibear	0.533047
Sona	0.529421

Introduction of Champion Recommendation

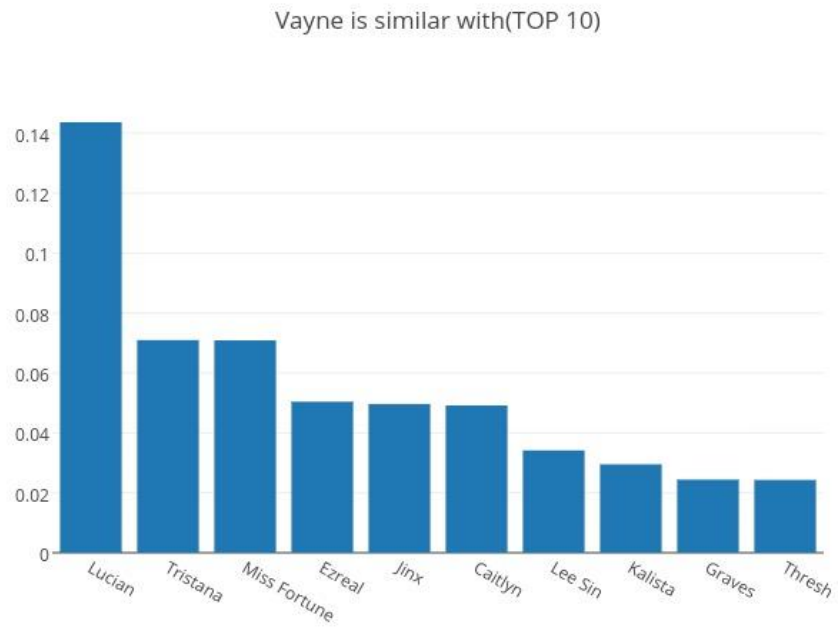
- **Bibliographic coupling matrix:**

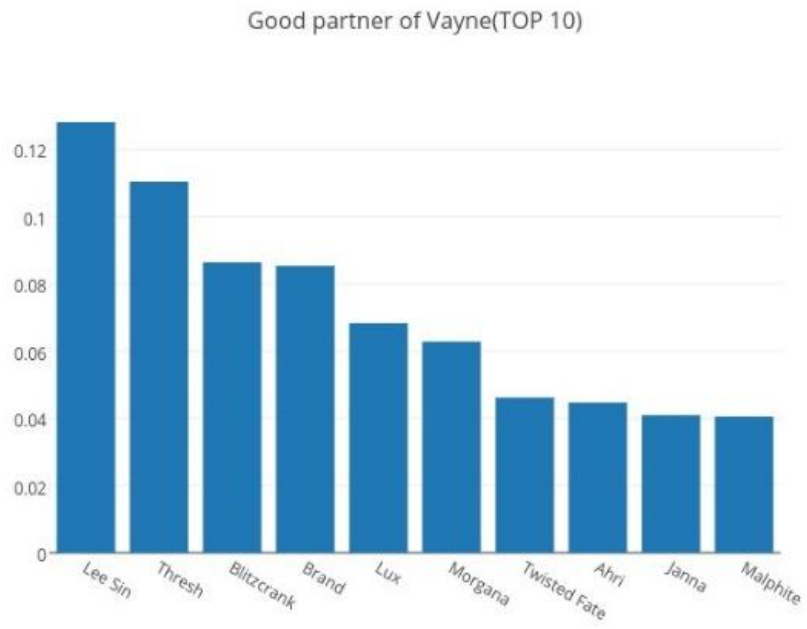


- Let A_{ij} represents the amount of times vertex i cites vertex j. The bibliographic coupling matrix, B, is defined as:
  - $\boldsymbol{B}_{ij} = \sum_{k=1}^{n} A_{ik}A_{jk} = \sum_{k=1}^{n} A_{ik}A_{kj}^{T}$
  - $\mathbf{B} = \mathbf{A}\boldsymbol{A}^{T}$
- While **A** is the adjacency matrix of the citation network.

Introduction of Champion Recommendation

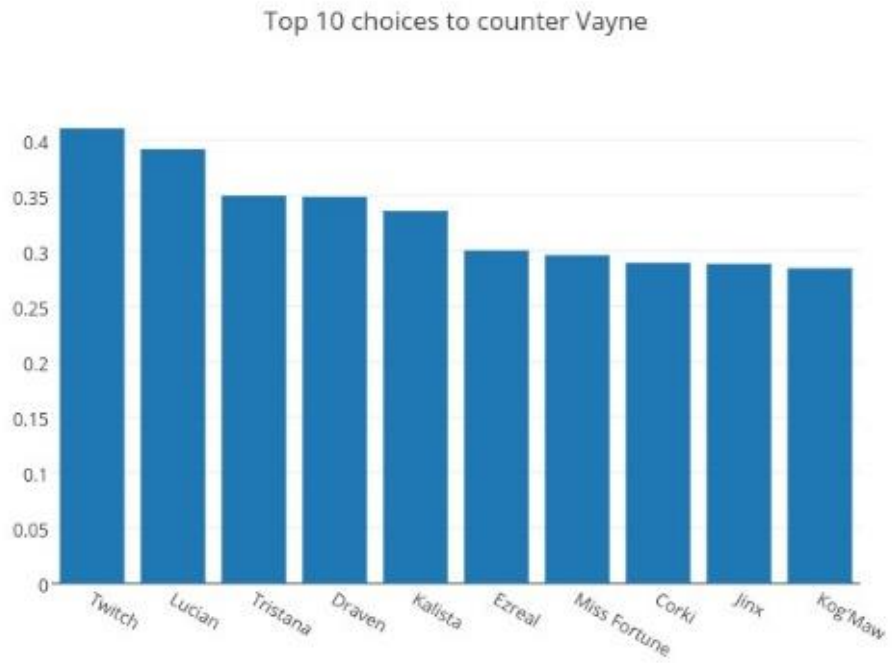
- 2. Recommend according to kill/assist matrices **K / A**:





Introduction of Champion Recommendation

- 2. Recommend according to kill/assist matrices:



Single Feature Win Rate Prediction

- 0-1 classification problem.
- Totally 31 features of each match.
- 220 thousand match – 80% for training, 20% for testing.
- Train models and select the best model and the best predict result.

Single Feature Win Rate Prediction

Feature type	Error Rate	Predict Rate
gold_earned	0.0267	0.9733
turret_kills	0.0427	0.9573
champion_level	0.0538	0.9462
kills	0.0668	0.9332
deaths	0.0673	0.9327
kda	0.0721	0.9279
damage_dealt	0.1028	0.8972
gold_spent	0.1043	0.8957
assists	0.1095	0.8905
largest_killing_spree	0.124	0.876
killing_sprees	0.131	0.869
damage_dealt_to_champions	0.131	0.869
largest_multi_kill	0.1916	0.8084
physical_damage_dealt	0.2307	0.7693
cs	0.248	0.752

Feature type	Error Rate	Predict Rate
physical_damage_dealt_to_champions	0.2543	0.7457
damage_taken	0.2582	0.7418
magic_damage_dealt_to_champions	0.3229	0.6771
magic_damage_taken	0.3334	0.6666
healing_done	0.3339	0.6661
largest_critical_strike	0.361	0.639
true_damage_dealt	0.3649	0.6351
magic_damage_dealt	0.3694	0.6306
wards_placed	0.3837	0.6163
crowd_control_dealt	0.3961	0.6039
true_damage_dealt_to_champions	0.4156	0.5844
true_damage_taken	0.4156	0.5844
physical_damage_taken	0.4221	0.5779
vision_wards_bought	0.4391	0.5609
units_healed	0.4567	0.5433
ward_kills	0.4758	0.5242

Outline

- Introduction
- Data Crawling
- Champion Matrix and Recommendation
- Champion Ability Ranking
- Champion Clustering and Match Prediction

Champion Ranking with Average Score

- Killing Ability Ranking

	Champion	Score	Position		Champion	Score	Position
Top 10	Akali	9.031286	Assassin	**Bottom 10**	Soraka	0.738921	Support, Mage
	Talon	8.934305	Assassin, Fighter		Janna	0.967073	Support, Mage
	Fizz	8.835916	Assassin		Braum	1.603554	Support, Tank
	Zed	8.432829	Assassin, Fighter		Nami	1.837762	Support, Mage
	Katarina	8.303200	Assassin, Mage		Alistar	1.851857	Tank, Support

Tops all have Assassin labels, while bottoms all have Support Labels.

Underperformed: Vi(55, F, A), Xin Zhao(60, F, A), Teemo(64, Mar, A)

Outperformed: Quinn(10, Mar, F), Corki(19, Mar)

Champion Ranking with Average Score

- Assisting Ability Ranking

	Champion	Score	Position		Champion	Score	Position
Top 5	Nami	15.268267	Support, Mage	**Bottom 5**	Master Yi	5.003692	Assassin, Fighter
	Janna	14.727506	Support, Mage		Tryndamere	5.052246	Fighter, Assassin
	Braum	14.124870	Support, Tank		Riven	5.103746	Fighter, Assassin
	Soraka	14.058027	Support, Mage		Jax	5.268868	Fighter, Assassin
	Thresh	13.811561	Support, Fighter		Fiora	5.318172	Fighter, Assassin

Tops all have Support labels, bottoms all have Assassin and Fighter Labels.

Underperformed: Kayle(75, F, S), Anivia(49, Ma, S)

Outperformed: Nautilus, (13, T, F), Rammus(18, T, F)

Graph Model of Champion Relationship

Champion Kill Matrix **K**:

- non-negative: $K_{ij} = k \geq 0$

  champion i kills j for k times in all matches

- asymmetric: $K_{ij} \neq K_{ji}$
- diagonal entries are all 0: $K_{ii} = 0$

  any champion can only be picked once in a match

K': **K** normalized by picks(i)

K'': **K** normalized by enemy_incidence(i,j)

Champion Assist Matrix **A: A, A'** and **A''**(normed by partner_incidence), with similar properties

Graph Model of Champion Relationship

- Relationship between each pair of champions
- Transitive assumption: if champion i kills j many times, j itself has high killing ability, then i also has high killing ability

- Eigenvector Centrality: **K'x** = w**x**
- find the eigenvector **x** corresponding to the largest eigenvalue w;

  x_i is the score of champion i's killing ability
- HITS: hub score of i in **K'** measures its killing ability

Champion Ranking with Graph Model

- Killing Ranking with Eigenvector Centrality of **K'**

	Champion	Score	Position		Champion	Score	Position
Top 5	Akali	0.137101	Assassin	**Bottom 5**	Soraka	0.010750	Support, Mage
	Talon	0.136831	Assassin, Fighter		Janna	0.014215	Support, Mage
	Fizz	0.133501	Assassin, Fighter		Braum	0.023332	Support, Tank
	Zed	0.129720	Assassin, Fighter		Nami	0.026539	Support, Mage
	Katarina	0.123811	Assassin, Mage		Alistar	0.027421	Tank, Support

No significant difference.

Champion Ranking with Graph Model

- Killing Ranking with Hub Centrality of **K'**

	Champion	Score	Position		Champion	Score	Position
	Talon	0.012690	Assassin, Fighter		Soraka	0.001080	Support, Mage
Top	Akali	0.012320	Assassin	**Bottom**	Janna	0.001501	Support, Mage
5	**Twitch**	0.012129	Marksman, Assassin	**5**	Braum	0.002402	Support, Tank
	Fizz	0.011963	Assassin, Fighter		Alistar	0.002754	Tank, Support
	Katarina	0.011612	Assassin, Mage		Nami	0.002773	Support, Mage

No significant difference: average score method is good enough

Champion Ranking with Graph Model

- Assisting Ranking with Eigenvector Centrality of **A'**

	Champion	Score	Position		Champion	Score	Position
	Nami	0.150016	Support, Mage		Master Yi	0.051729	Assassin, Fighter
Top	Janna	0.144490	Support, Mage	**Bottom**	Tryndamere	0.052331	Fighter, Assassin
5	Braum	0.138364	Support, Tank	**5**	Riven	0.053255	Fighter, Assassin
	Soraka	0.137169	Support, Mage		Jax	0.054587	Fighter, Assassin
	Thresh	0.135735	Support, Fighter		Fiora	0.055673	Fighter, Assassin

No significant difference.

Champion Ranking with Graph Model

- Assisting Ranking with Hub Centrality of **A'**

	Champion	Score	Position		Champion	Score	Position
Top 5	Nami	0.016397	Support, Mage	**Bottom 5**	**Vayne**	0.003561	Marksman, Assassin
	Janna	0.015918	Support, Mage		**Tristana**	0.003760	Marksman, Assassin
	Braum	0.015368	Support, Tank		**Draven**	0.003793	Marksman
	Thresh	0.014688	Support, Fighter		**Kalista**	0.003917	Marksman
	Soraka	0.014248	Support, Mage		**Lucian**	0.004176	Marksman

- Bottoms are quite different: Marksman instead of Fighter and Assassin.

Champion Ranking with Graph Model

- However, **K'** and **A'** gives us only biased result, because we ignore how the victim / assisted champion is picked in normalization
- Using **K'' and A''** is more reasonable, but Killing Ranking using eigenvector centrality with **K''** cannot get a stable solution in our experiments, and HITS cannot even converge
- So we try another method: PageRank

$x_i = \alpha \sum_j (K'')^T_{ij} x_j / k^{out}_j + \beta$

We use transpose of **K''** because i kills j means i has higher killing ability than j; α=0.85, β=0.15; similar for **A''**

Champion Ranking with Graph Model

- Killing Ranking with PageRank Centrality of **K”**

	Champion	Score	Position		Champion	Score	Position
Top 5	**Twisted Fate**	0.054610	Mage	**Bottom 5**	Soraka	0.001938	Support, Mage
	Akali	0.011221	Assassin		Janna	0.002134	Support, Mage
	Fizz	0.011182	Assassin, Fighter		Braum	0.002811	Support, Tank
	Talon	0.011080	Assassin, Fighter		Nami	0.003020	Support, Mage
	Zed	0.010720	Assassin, Fighter		**Alistar**	0.003111	Tank, Support

Twisted Fate is outstanding.

Champion Ranking with Graph Model

- Assisting Ranking with PageRank Centrality of **A''**

	Champion	Score	Position		Champion	Score	Position
Top 5	Nami	0.012476	Support, Mage	**Bottom 5**	Master Yi	0.005288	Assassin, Fighter
	Janna	0.011874	Support, Mage		Riven	0.005388	Fighter, Assassin
	Soraka	0.011514	Support, Mage		Tryndamere	0.005396	Fighter, Assassin
	Braum	0.011464	Support, Tank		Jax	0.005570	Fighter, Assassin
	Thresh	0.011218	Support, Fighter		Fiora	0.005572	Fighter, Assassin

No significant difference.

Champion Ability Ranking

- Some possible future Works in this part:
  - Deep analysis on exceptional results like non-convergence HITS
  - Combine the result from match data with static champion setting like health, skills, and growth curve (by level)
  - Comparison with the champion clustering results
  - Champion Popularity Ranking

Outline

- Introduction
- Data Crawling
- Champion Matrix and Recommendation
- Champion Ability Ranking
- Champion Clustering and Match Prediction

Champion Clustering

- Features

Match champion	Champions that involve in this match
Kills	Number of enemy champions killed
Deaths	Number of time champions die
Assists	Number of times assisting teammates to kill enemy champions
Gold earned	How much gold earned by champions
Magic damage	Magic damage dealt to enemy champions
Physical damage	Physical damage dealt to enemy champions
True damage	True damage dealt to enemy champions
Damage taken	Damage taken from enemy champions
Crowd control dealt	Sum of time of dealing control to enemy champions
Ward placed	Number of wards that have been placed
Ward kills	Number of wards that have been killed

Champion Clustering

- Algorithms
  - K-means

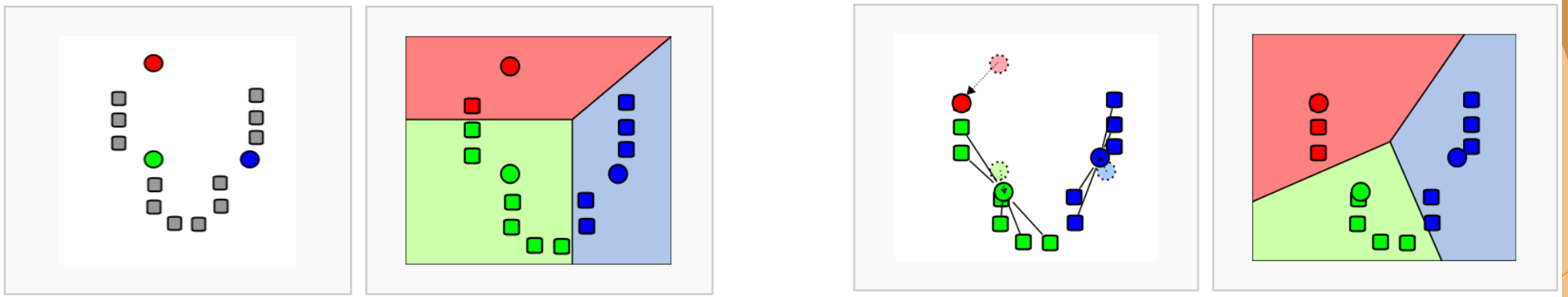
    Minimize within-cluster sum of squares

$$min \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2$$

1. Randomly initialize k centroids.
2. For each point, find the closest centroid and put this point into that cluster.
3. Re-calculate centroids.
4. Repeat step 2 and 3 until it converges.

Champion Clustering

- Algorithms
  - K-means



Champion Clustering

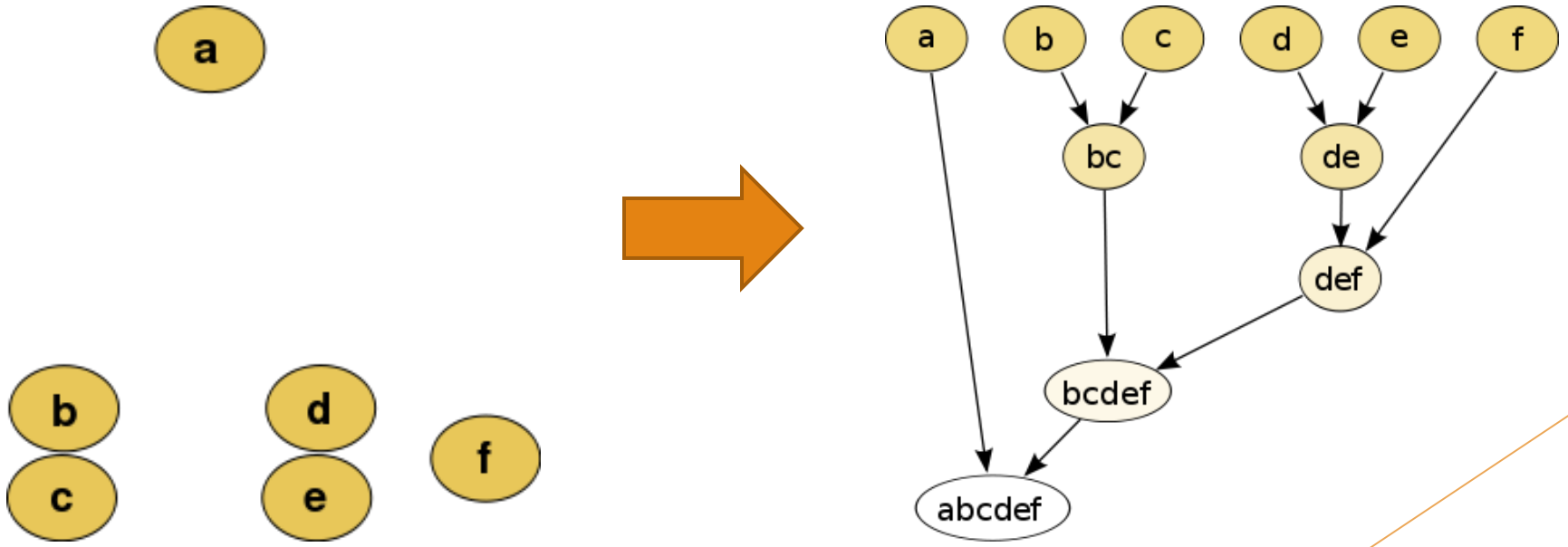
- Algorithms
  - Agglomerative

    Pair nearest points/clusters and finally obtain n clusters.

    1. Initialize with m points.
    2. For each point, find and be merged with the nearest cluster as a cluster.
    3. For each cluster, find and be merged with the nearest cluster as a new cluster.
    4. Repeat step 3 until there are exactly n clusters.

Champion Clustering

- Algorithms
  - Agglomerative



Champion Clustering

- Evaluation
  - Distortion: sum of distances between points and their centroid.

$$\text{Distortion} = \frac{\sum Euclidean(v, centroid)}{n}$$

  - Distance between clusters

$$\text{Distance} = \frac{\sum Euclidean(centroid_i, centroid_j)}{2m}$$

Champion Clustering

- Evaluation

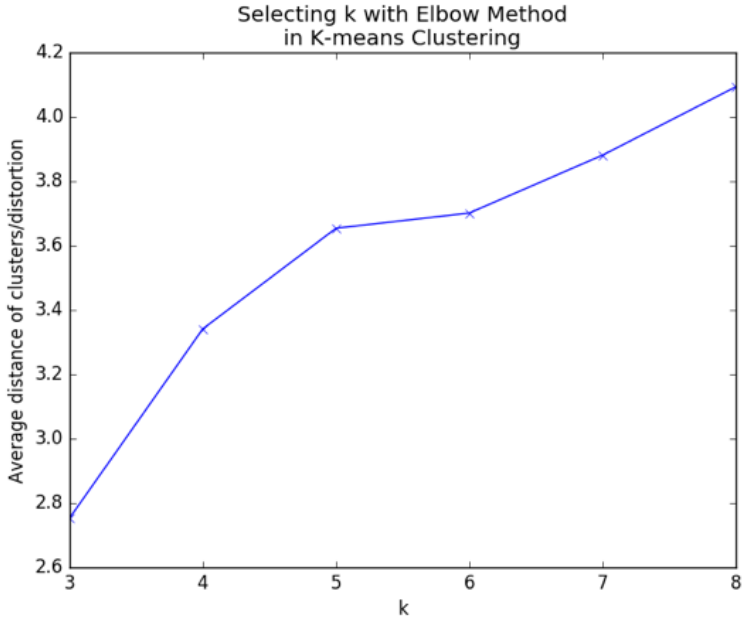
  Distortion and distance between clusters are both important. Low distortion and high distance between clusters are what we desire.

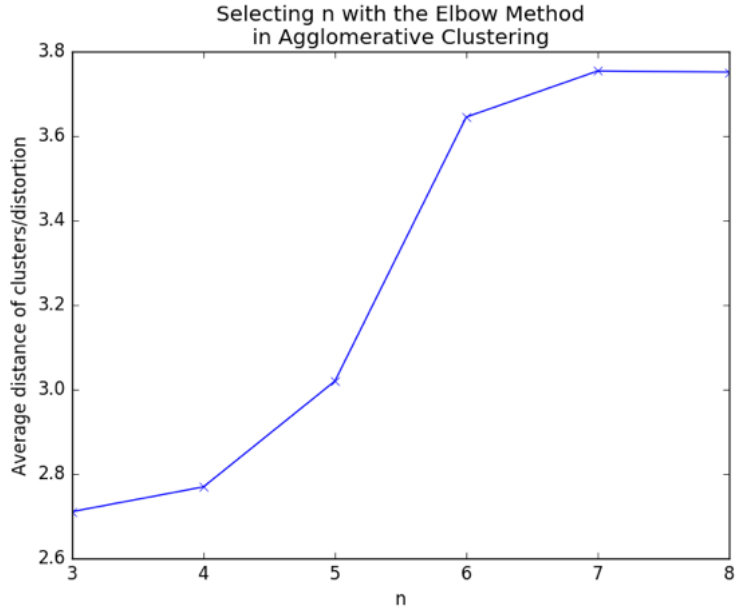
$$\mathrm{D} = \frac{distance}{distortion}$$

Champion Clustering

- Selecting k

  Elbow method: find k where D decreases the most and slows down the trend afterwards.





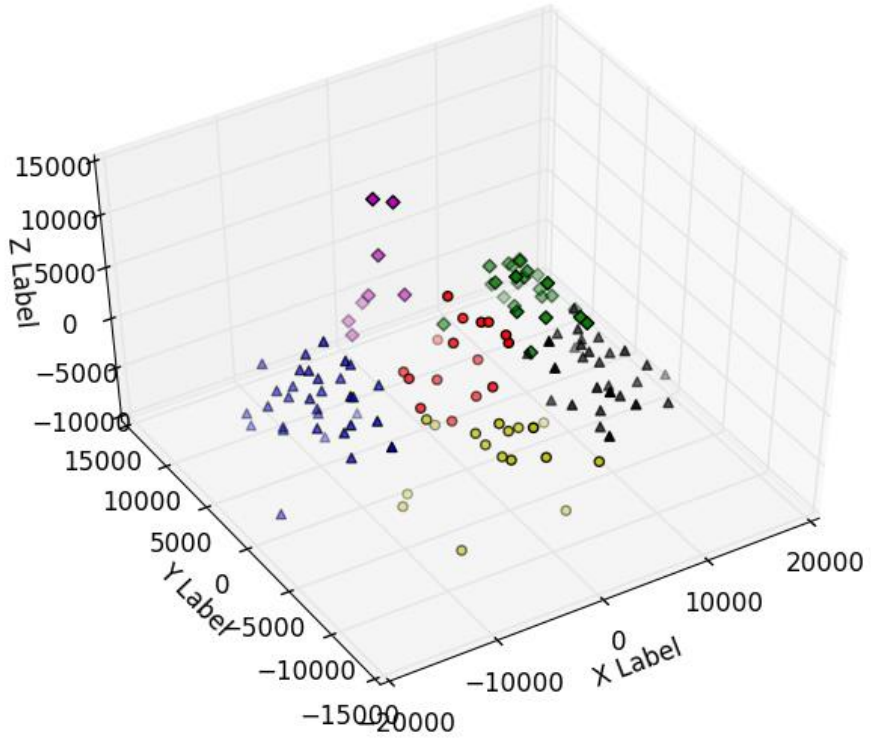
Champion Clustering

- Result

Index	Kills	Deaths	Assists	Gold	Magic damage	Physical damage	True damage	Damage taken	Control	Ward
1	4	5	8	11291	13340	2436	890	30760	685	11
2	5	5	6	11742	3515	11856	1366	29593	685	10
3	6	5	6	12328	2545	16919	846	20438	332	9
4	5	5	7	11801	19353	1624	716	18880	560	11
5	4	5	9	10684	9367	3626	682	24158	530	15
6	2	4	13	9864	8806	1765	216	16455	266	19

Champion Clustering

- Result illustration



Champion Clustering

- Interesting results
  - Fighters perform just like marksman.



  - Marksman is alike support character.

  - There are many more.

Win Prediction

1. Based on character composition

Feature is a vector representing two teams' character composition.

Cluster	1	2	3	4	5	6
Count	2	1	1	0	1	0

+

Cluster	1	2	3	4	5	6
Count	0	0	2	1	1	1

Label is win/lose of team 1.

Win Prediction

1. Based on character composition

Result is not satisfying. That means it's impossible to predict result before match happens with such features.

	Precision	Recall	F1-score
Lose	49%	50%	50%
Win	52%	49%	50%
Average	51%	49%	50%

To improve: include players' data.

Win Prediction

2. Based on match statistics
   - Single feature's result

Feature	Prediction precision
Gold difference	97%
Turret kills	95%
Levels	94%
Kills	93%
Deaths	93%
Assists	89%
Dragon kills	78%
Physical damage	74%
Damage taken	74%
Baron kills	80%
Magic damage	67%
Heal	66%
Wards placed	61%
Crowd control dealt	60%
True damage	58%
Ward kills	52%

Win Prediction

2. Based on match statistics
   - Included features

Damage	Physical damage, magic damage, true damage, damage taken
Buffs	Dragon kills, Baron kills
Wards	Number of wards placed, killed
Other	Heal, crowd control dealt

Win Prediction

2. Based on match statistics

This result's precision is higher than any of the included single features.

	Precision	Recall	F1-score
Lose	89%	90%	90%
Win	90%	89%	89%
Average	90%	90%	89%

Future work: use in-match statistics to predict trends, recommend strategies.

Thank you.