# 200 YEARS OF GAME DESIGN EXPERIENCE: A STUDY OF THE CHAMPIONS AND ITEMS IN LEAGUE OF LEGENDS

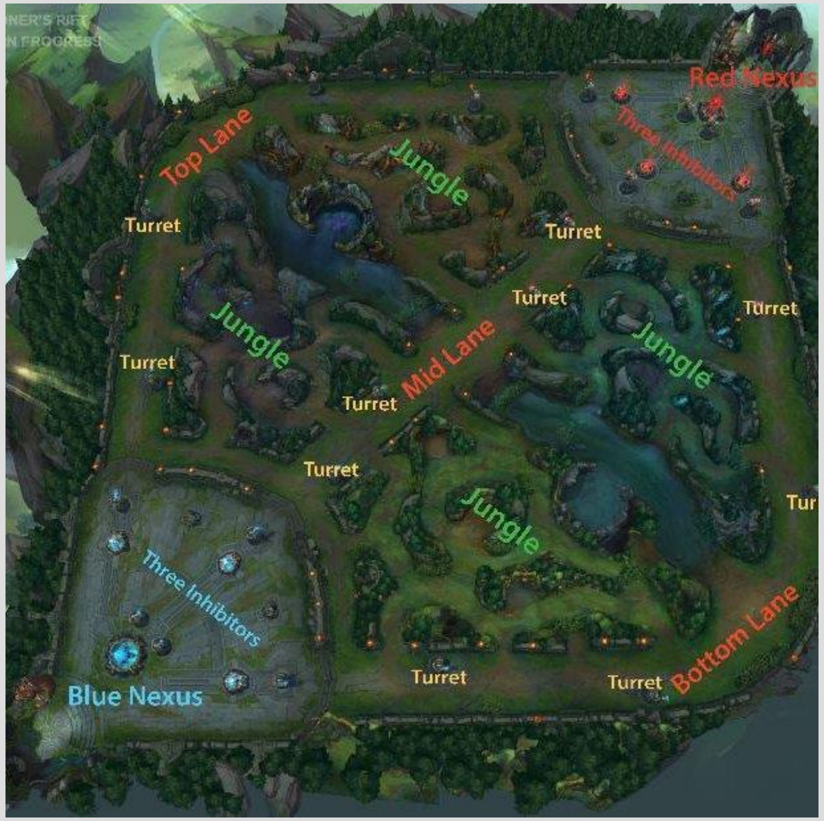
by Tobi Adewoye

## OVERVIEW

- League of Legends (2009) is a Multiplayer Online Battle Arena game. It has over 160 different playable characters, which are also known as champions, and about a hundred items that players can buy in-game to increase their power levels.
- As a result of the huge variance in games as a result of all the options, what is and isn't strong can be debatable.
- To try and figure out by what metrics power can be measured in this game, I analyzed the League of Legends 2022 professional season and used the data to compare and contrast champions and items based on age, gold efficiency, win rates, and other metrics.

# OVERVIEW

# LANES

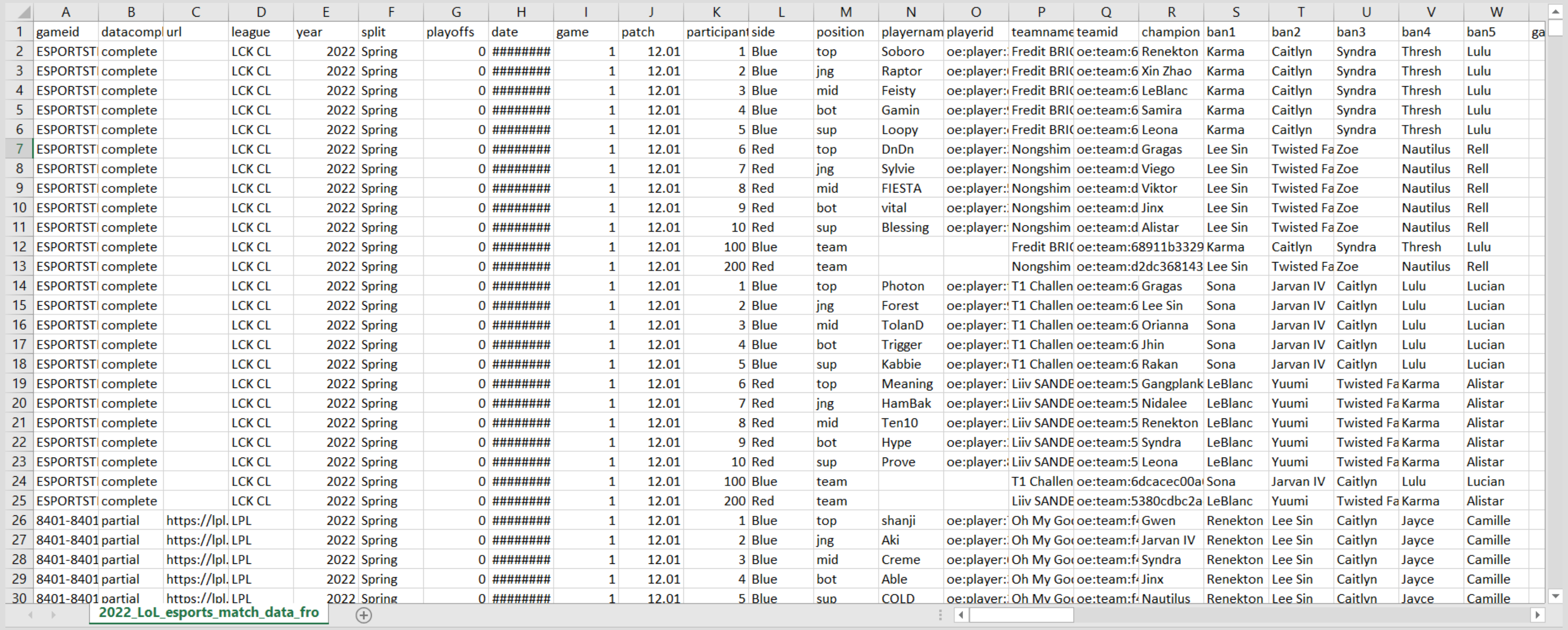

Red Nexus
Three Inhibitors
Top Lane
Jungle
Turret
Turret
Turret
Turret
Jungle
Mid Lane
Jungle
Turret
Turret
Turret
Jungle
Three Inhibitors
Blue Nexus
Turret
Turret
Bottom Lane

## CLASSES

- Assassin – Builds armor penetration and cooldown reduction items to assassinate weaker champions
- Fighter – Builds health and attack damage to deal damage over time while sustaining themselves in fights
- Mage – Builds ability power and cooldown reduction items to deal heavy damage
- Marksman – Builds attack speed and critical strike modifiers to deal heavy damage over time
- Support – Builds cooldown reduction to heal and shield allies
- Tank – Builds health and resists to act as a shield for the rest of the team to hide behind

# DATASET #1 – PROPLAY DATA

- This dataset contains information on every proplay game played in 2022.
- 149,233 rows, 123 columns

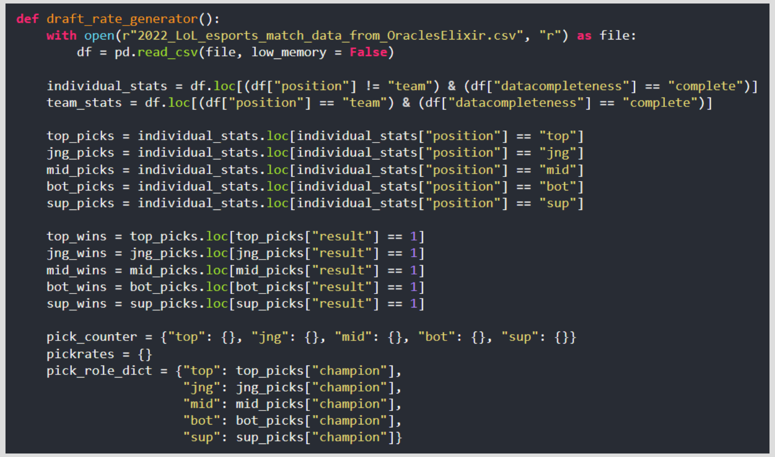


| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | gameid | datacompl | url | league | year | split | playoffs | date | game | patch | participant | side | position | playernam | playerid | teamname | teamid | champion | ban1 | ban2 | ban3 | ban4 | ban5 | ga |
| 2 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 1 | Blue | top | Soboro | oe:player: | Fredit BRI | oe:team:6 | Renekton | Karma | Caitlyn | Syndra | Thresh | Lulu | |
| 3 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 2 | Blue | jng | Raptor | oe:player: | Fredit BRI | oe:team:6 | Xin Zhao | Karma | Caitlyn | Syndra | Thresh | Lulu | |
| 4 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 3 | Blue | mid | Feisty | oe:player: | Fredit BRI | oe:team:6 | LeBlanc | Karma | Caitlyn | Syndra | Thresh | Lulu | |
| 5 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 4 | Blue | bot | Gamin | oe:player: | Fredit BRI | oe:team:6 | Samira | Karma | Caitlyn | Syndra | Thresh | Lulu | |
| 6 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 5 | Blue | sup | Loopy | oe:player: | Fredit BRI | oe:team:6 | Leona | Karma | Caitlyn | Syndra | Thresh | Lulu | |
| 7 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 6 | Red | top | DnDn | oe:player: | Nongshim | oe:team:d | Gragas | Lee Sin | Twisted Fa | Zoe | Nautilus | Rell | |
| 8 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 7 | Red | jng | Sylvie | oe:player: | Nongshim | oe:team:d | Viego | Lee Sin | Twisted Fa | Zoe | Nautilus | Rell | |
| 9 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 8 | Red | mid | FIESTA | oe:player: | Nongshim | oe:team:d | Viktor | Lee Sin | Twisted Fa | Zoe | Nautilus | Rell | |
| 10 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 9 | Red | bot | vital | oe:player: | Nongshim | oe:team:d | Jinx | Lee Sin | Twisted Fa | Zoe | Nautilus | Rell | |
| 11 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 10 | Red | sup | Blessing | oe:player: | Nongshim | oe:team:d | Alistar | Lee Sin | Twisted Fa | Zoe | Nautilus | Rell | |
| 12 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 100 | Blue | team | | | Fredit BRI | oe:team:68911b3329 | | Karma | Caitlyn | Syndra | Thresh | Lulu | |
| 13 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 200 | Red | team | | | Nongshim | oe:team:d2dc368143 | | Lee Sin | Twisted Fa | Zoe | Nautilus | Rell | |
| 14 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 1 | Blue | top | Photon | oe:player: | T1 Challen | oe:team:6 | Gragas | Sona | Jarvan IV | Caitlyn | Lulu | Lucian | |
| 15 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 2 | Blue | jng | Forest | oe:player: | T1 Challen | oe:team:6 | Lee Sin | Sona | Jarvan IV | Caitlyn | Lulu | Lucian | |
| 16 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 3 | Blue | mid | TolanD | oe:player: | T1 Challen | oe:team:6 | Orianna | Sona | Jarvan IV | Caitlyn | Lulu | Lucian | |
| 17 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 4 | Blue | bot | Trigger | oe:player: | T1 Challen | oe:team:6 | Jhin | Sona | Jarvan IV | Caitlyn | Lulu | Lucian | |
| 18 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 5 | Blue | sup | Kabbie | oe:player: | T1 Challen | oe:team:6 | Rakan | Sona | Jarvan IV | Caitlyn | Lulu | Lucian | |
| 19 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 6 | Red | top | Meaning | oe:player: | Liiv SANDE | oe:team:5 | Gangplank | LeBlanc | Yuumi | Twisted Fa | Karma | Alistar | |
| 20 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 7 | Red | jng | HamBak | oe:player: | Liiv SANDE | oe:team:5 | Nidalee | LeBlanc | Yuumi | Twisted Fa | Karma | Alistar | |
| 21 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 8 | Red | mid | Ten10 | oe:player: | Liiv SANDE | oe:team:5 | Renekton | LeBlanc | Yuumi | Twisted Fa | Karma | Alistar | |
| 22 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 9 | Red | bot | Hype | oe:player: | Liiv SANDE | oe:team:5 | Syndra | LeBlanc | Yuumi | Twisted Fa | Karma | Alistar | |
| 23 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 10 | Red | sup | Prove | oe:player: | Liiv SANDE | oe:team:5 | Leona | LeBlanc | Yuumi | Twisted Fa | Karma | Alistar | |
| 24 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 100 | Blue | team | | | T1 Challen | oe:team:6dcacec00a | | Sona | Jarvan IV | Caitlyn | Lulu | Lucian | |
| 25 | ESPORTSTI | complete | | LCK CL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 200 | Red | team | | | Liiv SANDE | oe:team:5380cdbc2a | | LeBlanc | Yuumi | Twisted Fa | Karma | Alistar | |
| 26 | 8401-8401 | partial | https://lpl. | LPL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 1 | Blue | top | shanji | oe:player: | Oh My Go | oe:team:f | Gwen | Renekton | Lee Sin | Caitlyn | Jayce | Camille | |
| 27 | 8401-8401 | partial | https://lpl. | LPL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 2 | Blue | jng | Aki | oe:player: | Oh My Go | oe:team:f | Jarvan IV | Renekton | Lee Sin | Caitlyn | Jayce | Camille | |
| 28 | 8401-8401 | partial | https://lpl. | LPL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 3 | Blue | mid | Creme | oe:player: | Oh My Go | oe:team:f | Syndra | Renekton | Lee Sin | Caitlyn | Jayce | Camille | |
| 29 | 8401-8401 | partial | https://lpl. | LPL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 4 | Blue | bot | Able | oe:player: | Oh My Go | oe:team:f | Jinx | Renekton | Lee Sin | Caitlyn | Jayce | Camille | |
| 30 | 8401-8401 | partial | https://lpl. | LPL | 2022 | Spring | 0 | ######## | 1 | 12.01 | 5 | Blue | sup | COLD | oe:player: | Oh My Go | oe:team:f | Nautilus | Renekton | Lee Sin | Caitlyn | Jayce | Camille | |

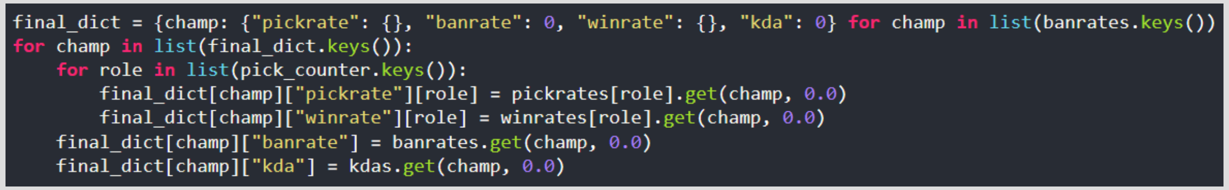
2022_LoL_esports_match_data_fro

# DATASET #1 – PROPLAY DATA

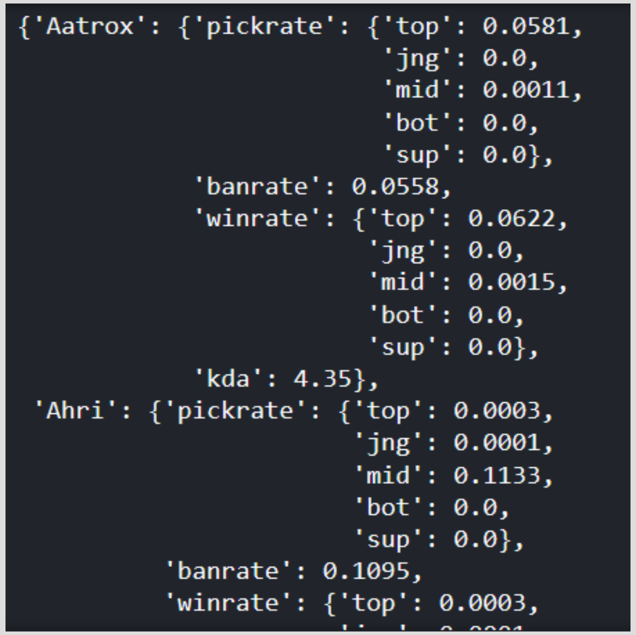
- I cleaned the data with pandas.
- The cleaned data contains information on each champion’s pick rate per role, ban rate, win rate per role, and Kill/Death/Assist ratio.



```python
def draft_rate_generator():
    with open(r"2022_LoL_esports_match_data_from_OraclesElixir.csv", "r") as file:
        df = pd.read_csv(file, low_memory = False)

    individual_stats = df.loc[(df["position"] != "team") & (df["datacompleteness"] == "complete")]
    team_stats = df.loc[(df["position"] == "team") & (df["datacompleteness"] == "complete")]

    top_picks = individual_stats.loc[individual_stats["position"] == "top"]
    jng_picks = individual_stats.loc[individual_stats["position"] == "jng"]
    mid_picks = individual_stats.loc[individual_stats["position"] == "mid"]
    bot_picks = individual_stats.loc[individual_stats["position"] == "bot"]
    sup_picks = individual_stats.loc[individual_stats["position"] == "sup"]

    top_wins = top_picks.loc[top_picks["result"] == 1]
    jng_wins = jng_picks.loc[jng_picks["result"] == 1]
    mid_wins = mid_picks.loc[mid_picks["result"] == 1]
    bot_wins = bot_picks.loc[bot_picks["result"] == 1]
    sup_wins = sup_picks.loc[sup_picks["result"] == 1]

    pick_counter = {"top": {}, "jng": {}, "mid": {}, "bot": {}, "sup": {}}
    pickrates = {}
    pick_role_dict = {"top": top_picks["champion"],
                      "jng": jng_picks["champion"],
                      "mid": mid_picks["champion"],
                      "bot": bot_picks["champion"],
                      "sup": sup_picks["champion"]}
```



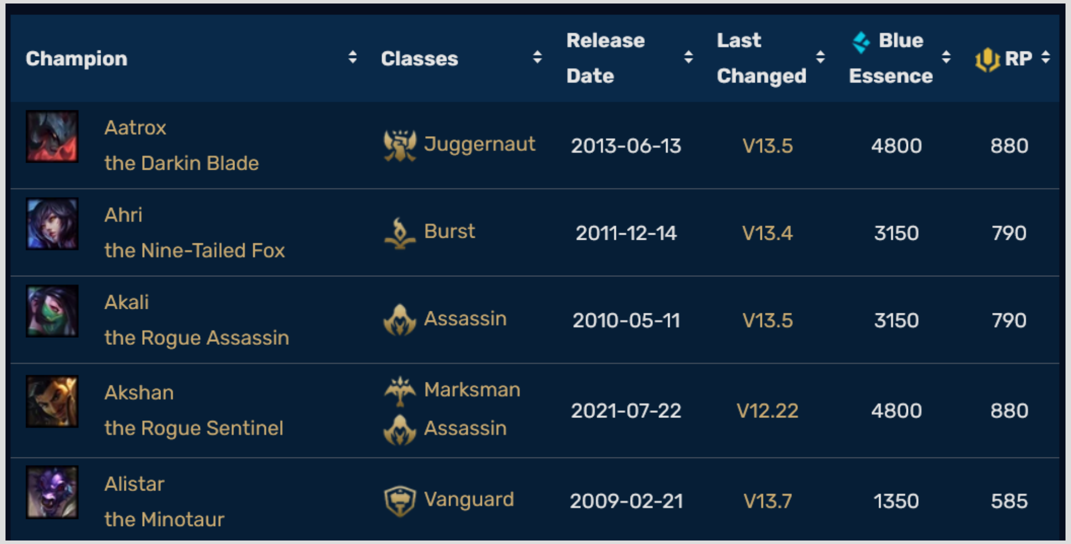
```python
final_dict = {champ: {"pickrate": {}, "banrate": 0, "winrate": {}, "kda": 0} for champ in list(banrates.keys())
for champ in list(final_dict.keys()):
    for role in list(pick_counter.keys()):
        final_dict[champ]["pickrate"][role] = pickrates[role].get(champ, 0.0)
        final_dict[champ]["winrate"][role] = winrates[role].get(champ, 0.0)
    final_dict[champ]["banrate"] = banrates.get(champ, 0.0)
    final_dict[champ]["kda"] = kdas.get(champ, 0.0)
```



```
{'Aatrox': {'pickrate': {'top': 0.0581,
                         'jng': 0.0,
                         'mid': 0.0011,
                         'bot': 0.0,
                         'sup': 0.0},
            'banrate': 0.0558,
            'winrate': {'top': 0.0622,
                        'jng': 0.0,
                        'mid': 0.0015,
                        'bot': 0.0,
                        'sup': 0.0},
            'kda': 4.35},
 'Ahri': {'pickrate': {'top': 0.0003,
                       'jng': 0.0001,
                       'mid': 0.1133,
                       'bot': 0.0,
                       'sup': 0.0},
          'banrate': 0.1095,
          'winrate': {'top': 0.0003,
```

# DATASET #2 – CHAMPION INFORMATION

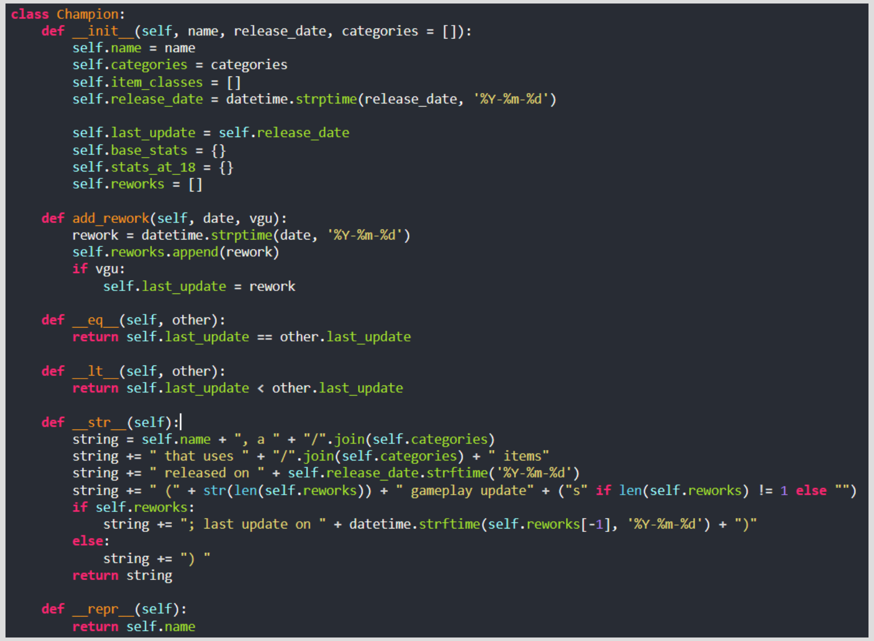
- This dataset contains information on all the champions in the game scraped from the League of Legends wiki using BeautifulSoup.
- Because some champions have received major reworks since they released, I judged a champion's age based on the date of their last major gameplay rework if they have one.



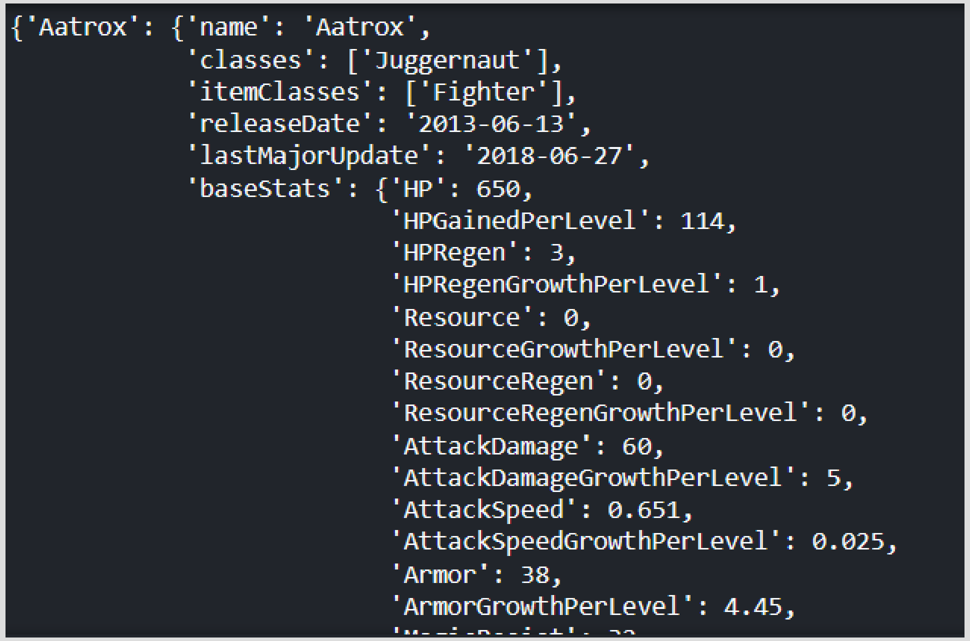
| Champion | Classes | Release Date | Last Changed | Blue Essence | RP |
|---|---|---|---|---|---|
| Aatrox the Darkin Blade | Juggernaut | 2013-06-13 | V13.5 | 4800 | 880 |
| Ahri the Nine-Tailed Fox | Burst | 2011-12-14 | V13.4 | 3150 | 790 |
| Akali the Rogue Assassin | Assassin | 2010-05-11 | V13.5 | 3150 | 790 |
| Akshan the Rogue Sentinel | Marksman Assassin | 2021-07-22 | V12.22 | 4800 | 880 |
| Alistar the Minotaur | Vanguard | 2009-02-21 | V13.7 | 1350 | 585 |

# DATASET #2 – CHAMPION INFORMATION

- The cleaned dataset was created using list and dictionary comprehensions, coupled with a custom class. The data was stored in a json file.



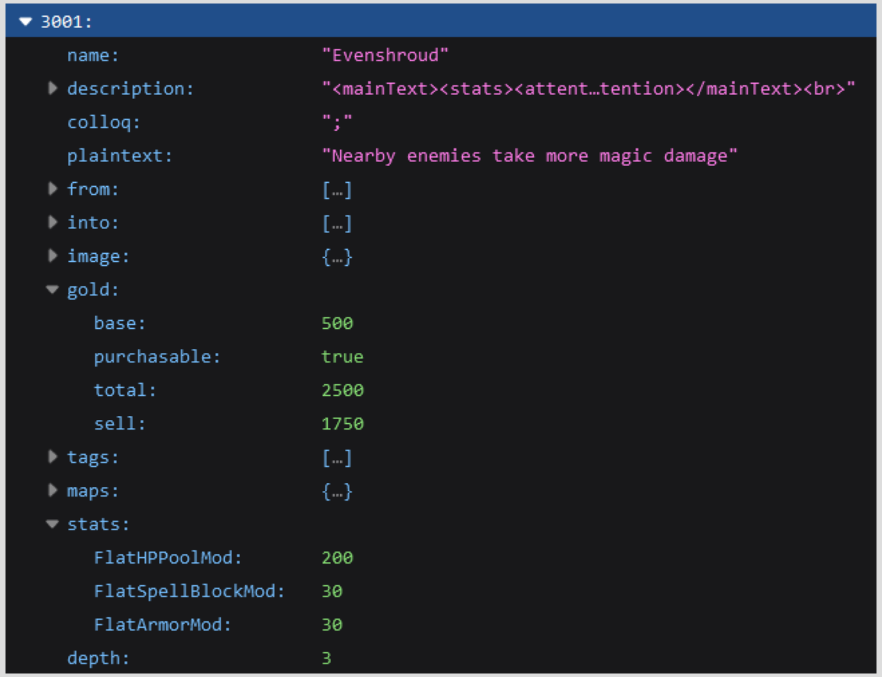
```python
class Champion:
    def __init__(self, name, release_date, categories = []):
        self.name = name
        self.categories = categories
        self.item_classes = []
        self.release_date = datetime.strptime(release_date, '%Y-%m-%d')

        self.last_update = self.release_date
        self.base_stats = {}
        self.stats_at_18 = {}
        self.reworks = []

    def add_rework(self, date, vgu):
        rework = datetime.strptime(date, '%Y-%m-%d')
        self.reworks.append(rework)
        if vgu:
            self.last_update = rework

    def __eq__(self, other):
        return self.last_update == other.last_update

    def __lt__(self, other):
        return self.last_update < other.last_update

    def __str__(self):
        string = self.name + ", a " + "/".join(self.categories)
        string += " that uses " + "/".join(self.categories) + " items"
        string += " released on " + self.release_date.strftime('%Y-%m-%d')
        string += " (" + str(len(self.reworks)) + " gameplay update" + ("s" if len(self.reworks) != 1 else "")
        if self.reworks:
            string += "; last update on " + datetime.strftime(self.reworks[-1], '%Y-%m-%d') + ")"
        else:
            string += ") "
        return string

    def __repr__(self):
        return self.name
```



```
{'Aatrox': {'name': 'Aatrox',
            'classes': ['Juggernaut'],
            'itemClasses': ['Fighter'],
            'releaseDate': '2013-06-13',
            'lastMajorUpdate': '2018-06-27',
            'baseStats': {'HP': 650,
                          'HPGainedPerLevel': 114,
                          'HPRegen': 3,
                          'HPRegenGrowthPerLevel': 1,
                          'Resource': 0,
                          'ResourceGrowthPerLevel': 0,
                          'ResourceRegen': 0,
                          'ResourceRegenGrowthPerLevel': 0,
                          'AttackDamage': 60,
                          'AttackDamageGrowthPerLevel': 5,
                          'AttackSpeed': 0.651,
                          'AttackSpeedGrowthPerLevel': 0.025,
                          'Armor': 38,
                          'ArmorGrowthPerLevel': 4.45,
```

## DATASET #3 – ITEM INFORMATION

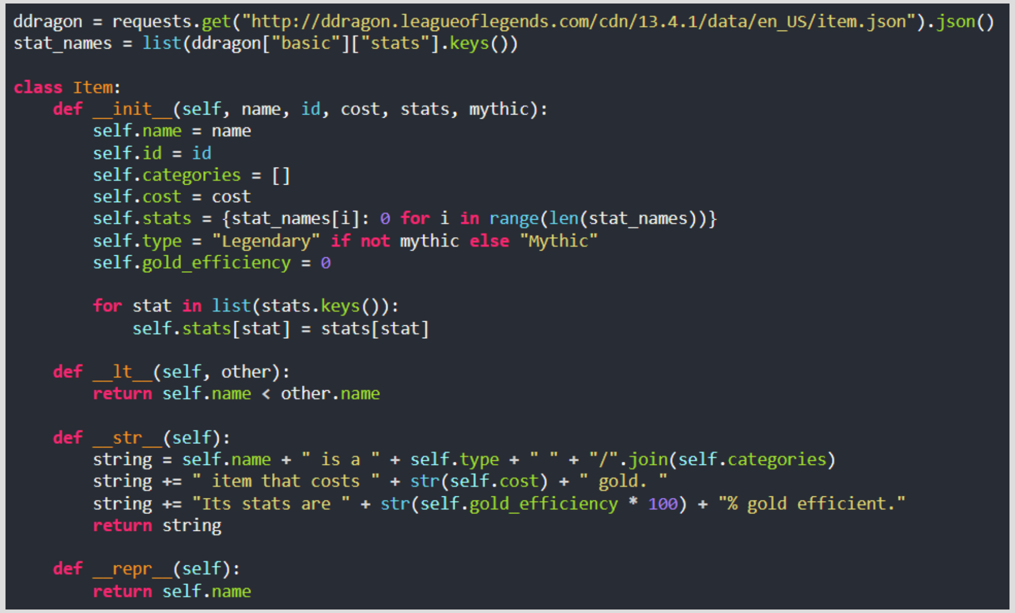
- The dataset contains item statistics, and was obtained using my API key for the League of Legends API with the requests module. I coupled this dataset with information on the gold efficiency of each item, which was scraped from the wiki.



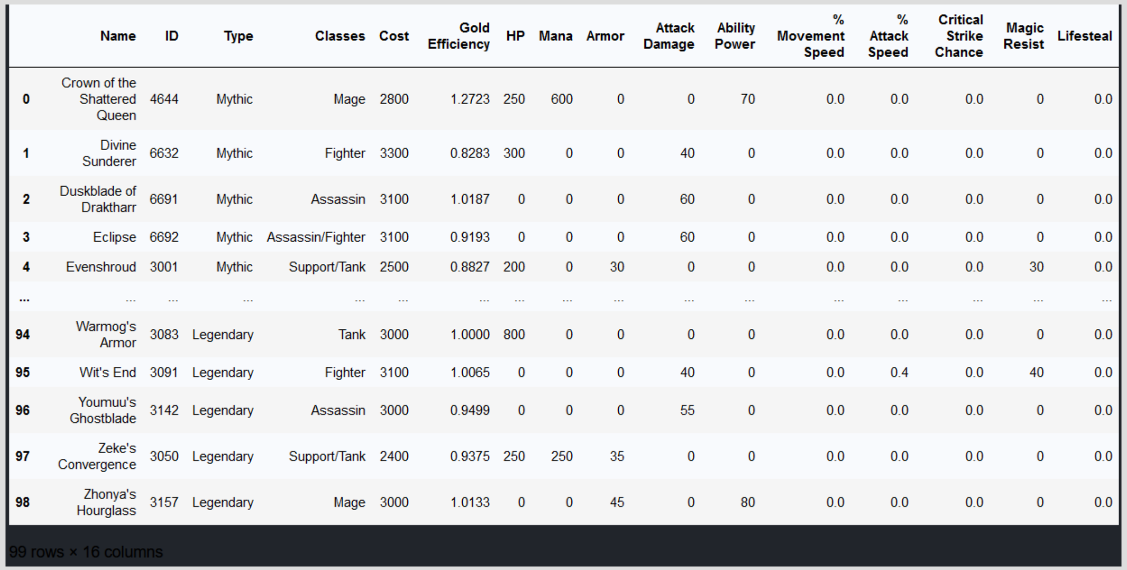
```
▼ 3001:
    name:                "Evenshroud"
  ▶ description:         "<mainText><stats><attent…tention></mainText><br>"
    colloq:              ";"
    plaintext:           "Nearby enemies take more magic damage"
  ▶ from:                […]
  ▶ into:                […]
  ▶ image:               {…}
  ▼ gold:
      base:              500
      purchasable:       true
      total:             2500
      sell:              1750
  ▶ tags:                […]
  ▶ maps:                {…}
  ▼ stats:
      FlatHPPoolMod:     200
      FlatSpellBlockMod: 30
      FlatArmorMod:      30
    depth:               3
```

# DATASET #3 – ITEM INFORMATION

- The cleaned dataset was created using a custom class and list and dictionary comprehensions. I had to specially exclude items outside of the scope of the project, and get rid of irrelevant stats.



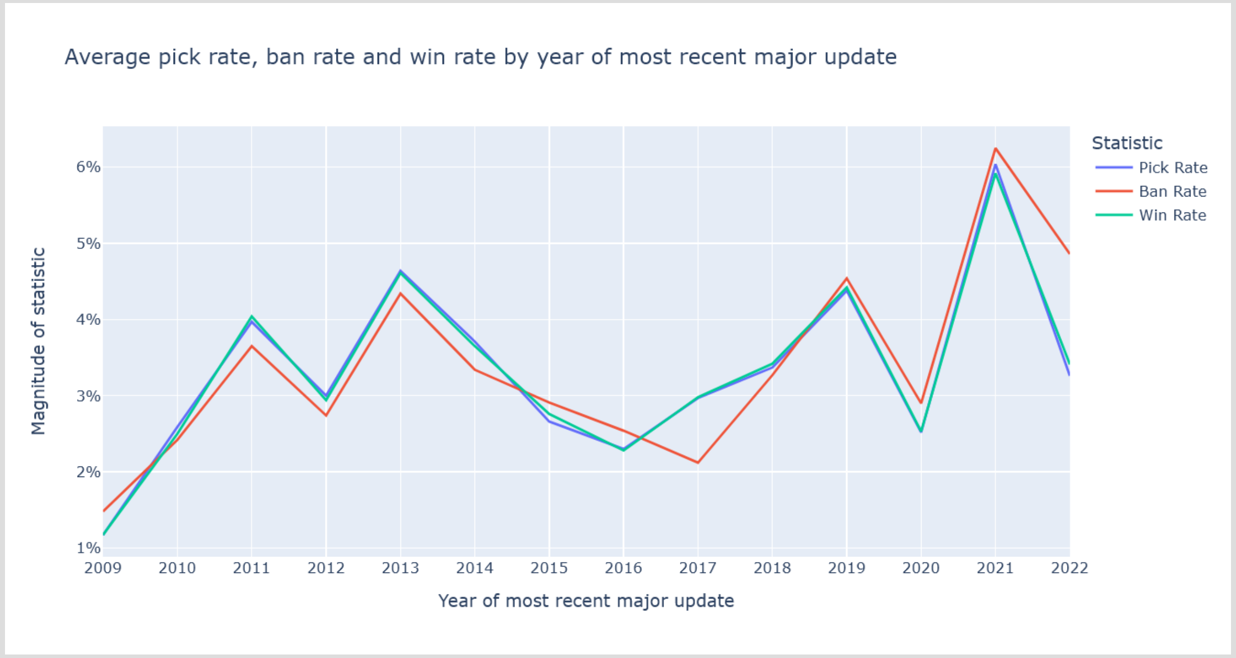
```python
ddragon = requests.get("http://ddragon.leagueoflegends.com/cdn/13.4.1/data/en_US/item.json").json()
stat_names = list(ddragon["basic"]["stats"].keys())

class Item:
    def __init__(self, name, id, cost, stats, mythic):
        self.name = name
        self.id = id
        self.categories = []
        self.cost = cost
        self.stats = {stat_names[i]: 0 for i in range(len(stat_names))}
        self.type = "Legendary" if not mythic else "Mythic"
        self.gold_efficiency = 0

        for stat in list(stats.keys()):
            self.stats[stat] = stats[stat]

    def __lt__(self, other):
        return self.name < other.name

    def __str__(self):
        string = self.name + " is a " + self.type + " " + "/".join(self.categories)
        string += " item that costs " + str(self.cost) + " gold. "
        string += "Its stats are " + str(self.gold_efficiency * 100) + "% gold efficient."
        return string

    def __repr__(self):
        return self.name
```



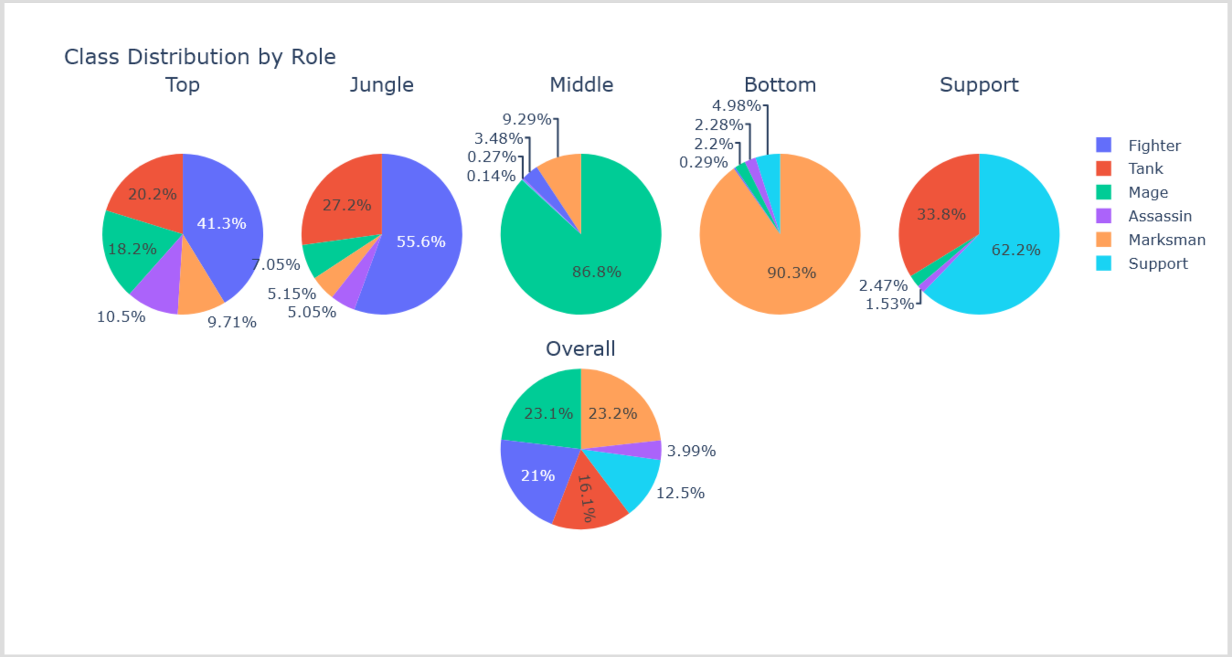
| | Name | ID | Type | Classes | Cost | Gold Efficiency | HP | Mana | Armor | Attack Damage | Ability Power | % Movement Speed | % Attack Speed | Critical Strike Chance | Magic Resist | Lifesteal |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Crown of the Shattered Queen | 4644 | Mythic | Mage | 2800 | 1.2723 | 250 | 600 | 0 | 0 | 70 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **1** | Divine Sunderer | 6632 | Mythic | Fighter | 3300 | 0.8283 | 300 | 0 | 0 | 40 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **2** | Duskblade of Draktharr | 6691 | Mythic | Assassin | 3100 | 1.0187 | 0 | 0 | 0 | 60 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **3** | Eclipse | 6692 | Mythic | Assassin/Fighter | 3100 | 0.9193 | 0 | 0 | 0 | 60 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **4** | Evenshroud | 3001 | Mythic | Support/Tank | 2500 | 0.8827 | 200 | 0 | 30 | 0 | 0 | 0.0 | 0.0 | 0.0 | 30 | 0.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **94** | Warmog's Armor | 3083 | Legendary | Tank | 3000 | 1.0000 | 800 | 0 | 0 | 0 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **95** | Wit's End | 3091 | Legendary | Fighter | 3100 | 1.0065 | 0 | 0 | 0 | 40 | 0 | 0.0 | 0.4 | 0.0 | 40 | 0.0 |
| **96** | Youmuu's Ghostblade | 3142 | Legendary | Assassin | 3000 | 0.9499 | 0 | 0 | 0 | 55 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **97** | Zeke's Convergence | 3050 | Legendary | Support/Tank | 2400 | 0.9375 | 250 | 250 | 35 | 0 | 0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |
| **98** | Zhonya's Hourglass | 3157 | Legendary | Mage | 3000 | 1.0133 | 0 | 0 | 45 | 0 | 80 | 0.0 | 0.0 | 0.0 | 0 | 0.0 |

99 rows × 16 columns

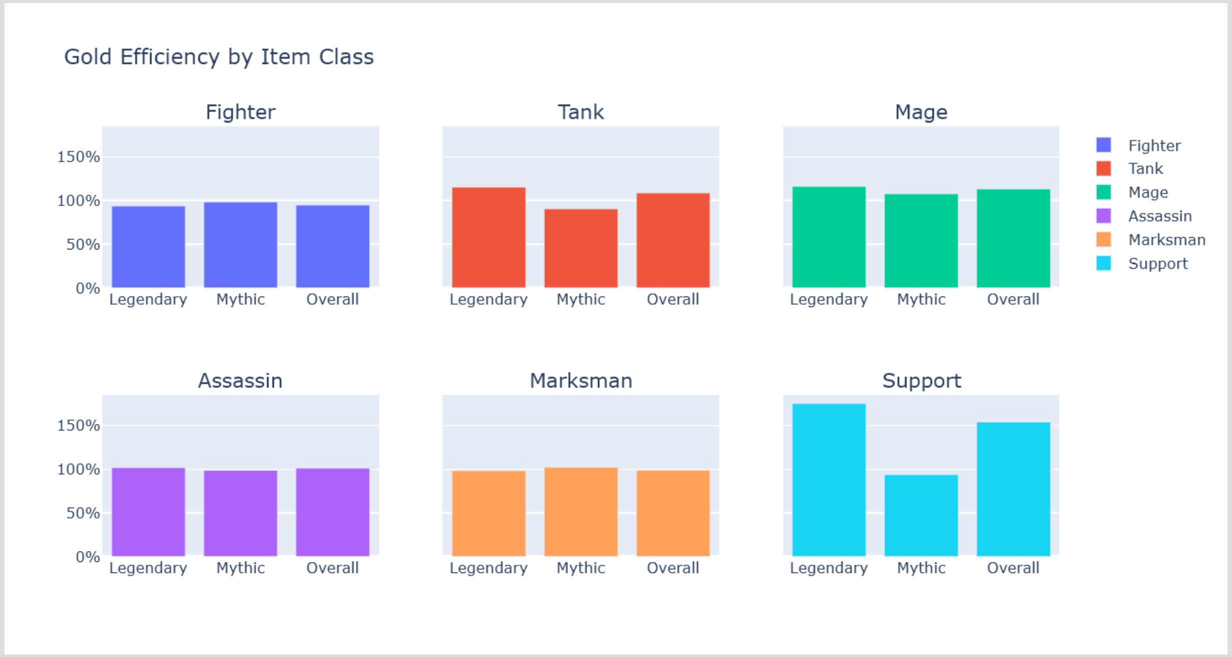
# INSIGHT #1 – AGE VS PICK/BAN/WIN RATES



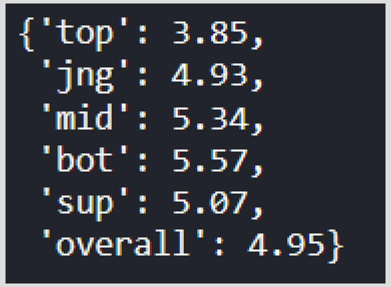
# INSIGHT #2 – CLASS DISTRIBUTION BY ROLE



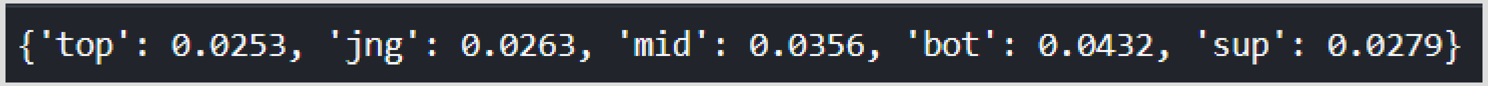
# INSIGHT #3 – GOLD EFFICIENCY BY ITEM CLASS



## INSIGHTS #4 AND #5 – ROLES AND THEIR AVERAGE K/D/A AND BANRATES



```
{'top': 3.85,
 'jng': 4.93,
 'mid': 5.34,
 'bot': 5.57,
 'sup': 5.07,
 'overall': 4.95}
```



```
{'top': 0.0253, 'jng': 0.0263, 'mid': 0.0356, 'bot': 0.0432, 'sup': 0.0279}
```

## RESULTS AND CONCLUSION

- Because of how multifaceted the game is, it is difficult to point to any single metric and say that it can serve as a way of measuring the balance of anything in the game.
- However, there is a slight implication that newer champions are somewhat stronger than older ones.

## CHALLENGES

- I had a tough time using pandas and Python’s built-in OOP functionality, because I still don’t 100% understand how they work.
- I think embarking on this project really helped me get a better understanding of data manipulation.

## THANK YOU!