Are Tanks Really Overpowered Chat?

Dylan Armbruster

Table of contents

Introduction

Within the multi-player online platform League of Legends, champions are divided into distinct classes based upon abilities and purpose. One of those classes, Tanks, is perhaps the most likely class to bear community criticism and design scrutiny. Many players express frustration with playing against them. Citing that despite having diminished offense, they carry high starting endurance values which contribute to their undue strength, specifically where they remain strong despite still capable of inflicting real damage or utility.

This study seeks to answer a key question: Does Tank-class champion status causally increase win rate, after adjusting for baseline strength factors such as health, armor, and resistances?

Because simple comparisons of win rates between champion classes are confounded by many baseline differences, we turn to a causal inference strategy specifically, propensity scores to estimate the effect of class designation.

Specifically, we utilize Propensity Scores and Propensity Score Matching (PSM) to equate observed differences in champion statistics. PSM allows for a more accurate comparison by equating Tank and non-Tank champions on relevant covariates and permits us to estimate the Average Treatment Effect (ATE) of Tank status on game outcomes. Our objective is to split out the impact of class designation from other traits that independently affect win rate, thereby offering insight into whether or not Tanks are objectively privileged in competitive play.

All statistical analyses were performed using R (version 4.4.1), and a significance level of $=$ 0.05 was used for hypothesis testing. The hypothesis testing framework we decided to work with was the Neyman-Pearson Framework.

Propensity Score

A propensity score is the conditional probability that a subject receives the treatment, given their observed characteristics (covariates) [2, pg 201 - 202]. Written as, $P(Z = 1|x)$ where Z is the label for what treatment someone was assigned to. Formally, if we consider a group s with n_s individuals, and each individual i has a treatment probability π_{si}, then the average probability across the group is:

$\lambda(x_s) = \frac{1}{n} \sum_{i=1} \pi_{si}$

Here, $\lambda(x_s)$ is the overall chance that a randomly selected subject from stratum s receives the treatment. This value always lies between 0 and 1.

Think of a propensity score as a way to summarize, in one number, how likely someone is to get a treatment based on their background information. Instead of looking at all their details separately, we bundle it into a single probability. Imagine, "given who you are, there's a 70% chance you would have been treated." Now, we don't have to try and make comparable groups based on background information, now we can just focus on matching propensity score values. For more information on Propensity Scores, the standard paper is the influential paper by Rosenbaum and Rubin [3].

Motivation

In the design of experiments, randomization plays a central role. It is different from random sampling in survey design. Randomization ensures that, on average, every subject has the same probability of being assigned to the treatment group. This process not only guarantees valid standard errors but also justifies the use of Fisher's significance tests. Most importantly, randomization provides an unbiased estimate of the treatment effect.

Now, consider if we could replicate this mechanism in an observational study - creating groups of subjects who have similar probabilities of receiving treatment. Under a key assumption known as strong ignorability, we can approximate the conditions of a randomized experiment. If strong ignorability holds, the estimated Average Treatment Effect (ATE) from the observational study would also be unbiased.

Propensity Score Matching (PSM) is a somewhat controversial method. The debate arises from statisticians' concerns about its misuse, similar to criticisms of how confidence intervals or p-values are sometimes misinterpreted, particularly in fields like epidemiology. Some critics argue that PSM should be abandoned altogether, with King et al. (2019) [4] being a key reference for this stance. For a balanced perspective, Senn et al. (2007) [5] offers a less critical comparison between PSM and standard covariate adjustment. Additionally, those interested in a defense of PSM, including a direct response to King et al., may find further insights in Wan 2025 [6]. Finally, a thorough walk through of Propensity Score Matching is given by Caliendo et al 2008 [7].

With this motivation in place, we next demonstrate how Propensity Score Matching is implemented using a well-known example, the Lalonde dataset.

Propensity Score Matching

To demonstrate Propensity Scores in action, we will follow the work flow and example provided by Noah Greifer on what's called, Propensity Score Matching, using his MatchIt library.

In the tutorial, Greifer uses the Lalonde Data set, which is a subset of data sets from the National Supported Work Demonstration used by Dehejia and Wahba to evaluate propensity score matching methods. A table of the first few rows is provided below:

```
     treat age educ   race married nodegree re74 re75        re78
NSW1     1  37   11  black       1        1    0    0  9930.0460
NSW2     1  22    9 hispan       0        1    0    0  3595.8940
NSW3     1  30   12  black       0        0    0    0 24909.4500
NSW4     1  27   11  black       0        1    0    0  7506.1460
NSW5     1  33    8  black       0        1    0    0   289.7899
NSW6     1  22    9  black       0        1    0    0  4056.4940
```

The workflow consistents of, selecting the type of effect to be estimated, selecting the target population to which the treatment effect is to generalize, selecting the matching algorithm, and selecting the covariates for which balance is required for an unbiased estimate of the treatment effect.

First, we check for initial imbalances in the lalonde data set (this is prior to any matching):

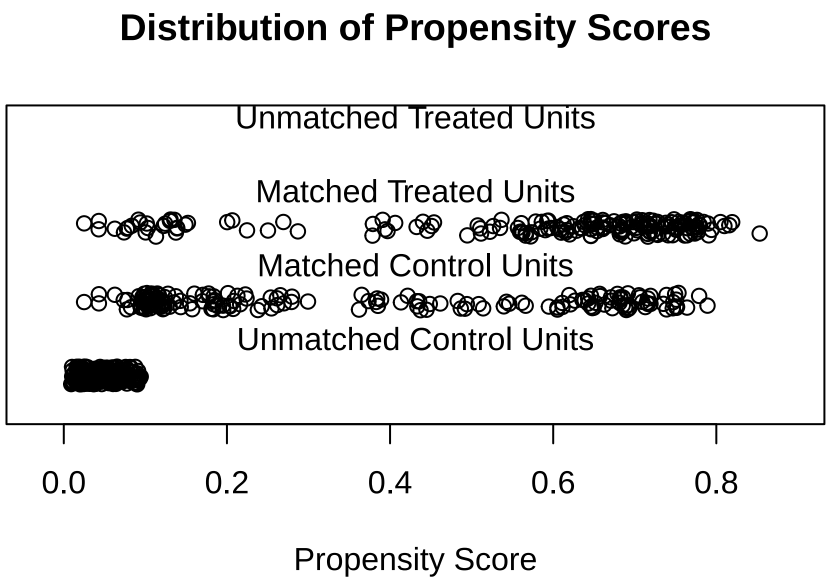
```
Call:
matchit(formula = treat ~ age + educ + race + married + nodegree +
    re74 + re75, data = df, method = NULL, distance = "glm")

Summary of Balance for All Data:
           Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
distance          0.5774        0.1822          1.7941     0.9211    0.3774
age              25.8162       28.0303         -0.3094     0.4400    0.0813
educ             10.3459       10.2354          0.0550     0.4959    0.0347
raceblack         0.8432        0.2028          1.7615          .    0.6404
racehispan        0.0595        0.1422         -0.3498          .    0.0827
racewhite         0.0973        0.6550         -1.8819          .    0.5577
married           0.1892        0.5128         -0.8263          .    0.3236
nodegree          0.7081        0.5967          0.2450          .    0.1114
re74           2095.5737     5619.2365         -0.7211     0.5181    0.2248
re75           1532.0553     2466.4844         -0.2903     0.9563    0.1342
           eCDF Max
distance     0.6444
age          0.1577
educ         0.1114
raceblack    0.6404
racehispan   0.0827
racewhite    0.5577
married      0.3236
nodegree     0.1114
re74         0.4470
re75         0.2876

Sample Sizes:
          Control Treated
All           429     185
Matched       429     185
Unmatched       0       0
Discarded       0       0
```

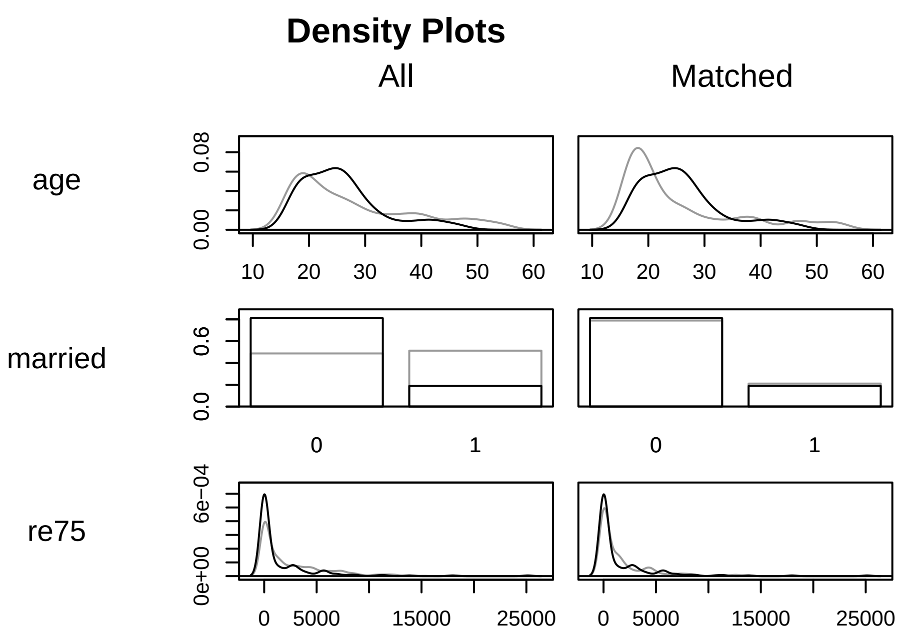
Looking at the Std. Mean Diff column, we notice quite a few covariate values that are far from 0 (values closer to 0 indicate good balance).

Now, we preform propensity score matching, deploying the 1:1 nearest neighbor algorithm. Looking again at the Std. Mean Diff column, we notice quite a few more covariates have scores closer to 0, indicating good balance.

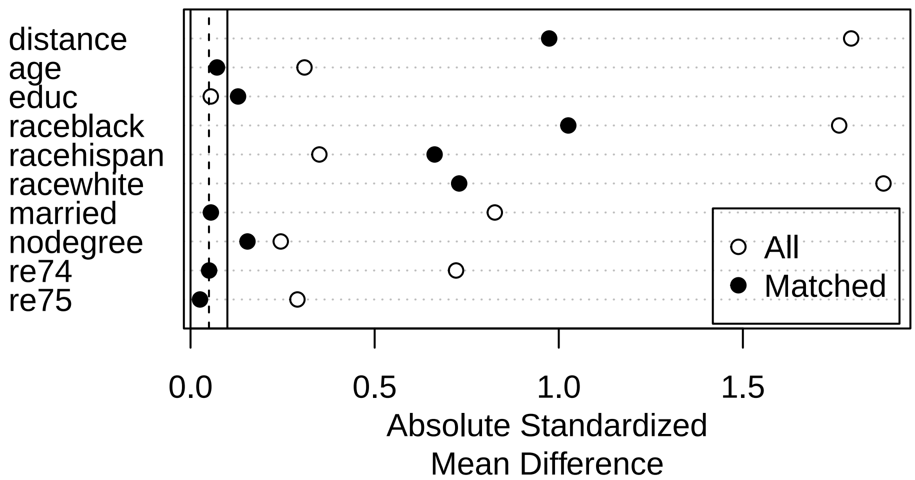
```
Call:
matchit(formula = treat ~ age + educ + race + married + nodegree +
    re74 + re75, data = df, method = "nearest", distance = "glm")

Summary of Balance for Matched Data:
           Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
distance          0.5774        0.3629          0.9739     0.7566    0.1321
age              25.8162       25.3027          0.0718     0.4568    0.0847
educ             10.3459       10.6054         -0.1290     0.5721    0.0239
raceblack         0.8432        0.4703          1.0259          .    0.3730
racehispan        0.0595        0.2162         -0.6629          .    0.1568
racewhite         0.0973        0.3135         -0.7296          .    0.2162
married           0.1892        0.2108         -0.0552          .    0.0216
nodegree          0.7081        0.6378          0.1546          .    0.0703
re74           2095.5737     2342.1076         -0.0505     1.3289    0.0469
re75           1532.0553     1614.7451         -0.0257     1.4956    0.0452
           eCDF Max Std. Pair Dist.
distance     0.4216          0.9740
age          0.2541          1.3938
educ         0.0757          1.2474
raceblack    0.3730          1.0259
racehispan   0.1568          1.0743
racewhite    0.2162          0.8390
married      0.0216          0.8281
nodegree     0.0703          1.0106
re74         0.2757          0.7965
re75         0.2054          0.7381

Sample Sizes:
          Control Treated
All           429     185
Matched       185     185
Unmatched     244       0
Discarded       0       0
```

Below are two plots for looking at the Distribution of of Propensity Scores.





Additionally, we can view the matching using plots. The last plot being what's called a Love plot. This providing a better idea of how well the matching algorithm worked



Data Sets

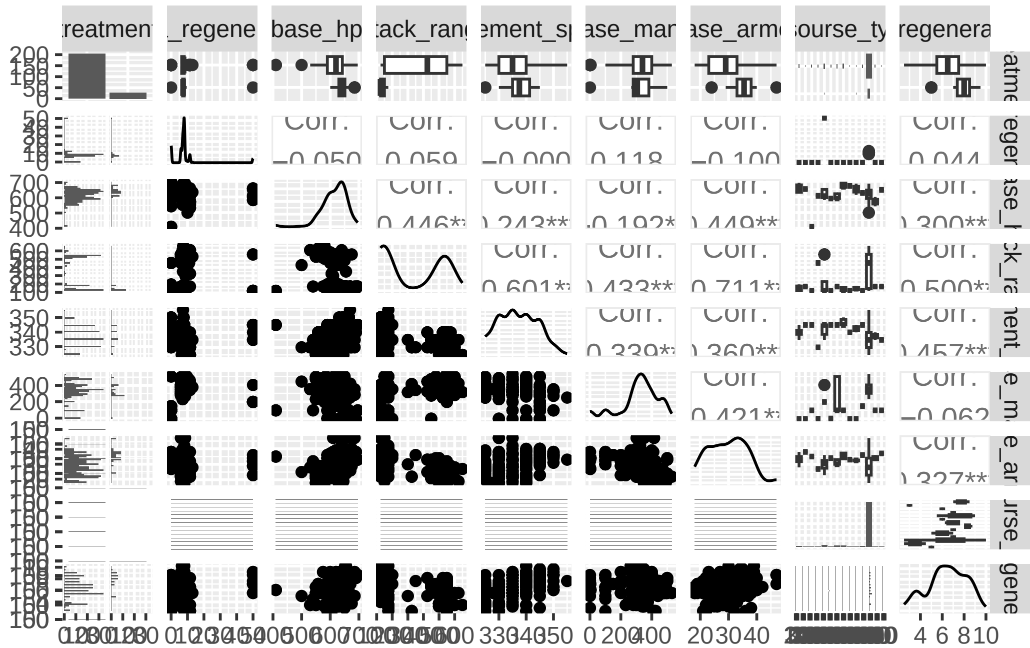
Below are links to the data sets used. The first two are pulled form Kaggle while the last one was pulled from the League of Legends Wiki

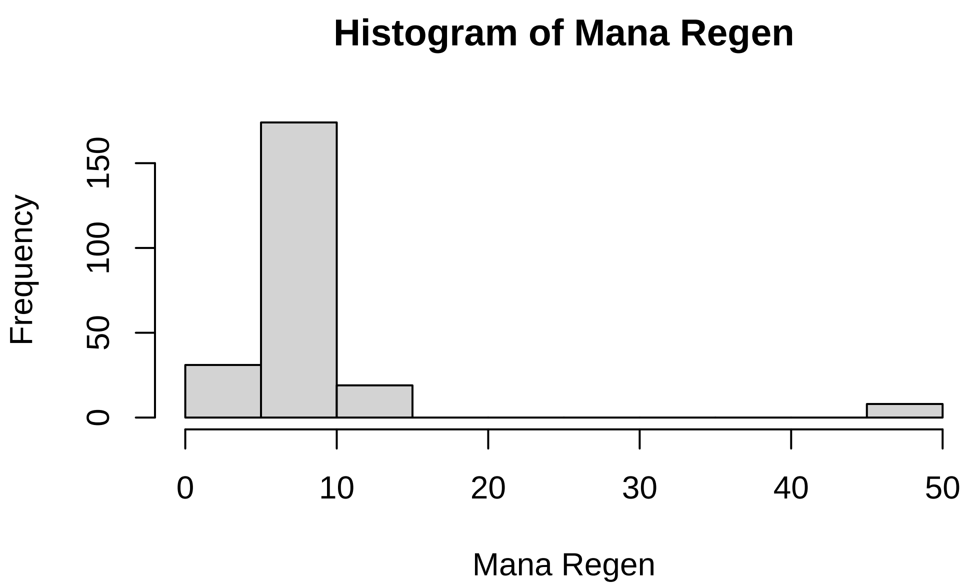
1. Legends Stats: S12 data set: https://www.kaggle.com/datasets/vivovinco/league-of-legends-champion-stats
2. League of Legends champions: https://www.kaggle.com/datasets/cutedango/league-of-legends-champions
3. Missing Data that was filled in came frame Wiki League: https://wiki.leagueoflegends.com/en-us/List_of_champions

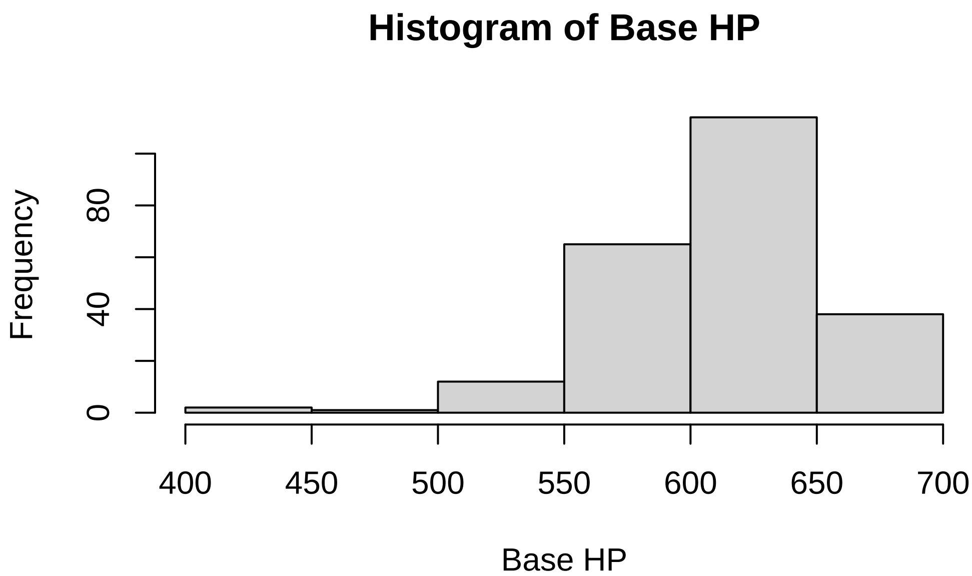
Analysis:

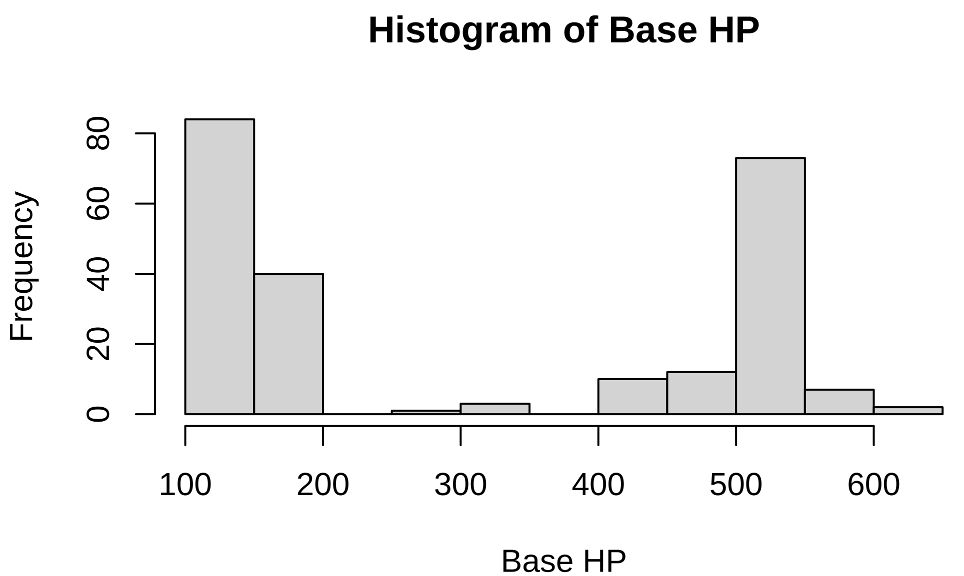
EDA:

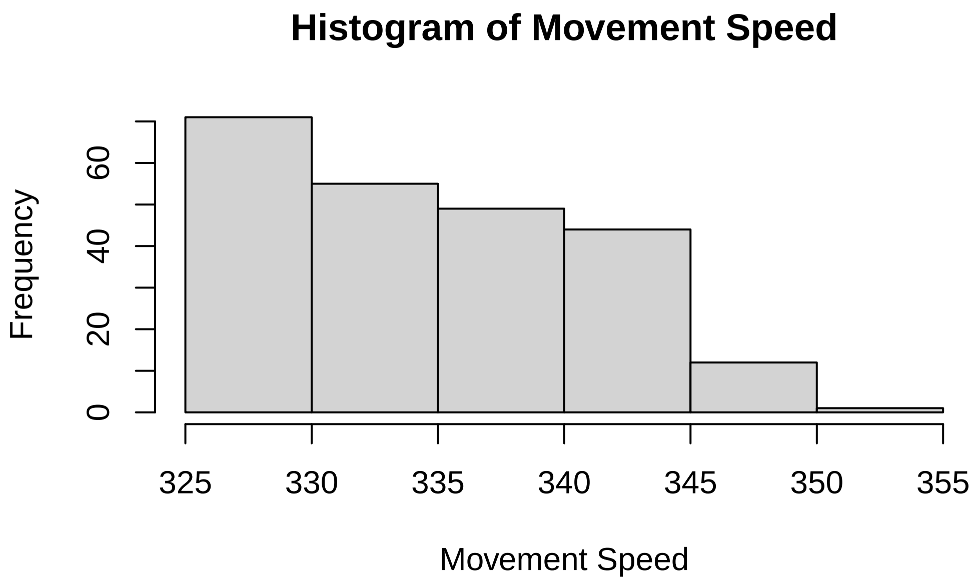
Here we inspect the data frame with a view plots of various variables of interest:

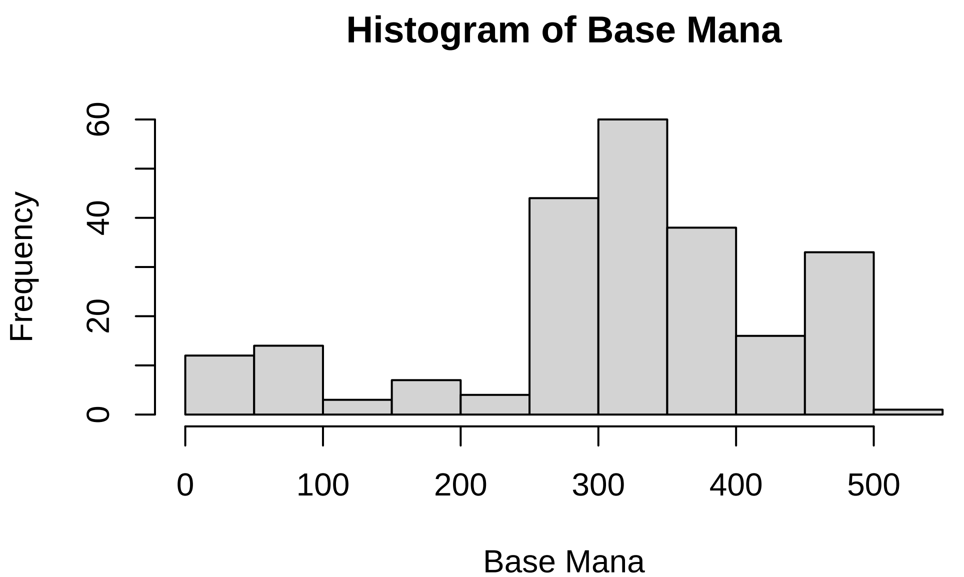

reatmen
_regene
base_hp
ack_ran
ement_s
ase_mar
ase_arm
ourse_ty
regenera

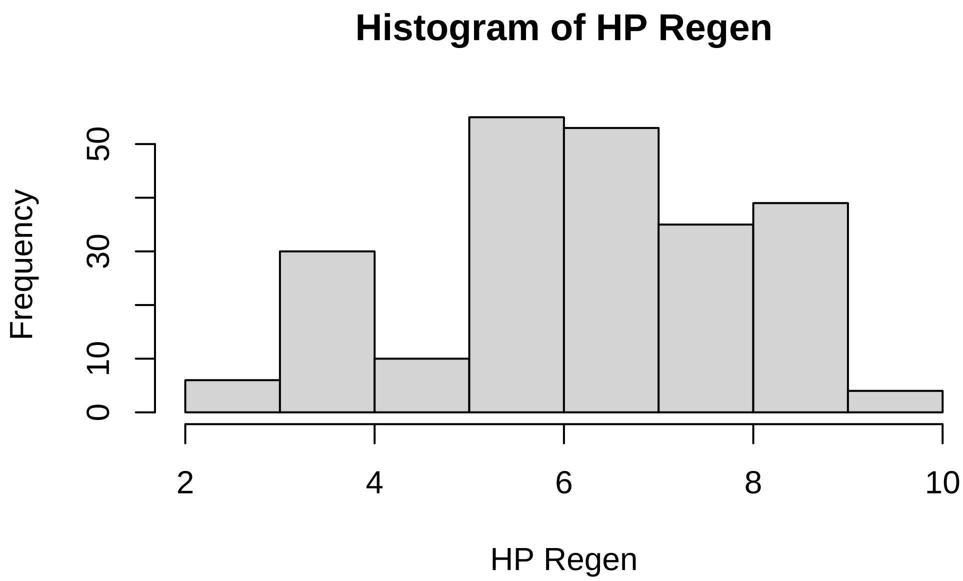

Histogram of Mana Regen
Frequency
Mana Regen

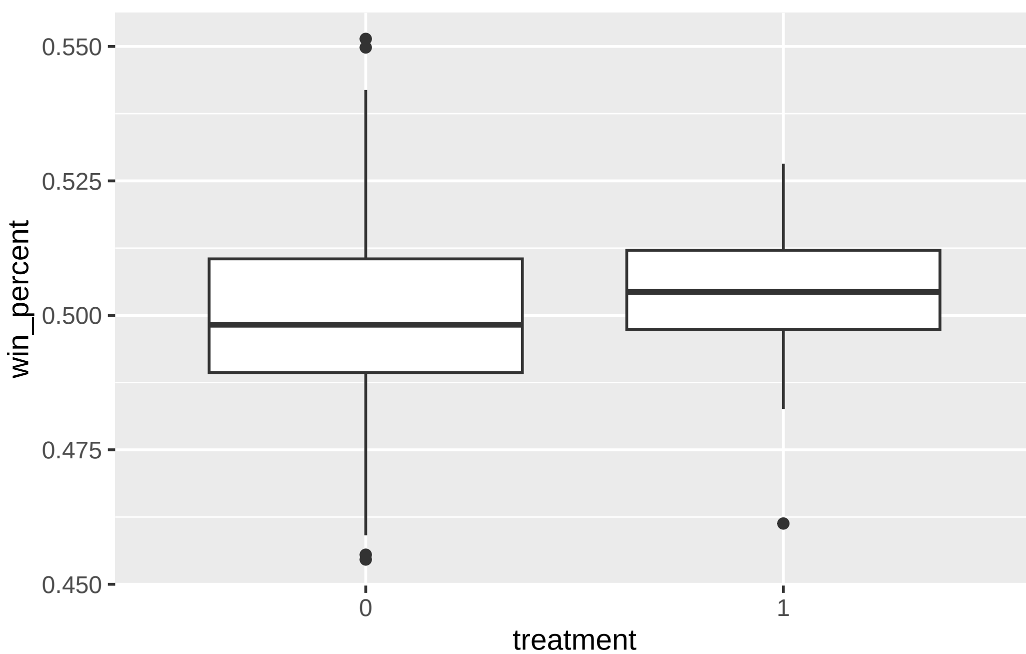

Histogram of Base HP
Frequency
0
40
80
400
450
500
550
600
650
700
Base HP

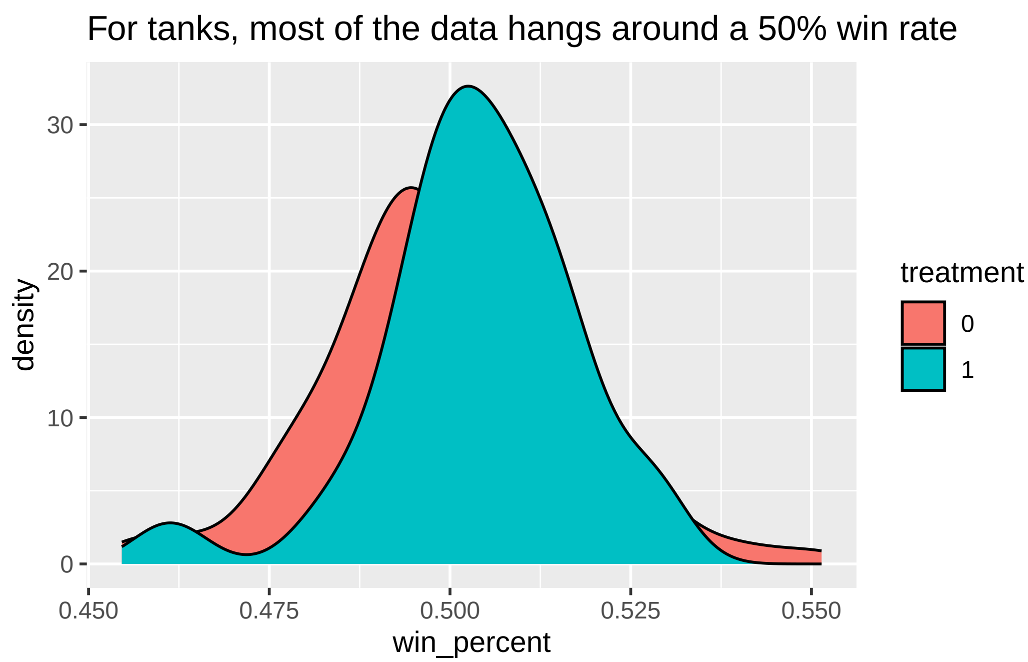

Histogram of Base HP
Frequency
0
20
40
60
80
100
200
300
400
500
600
Base HP


Histogram of Movement Speed
Frequency
0
20
40
60
325
330
335
340
345
350
355
Movement Speed


Histogram of Base Mana
Frequency
0
20
40
60
0
100
200
300
400
500
Base Mana


Histogram of HP Regen
Frequency
0
10
30
50
2
4
6
8
10
HP Regen


win_percent
0.550
0.525
0.500
0.475
0.450
0
1
treatment



Matching Methods

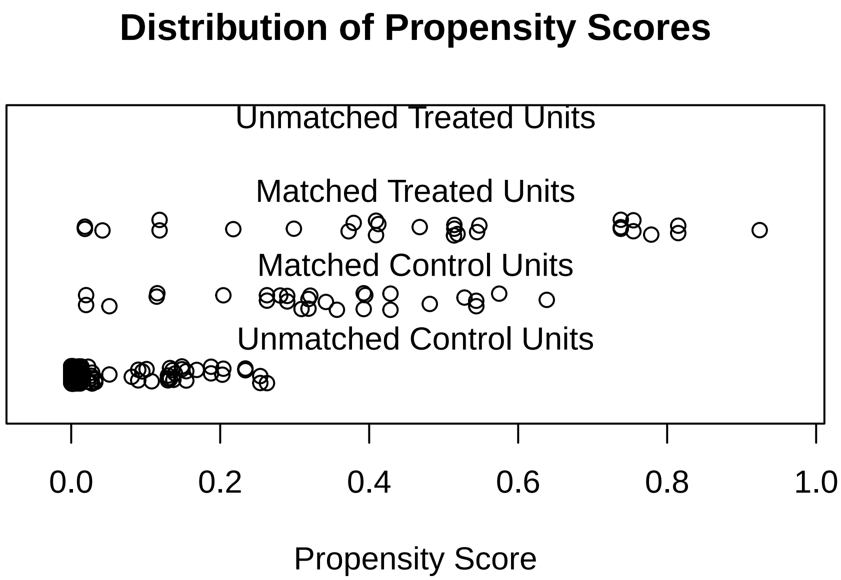
Now we deploy several matching method algorithms and plot how well they performed. The matching algorithm with the greatest number of matches will be selected:

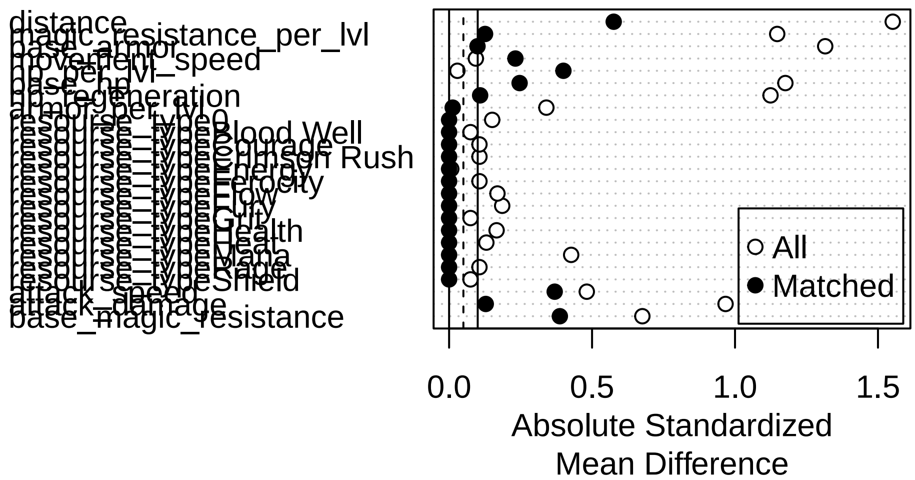
Nearest Neighbor

```
Call:
MatchIt::matchit(formula = treatment ~ magic_resistance_per_lvl +
    base_armor + movement_speed + hp_per_lvl + base_hp + hp_regeneration +
    armor_per_lvl + resourse_type + attack_speed + attack_damage +
    base_magic_resistance, data = League, method = "nearest",
    distance = "glm")

Summary of Balance for Matched Data:
                         Means Treated Means Control Std. Mean Diff.
distance                        0.4819        0.3295          0.5757
magic_resistance_per_lvl        2.0500        2.0946         -0.1258
base_armor                     35.4286       34.9286          0.0993
movement_speed                337.3214      338.7500         -0.2320
hp_per_lvl                    104.2500      108.1786         -0.3998
```

```
base_hp                       642.0000      635.8929          0.2465
hp_regeneration                 7.7500        7.8929         -0.1086
armor_per_lvl                   4.7929        4.7982         -0.0125
resourse_type0                  0.0000        0.0000          0.0000
resourse_typeBlood Well         0.0000        0.0000          0.0000
resourse_typeCourage            0.0000        0.0000          0.0000
resourse_typeCrimson Rush       0.0000        0.0000          0.0000
resourse_typeEnergy             0.0357        0.0357          0.0000
resourse_typeFerocity           0.0000        0.0000          0.0000
resourse_typeFlow               0.0000        0.0000          0.0000
resourse_typeFury               0.0000        0.0000          0.0000
resourse_typeGrit               0.0000        0.0000          0.0000
resourse_typeHealth             0.0357        0.0357          0.0000
resourse_typeHeat               0.0000        0.0000          0.0000
resourse_typeMana               0.9286        0.9286          0.0000
resourse_typeRage               0.0000        0.0000          0.0000
resourse_typeShield             0.0000        0.0000          0.0000
attack_speed                    0.6774        0.6556          0.3693
attack_damage                  62.4286       61.9286          0.1282
base_magic_resistance          31.5000       32.0000         -0.3873
                          Var. Ratio eCDF Mean eCDF Max Std. Pair Dist.
distance                      2.5772    0.0537   0.3929          0.5790
magic_resistance_per_lvl      0.0000    0.0255   0.1071          0.1258
base_armor                    0.8850    0.0586   0.3214          0.9360
movement_speed                1.6223    0.0536   0.1786          0.8698
hp_per_lvl                    1.1970    0.0903   0.2500          1.0394
base_hp                       0.9645    0.0628   0.1786          0.9415
hp_regeneration               0.9805    0.0476   0.1429          1.1399
armor_per_lvl                 1.0491    0.0339   0.1429          1.2144
resourse_type0                     .    0.0000   0.0000          0.0000
resourse_typeBlood Well            .    0.0000   0.0000          0.0000
resourse_typeCourage               .    0.0000   0.0000          0.0000
resourse_typeCrimson Rush          .    0.0000   0.0000          0.0000
resourse_typeEnergy                .    0.0000   0.0000          0.0714
resourse_typeFerocity              .    0.0000   0.0000          0.0000
resourse_typeFlow                  .    0.0000   0.0000          0.0000
resourse_typeFury                  .    0.0000   0.0000          0.0000
resourse_typeGrit                  .    0.0000   0.0000          0.0000
resourse_typeHealth                .    0.0000   0.0000          0.0714
resourse_typeHeat                  .    0.0000   0.0000          0.0000
resourse_typeMana                  .    0.0000   0.0000          0.1429
resourse_typeRage                  .    0.0000   0.0000          0.0000
resourse_typeShield                .    0.0000   0.0000          0.0000
```

```
attack_speed                     5.6926    0.1371   0.3571          0.8217
attack_damage                    0.9880    0.0286   0.1071          1.2634
base_magic_resistance            0.6081    0.0278   0.0714          0.7746

Sample Sizes:
          Control Treated
All           204      28
Matched        28      28
Unmatched     176       0
Discarded       0       0
```



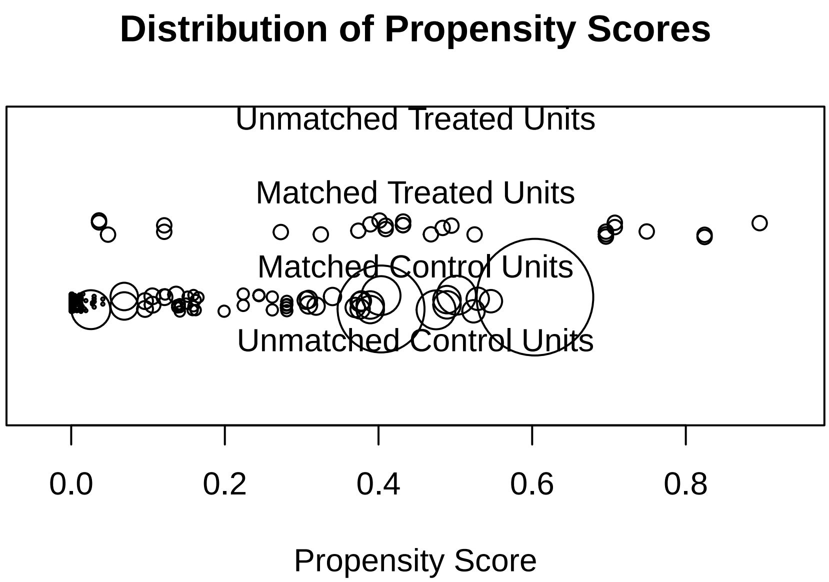


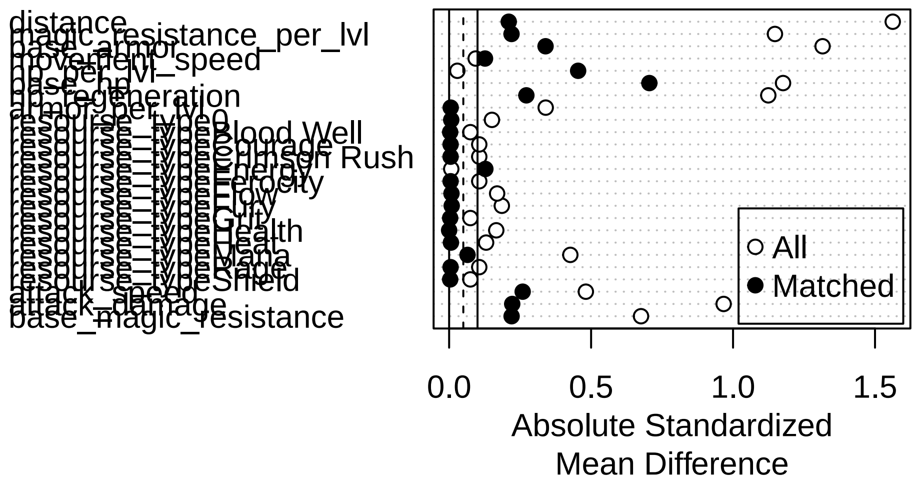
Full Matching on probit

```
Call:
MatchIt::matchit(formula = treatment ~ magic_resistance_per_lvl +
    base_armor + movement_speed + hp_per_lvl + base_hp + hp_regeneration +
    armor_per_lvl + resourse_type + attack_speed + attack_damage +
    base_magic_resistance, data = League, method = "full", distance = "glm",
    link = "probit")

Summary of Balance for Matched Data:
                         Means Treated Means Control Std. Mean Diff.
distance                        0.4646        0.4124          0.2100
magic_resistance_per_lvl        2.0500        2.1280         -0.2198
base_armor                     35.4286       33.7182          0.3396
movement_speed                337.3214      338.1002         -0.1264
hp_per_lvl                    104.2500      108.7151         -0.4544
base_hp                       642.0000      624.5300          0.7052
hp_regeneration                 7.7500        8.1080         -0.2721
armor_per_lvl                   4.7929        4.7905          0.0056
resourse_type0                  0.0000        0.0010         -0.0074
resourse_typeBlood Well         0.0000        0.0002         -0.0037
resourse_typeCourage            0.0000        0.0005         -0.0052
```

```
resourse_typeCrimson Rush       0.0000        0.0005          -0.0052
resourse_typeEnergy             0.0357        0.0120           0.1277
resourse_typeFerocity           0.0000        0.0005          -0.0052
resourse_typeFlow               0.0000        0.0012          -0.0083
resourse_typeFury               0.0000        0.0014          -0.0091
resourse_typeGrit               0.0000        0.0002          -0.0037
resourse_typeHealth             0.0357        0.0357           0.0000
resourse_typeHeat               0.0000        0.0007          -0.0064
resourse_typeMana               0.9286        0.9453          -0.0648
resourse_typeRage               0.0000        0.0005          -0.0052
resourse_typeShield             0.0000        0.0002          -0.0037
attack_speed                    0.6774        0.6621           0.2590
attack_damage                  62.4286       61.5618           0.2222
base_magic_resistance          31.5000       31.7841          -0.2200
                          Var. Ratio eCDF Mean eCDF Max Std. Pair Dist.
distance                      1.4929    0.0244   0.3214          0.1323
magic_resistance_per_lvl      0.0000    0.0827   0.2500          1.1424
base_armor                    0.7258    0.0803   0.2815          0.9744
movement_speed                1.5278    0.0528   0.1433          1.0554
hp_per_lvl                    0.5342    0.1019   0.2392          1.8308
base_hp                       0.5061    0.1194   0.3034          1.4372
hp_regeneration               1.0193    0.0489   0.2750          1.5190
armor_per_lvl                 1.1726    0.0740   0.1384          0.7644
resourse_type0                     .    0.0010   0.0010          0.1424
resourse_typeBlood Well            .    0.0002   0.0002          0.0707
resourse_typeCourage               .    0.0005   0.0005          0.1002
resourse_typeCrimson Rush          .    0.0005   0.0005          0.1002
resourse_typeEnergy                .    0.0237   0.0237          0.1996
resourse_typeFerocity              .    0.0005   0.0005          0.1002
resourse_typeFlow                  .    0.0012   0.0012          0.1596
resourse_typeFury                  .    0.0014   0.0014          0.1753
resourse_typeGrit                  .    0.0002   0.0002          0.0707
resourse_typeHealth                .    0.0000   0.0000          0.0185
resourse_typeHeat                  .    0.0007   0.0007          0.1230
resourse_typeMana                  .    0.0167   0.0167          0.7011
resourse_typeRage                  .    0.0005   0.0005          0.1002
resourse_typeShield                .    0.0002   0.0002          0.0707
attack_speed                  6.7433    0.2220   0.3928          0.5171
attack_damage                 1.9654    0.0786   0.3327          1.2046
base_magic_resistance         1.2998    0.0227   0.0880          1.2767

Sample Sizes:
              Control Treated
```

```
All              204.        28
Matched (ESS)      6.96      28
Matched          204.        28
Unmatched          0.         0
Discarded          0.         0
```





Optimal Matching on probit

```
Call:
MatchIt::matchit(formula = treatment ~ magic_resistance_per_lvl +
    base_armor + movement_speed + hp_per_lvl + base_hp + hp_regeneration +
    armor_per_lvl + resourse_type + attack_speed + attack_damage +
    base_magic_resistance, data = League, method = "optimal",
    distance = "glm", link = "probit")

Summary of Balance for Matched Data:
                          Means Treated Means Control Std. Mean Diff.
distance                         0.4646        0.3479          0.4692
magic_resistance_per_lvl         2.0500        2.0946         -0.1258
base_armor                      35.4286       34.4286          0.1986
movement_speed                 337.3214      340.3571         -0.4929
hp_per_lvl                     104.2500      107.9286         -0.3743
base_hp                        642.0000      636.4643          0.2235
hp_regeneration                  7.7500        8.0893         -0.2578
armor_per_lvl                    4.7929        4.7696          0.0543
resourse_type0                   0.0000        0.0000          0.0000
resourse_typeBlood Well          0.0000        0.0000          0.0000
resourse_typeCourage             0.0000        0.0000          0.0000
```

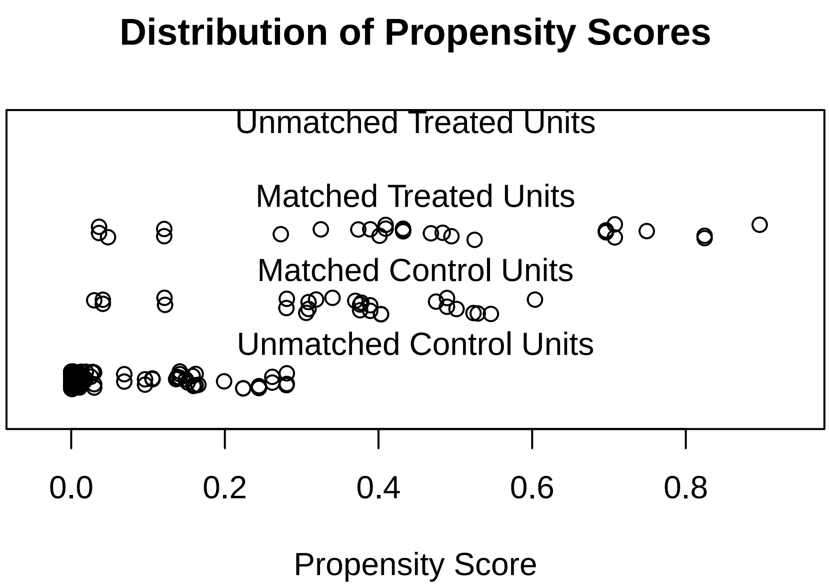
resourse_typeCrimson Rush	0.0000	0.0000	0.0000
resourse_typeEnergy	0.0357	0.0000	0.1925
resourse_typeFerocity	0.0000	0.0000	0.0000
resourse_typeFlow	0.0000	0.0000	0.0000
resourse_typeFury	0.0000	0.0000	0.0000
resourse_typeGrit	0.0000	0.0000	0.0000
resourse_typeHealth	0.0357	0.0357	0.0000
resourse_typeHeat	0.0000	0.0000	0.0000
resourse_typeMana	0.9286	0.9643	-0.1387
resourse_typeRage	0.0000	0.0000	0.0000
resourse_typeShield	0.0000	0.0000	0.0000
attack_speed	0.6774	0.6607	0.2838
attack_damage	62.4286	62.3571	0.0183
base_magic_resistance	31.5000	31.8929	-0.3043

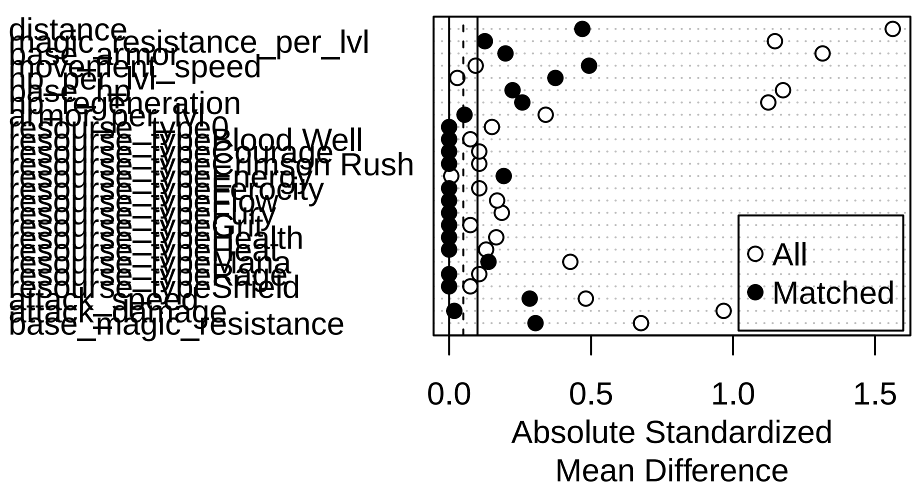
	Var. Ratio	eCDF Mean	eCDF Max	Std. Pair Dist.
distance	2.5040	0.0441	0.3571	0.4731
magic_resistance_per_lvl	0.0000	0.0255	0.1071	0.1258
base_armor	0.8049	0.0586	0.3214	1.1204
movement_speed	1.2099	0.0848	0.2500	1.0148
hp_per_lvl	1.1863	0.0882	0.2500	0.9849
base_hp	1.1033	0.0589	0.1786	1.0510
hp_regeneration	1.1258	0.0456	0.1786	0.9092
armor_per_lvl	0.9281	0.0286	0.1429	1.1560
resourse_type0	.	0.0000	0.0000	0.0000
resourse_typeBlood Well	.	0.0000	0.0000	0.0000
resourse_typeCourage	.	0.0000	0.0000	0.0000
resourse_typeCrimson Rush	.	0.0000	0.0000	0.0000
resourse_typeEnergy	.	0.0357	0.0357	0.1925
resourse_typeFerocity	.	0.0000	0.0000	0.0000
resourse_typeFlow	.	0.0000	0.0000	0.0000
resourse_typeFury	.	0.0000	0.0000	0.0000
resourse_typeGrit	.	0.0000	0.0000	0.0000
resourse_typeHealth	.	0.0000	0.0000	0.0714
resourse_typeHeat	.	0.0000	0.0000	0.0000
resourse_typeMana	.	0.0357	0.0357	0.4160
resourse_typeRage	.	0.0000	0.0000	0.0000
resourse_typeShield	.	0.0000	0.0000	0.0000
attack_speed	6.4174	0.1682	0.3571	0.9096
attack_damage	1.1998	0.0200	0.1071	1.2818
base_magic_resistance	5.1852	0.0317	0.1071	0.3043

Sample Sizes:

```
          Control Treated
```

```
All            204      28
Matched         28      28
Unmatched      176       0
Discarded        0       0
```





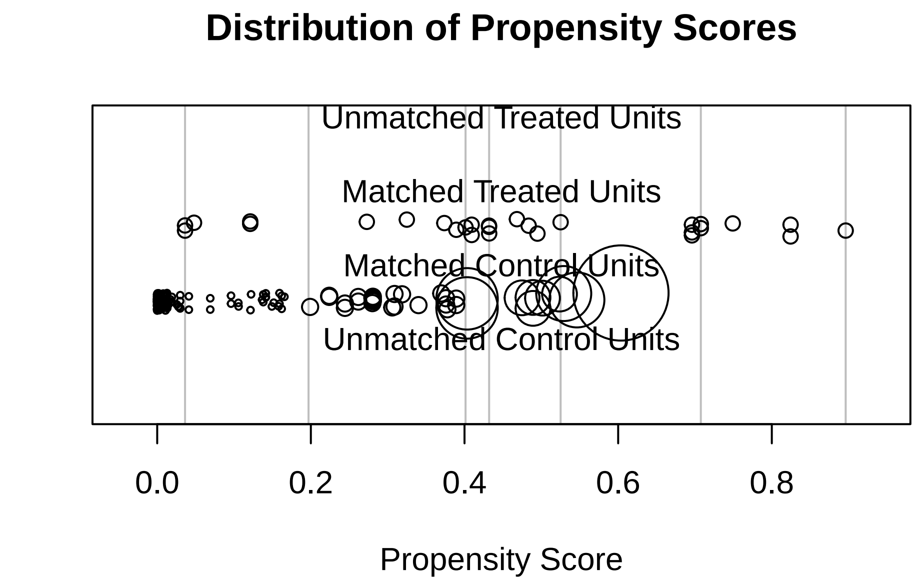
Subclass matching on probit

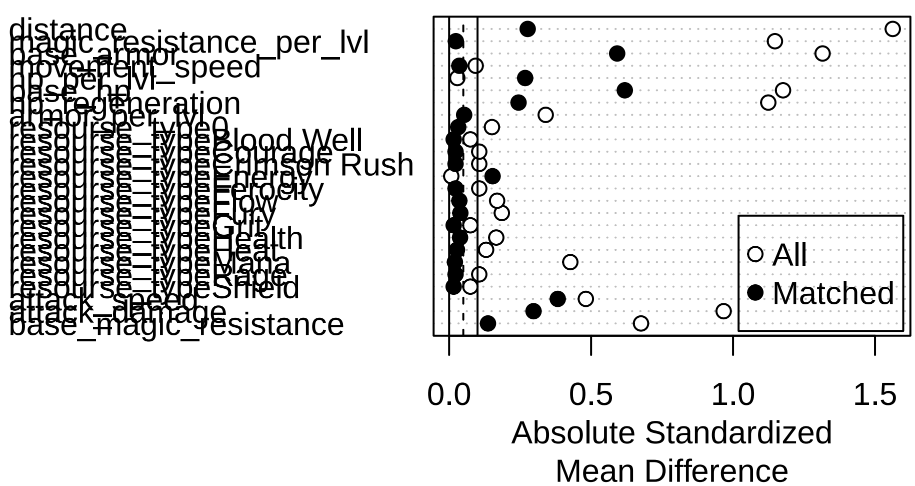
```
Call:
MatchIt::matchit(formula = treatment ~ magic_resistance_per_lvl +
    base_armor + movement_speed + hp_per_lvl + base_hp + hp_regeneration +
    armor_per_lvl + resourse_type + attack_speed + attack_damage +
    base_magic_resistance, data = League, method = "subclass",
    distance = "glm", link = "probit")

Summary of Balance Across Subclasses
                          Means Treated Means Control Std. Mean Diff.
distance                         0.4646        0.3958          0.2767
magic_resistance_per_lvl         2.0500        2.0584         -0.0237
base_armor                      35.4286       32.4477          0.5919
movement_speed                 337.3214      337.5433         -0.0360
hp_per_lvl                     104.2500      106.8821         -0.2679
base_hp                        642.0000      626.6716          0.6188
hp_regeneration                  7.7500        8.0718         -0.2445
armor_per_lvl                    4.7929        4.7702          0.0530
resourse_type0                   0.0000        0.0042         -0.0321
resourse_typeBlood Well          0.0000        0.0010         -0.0159
resourse_typeCourage             0.0000        0.0021         -0.0226
resourse_typeCrimson Rush        0.0000        0.0021         -0.0226
resourse_typeEnergy              0.0357        0.0073          0.1531
resourse_typeFerocity            0.0000        0.0021         -0.0226
resourse_typeFlow                0.0000        0.0052         -0.0360
resourse_typeFury                0.0000        0.0063         -0.0395
resourse_typeGrit                0.0000        0.0010         -0.0159
resourse_typeHealth              0.0357        0.0286          0.0385
resourse_typeHeat                0.0000        0.0031         -0.0278
resourse_typeMana                0.9286        0.9338         -0.0204
resourse_typeRage                0.0000        0.0021         -0.0226
resourse_typeShield              0.0000        0.0010         -0.0159
attack_speed                     0.6774        0.6549          0.3826
attack_damage                   62.4286       61.2687          0.2973
base_magic_resistance           31.5000       31.6767         -0.1369
                          Var. Ratio eCDF Mean eCDF Max
distance                      1.4000    0.0657   0.3214
magic_resistance_per_lvl      0.0000    0.1042   0.2071
base_armor                    0.7301    0.1124   0.3731
movement_speed                1.6085    0.0492   0.1496
```

hp_per_lvl	0.6492	0.0770	0.2281
base_hp	0.4405	0.1030	0.2589
hp_regeneration	0.7704	0.0512	0.2565
armor_per_lvl	1.0647	0.0498	0.1009
resourse_type0	.	0.0042	0.0042
resourse_typeBlood Well	.	0.0010	0.0010
resourse_typeCourage	.	0.0021	0.0021
resourse_typeCrimson Rush	.	0.0021	0.0021
resourse_typeEnergy	.	0.0284	0.0284
resourse_typeFerocity	.	0.0021	0.0021
resourse_typeFlow	.	0.0052	0.0052
resourse_typeFury	.	0.0063	0.0063
resourse_typeGrit	.	0.0010	0.0010
resourse_typeHealth	.	0.0071	0.0071
resourse_typeHeat	.	0.0031	0.0031
resourse_typeMana	.	0.0053	0.0053
resourse_typeRage	.	0.0021	0.0021
resourse_typeShield	.	0.0010	0.0010
attack_speed	7.7057	0.2047	0.3562
attack_damage	0.9600	0.0574	0.2923
base_magic_resistance	0.8229	0.0216	0.0842

Sample Sizes:

	Control	Treated
All	204.	28
Matched (ESS)	12.95	28
Matched	204.	28
Unmatched	0.	0
Discarded	0.	0



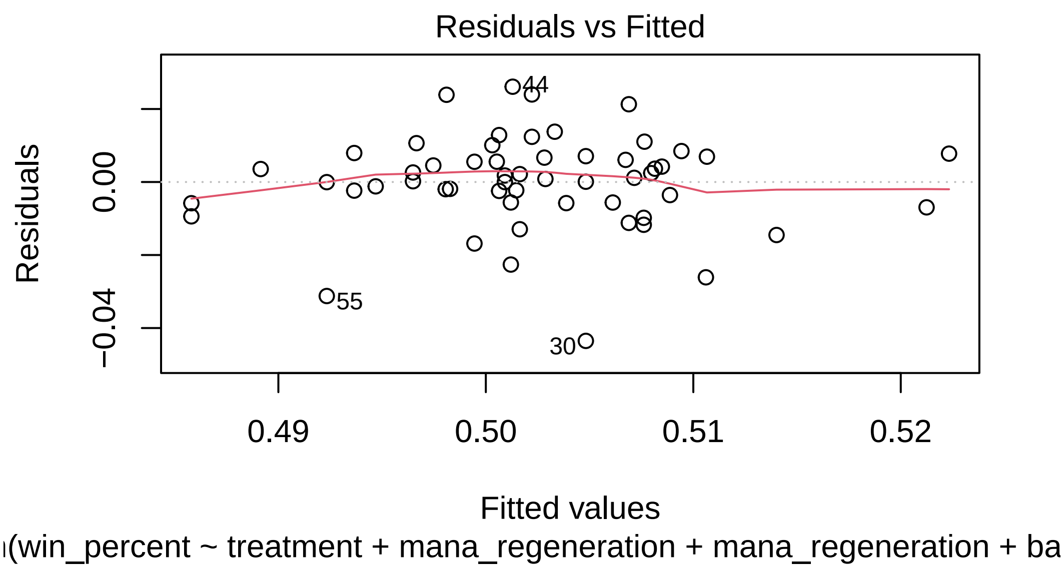


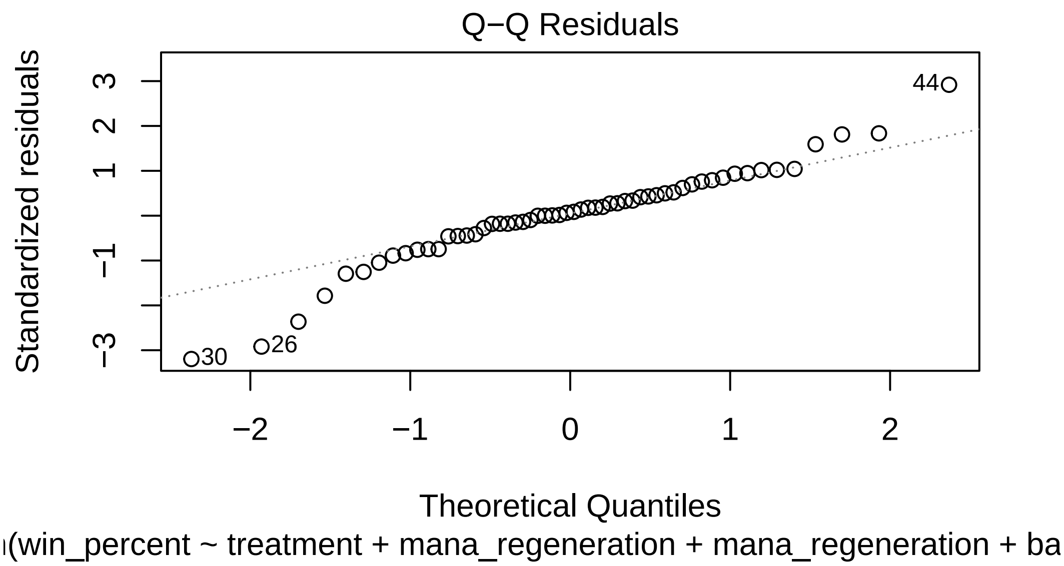
The most optimal matching algorithm was Nearest Neighbor based off of visual inspection of the plots.

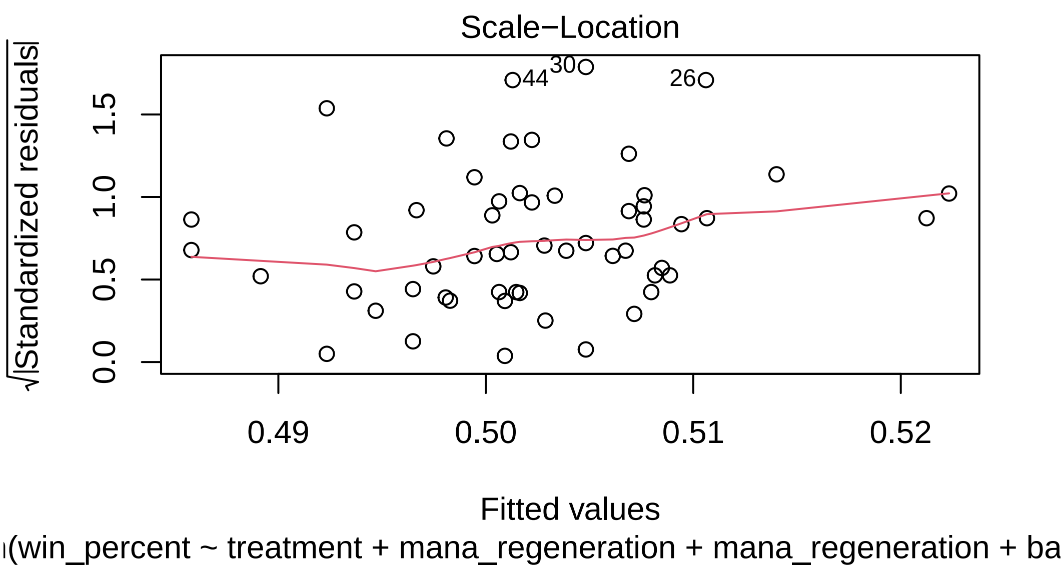
To evaluate whether our model assumptions hold and ensure the validity of our estimated treatment effect, we next conduct regression diagnostics on the matched data set.

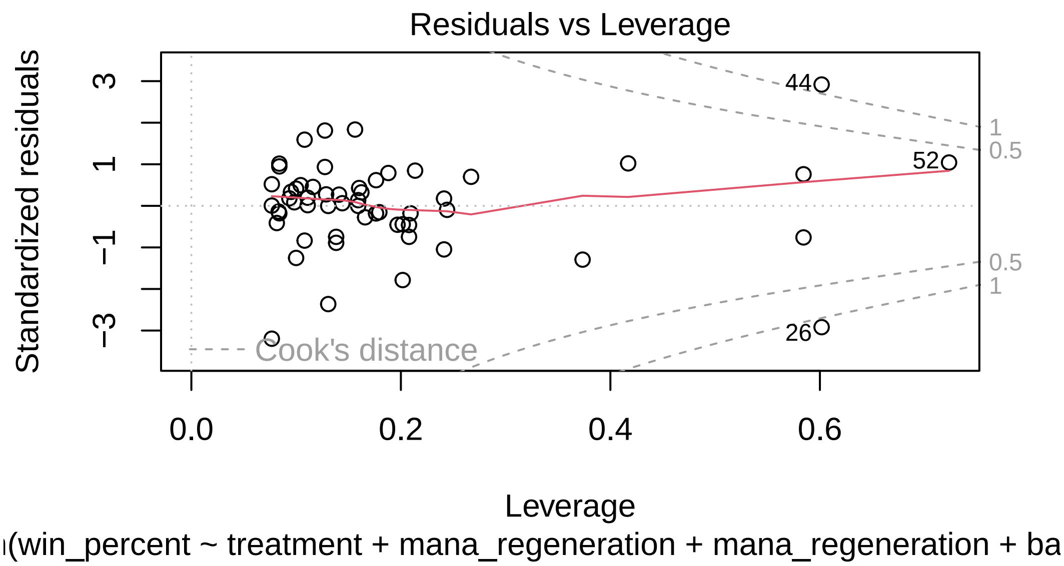
Diagnostics

The Residuals vs. Fitted plot shows residuals fairly evenly distributed around zero with no visible pattern, suggesting linearity and homoscedasticity are not considerably violated. The Q-Q plot suggests residuals do follow a normal distribution with some deviations in the tails, particularly for quite a few outliers (e.g., points 26, 30, and 44). The Residuals vs. Leverage plot indicates that most points have low leverage and standardized residuals in ± 2, with only a few observations (most notably 44 and 52) having near-moderate Cook's distance values. Overall, there is no indication of extreme influence or gross violations, and the model appears to be suitable for inference.




Q−Q Residuals
Standardized residuals
Theoretical Quantiles
(win_percent ~ treatment + mana_regeneration + mana_regeneration + ba


Scale−Location
√|Standardized residuals|
Fitted values
(win_percent ~ treatment + mana_regeneration + mana_regeneration + ba



Data Frame

Data for League of Legend Champion Features for Season 12 were extracted from two data sets that were joined on Champion name. Here we show the first few rows and columns of the joined data frame.

```
# A tibble: 6 x 35
  name   tags       role.x range_type resourse_type base_hp hp_per_lvl base_mana
  <chr>  <chr>      <chr>  <chr>      <chr>           <dbl>      <dbl>     <dbl>
1 Aatrox Fighter    Top    Melee      Blood Well        650        114         0
2 Ahri   Mage,Assa~ Middle Ranged     Mana              590        104       418
3 Akali  Assassin   Top,M~ Melee      Energy            600        119       200
4 Akali  Assassin   Top,M~ Melee      Energy            600        119       200
5 Akshan Marksman,~ Middle Ranged     Mana              630        107       350
6 Akshan Marksman,~ Middle Ranged     Mana              630        107       350
# i 27 more variables: mana_per_lvl <dbl>, movement_speed <dbl>,
#   base_armor <dbl>, armor_per_lvl <dbl>, base_magic_resistance <dbl>,
#   magic_resistance_per_lvl <dbl>, attack_range <dbl>, hp_regeneration <dbl>,
#   hp_regeneration_per_lvl <dbl>, mana_regeneration <dbl>,
#   mana_regeneration_per_lvl <dbl>, attack_damage <dbl>,
#   attack_damage_per_lvl <dbl>, attack_speed_per_lvl <dbl>,
#   attack_speed <dbl>, as_ratio <dbl>, class <chr>, role.y <chr>, ...
```

Methods

A Priori Steps for Test:

We define a minimum effect size (MES) of 0.03, corresponding to a 3 percentage point difference in win rate, as the threshold of practical importance in champion balancing. We use a two-tailed t-test to compare matched Tank and non-Tank win rates, with = 0.05. The null hypothesis ($H_0 = \mu_{tank} - \mu_{non-tank} = 0$) assumes that the difference in win rates between Tank and non-Tank champions is 0. The alternative hypothesis posits a win rate difference greater than 3 percentage points between Tank and non-Tank champions.

Power Analysis:

Using an expected minimum effect size of 3 percentage points (MES = 0.03) and the observed standard deviation of win rates ($\sigma = 0.01656$), we calculated that a sample of just 6 champions would provide 80% power to detect such an effect using a two-tailed t-test with $\alpha = 0.05$.

```
     Two-sample t test power calculation 

              n = 5.919827
              d = 1.811
      sig.level = 0.05
          power = 0.8
    alternative = two.sided

NOTE: n is number in *each* group
```

Check Imbalances

Per the standard workflow, we show pre-matching data. Recall that, we're looking at the Std. Mean Diff and values closer to 0 indicate good matching.

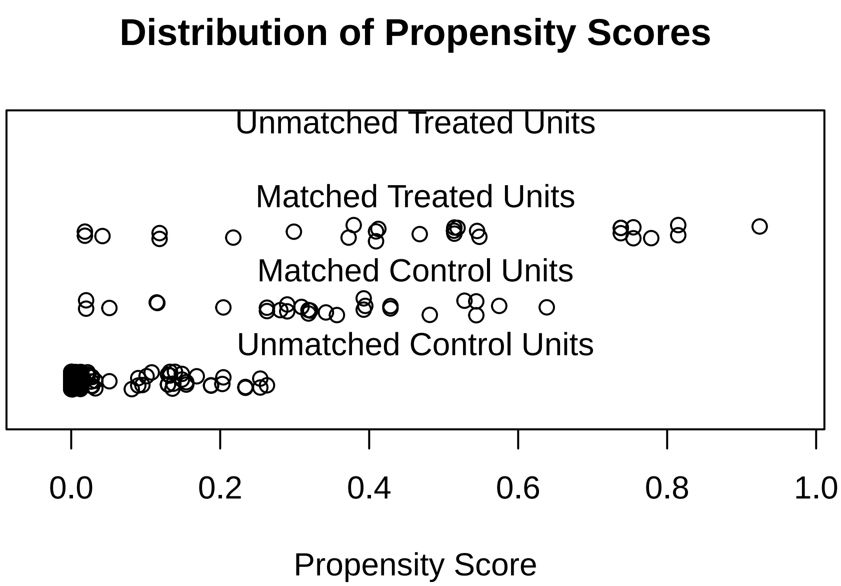
```
Call:
MatchIt::matchit(formula = treatment ~ mana_regeneration + base_hp + 
    attack_range + movement_speed + base_mana + base_armor + 
    resourse_type + hp_regeneration, data = League, method = NULL, 
    distance = "glm")

Summary of Balance for All Data:
                          Means Treated Means Control Std. Mean Diff.
```

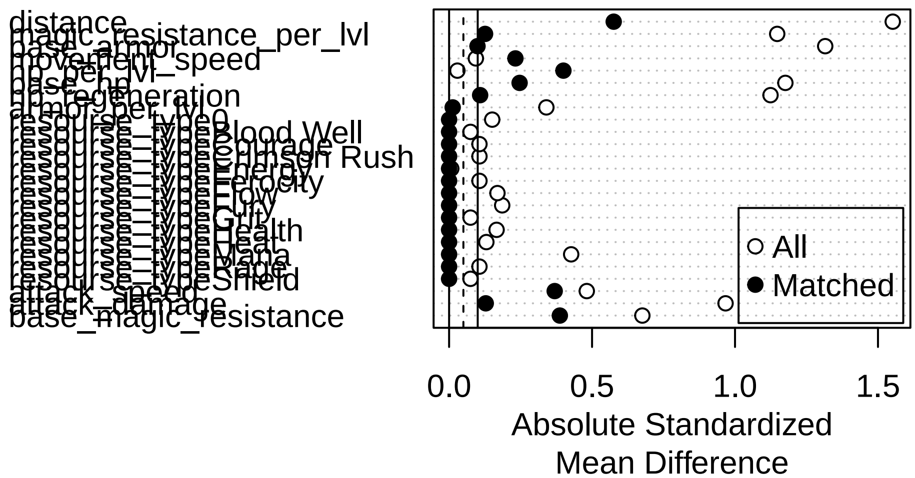
distance	0.4134	0.0805	1.7558
mana_regeneration	8.6643	8.2169	0.0541
base_hp	642.0000	612.8676	1.1760
attack_range	139.2857	346.4706	-9.9166
movement_speed	337.3214	336.7451	0.0936
base_mana	324.6429	320.4559	0.0456
base_armor	35.4286	28.8039	1.3154
resourse_type0	0.0000	0.0196	-0.1508
resourse_typeBlood Well	0.0000	0.0049	-0.0748
resourse_typeCourage	0.0000	0.0098	-0.1061
resourse_typeCrimson Rush	0.0000	0.0098	-0.1061
resourse_typeEnergy	0.0357	0.0343	0.0075
resourse_typeFerocity	0.0000	0.0098	-0.1061
resourse_typeFlow	0.0000	0.0245	-0.1690
resourse_typeFury	0.0000	0.0294	-0.1856
resourse_typeGrit	0.0000	0.0049	-0.0748
resourse_typeHealth	0.0357	0.0049	0.1660
resourse_typeHeat	0.0000	0.0147	-0.1303
resourse_typeMana	0.9286	0.8186	0.4269
resourse_typeRage	0.0000	0.0098	-0.1061
resourse_typeShield	0.0000	0.0049	-0.0748
hp_regeneration	7.7500	6.2708	1.1241

	Var. Ratio	eCDF Mean	eCDF Max
distance	1.5657	0.4238	0.7654
mana_regeneration	0.9510	0.0822	0.2262
base_hp	0.3590	0.1886	0.3908
attack_range	0.0117	0.3408	0.5441
movement_speed	0.7456	0.0350	0.1043
base_mana	0.5363	0.0912	0.1877
base_armor	0.6279	0.2529	0.5385
resourse_type0	.	0.0196	0.0196
resourse_typeBlood Well	.	0.0049	0.0049
resourse_typeCourage	.	0.0098	0.0098
resourse_typeCrimson Rush	.	0.0098	0.0098
resourse_typeEnergy	.	0.0014	0.0014
resourse_typeFerocity	.	0.0098	0.0098
resourse_typeFlow	.	0.0245	0.0245
resourse_typeFury	.	0.0294	0.0294
resourse_typeGrit	.	0.0049	0.0049
resourse_typeHealth	.	0.0308	0.0308
resourse_typeHeat	.	0.0147	0.0147
resourse_typeMana	.	0.1099	0.1099
resourse_typeRage	.	0.0098	0.0098

```
resourse_typeShield                      .     0.0049   0.0049
hp_regeneration                     0.5843     0.1746   0.4706

Sample Sizes:
          Control Treated
All           204      28
Matched       204      28
Unmatched       0       0
Discarded       0       0
```

After deploying NN matching, we show how well the matching algorithm performed. Per our EDA, this was the best matching algorithm. Now we have a lot more champions that are similar. As to be expected, quite a lot of champions were dropped for not being comparable.





Results

Under our pre-specified decision rule, with a test that had sufficient power to detect a minimum meaningful effect size of 3 percentage points, the observed test statistic fell within the acceptance region.. Therefore, in accordance with the above framework, we will behave as if the main hypothesis is true, that Tank-class assignment does not meaningfully affect win rate after adjusting for champion stats. Had Tanks truly conferred such an advantage, we should have observed a more extreme result. We will add these results to our background knowledge for future studies.

We recommend that Champion item and Player Rank data be included in future studies. We have reason to think that Champion Items may have a large causal impact on Champion Win Rate, contingent on the season. Particularly if the Tank class is favored by the items that are in season.

	Estimate	Std. Error	t value	p-value	Critical t-value
treatment1	0.0025	0.0044	0.5825	0.5631	2.018

Limitations

There were some unobserved variables that could contribute to biasing my effect to some level. For example, data on Champion Items was not included and we believe that this variable would have a large positive effect specifically for tanks during certain seasons where Tank items are strong. Throughout the game, players need to purchase items for their champions to get stronger and there have been seasons where certain classes would benefit from item changes more than others. Another would be the Rank of the Win Rate. There are several Ranks within League of Legends.

Conclusion

In this study, we analyzed whether the Tank class name in League of Legends boosts causally a champion's win rate. Applying Propensity Score Matching to eliminate confounding champion attributes—e.g., health, resistances, and mobility. We attempted to isolate the effect of class status from other performance determinants.

After matching, we conducted hypothesis testing in a Neyman-Pearson framework. Our test was sufficiently powerful to identify a minimum practically significant effect (3 percentage points), but the win rate difference between Tank and non-Tank champions that we found was small and statistically non-significant.

Our results can be added to the background knowledge of future studies with similar research questions. The perception of Tanks being inherently overpowered may be better explained by seasons where tank items are much stronger than for other classes.

Citations

1. Rosenbaum, Paul R. *Design of Observational Studies.* Springer, 2020.
2. Rosenbaum, Paul R. Observational Studies. Springer
3. Rosenbaum, Paul R., and Donald B. Rubin. "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika*, vol. 70, no. 1, Jan. 1983, pp. 41–55. https://doi.org/10.1093/biomet/70.1.41.
4. King, Gary, and Richard Nielsen. "Why Propensity Scores Should Not Be Used for Matching." *Political Analysis*, vol. 27, no. 4, 2019, gking.harvard.edu/publications/why-propensity-scores-should-not-be-used-formatching.
5. Senn, Stephen, et al. "Stratification for the Propensity Score Compared with Linear Regression Techniques to Assess the Effect of Treatment or Exposure." *Statistics in Medicine*, vol. 26, no. 30, 3 Dec. 2007, pp. 5529–5544, https://doi.org/10.1002/sim.3133. Accessed 11 Oct. 2020.
6. Wan, Fei. "Propensity Score Matching: Should We Use It in Designing Observational Studies?" *BMC Medical Research Methodology*, vol. 25, no. 1, 29 Jan. 2025, https://doi.org/10.1186/s12874-025-02481-w.

7. Caliendo, Marco, and Sabine Kopeinig. "SOME PRACTICAL GUIDANCE for the IMPLEMENTATION of PROPENSITY SCORE MATCHING." Journal of Economic Surveys, vol. 22, no. 1, Feb. 2008, pp. 31–72, https://doi.org/10.1111/j.1467-6419.2007.00527.x.