A Data Analysis of My LoL Gameplay

Diego Perez

University of Utah

DS 2500: Intro to Data Science

Dr. Marina Kogan

May 1, 2024

A Data Analysis of My LoL Gameplay

Introduction

League of Legends is one of the most popular esports video games in the world with some players making millions of dollars. Players of League of Legends, or LoL for short, earn this money by professionally competing, or by streaming on sites like Twitch and Youtube. LoL is a complex game that requires continuous improvement and learning. While I do not professionally compete nor stream, I do play this game regularly. One of the most satisfying parts of this game is watching yourself grow into a better player overtime. My goal is to see how I can improve my gameplay so that I can get that satisfaction. Even though I have no goal to become a competitor or streamer, it is possible that with improvement I could unlock those doors. Also, the conclusions I draw here can be applied to the game as a whole and help other players of my skill level improve as well. While most players would simply go back and watch their old games to see points of improvement, I choose to statistically analyze my gameplay on a large scale to get a solid understanding of my gameplay.

Central Question

The central question to this data analysis is: what components of my League of Legends gameplay contribute to getting more assists? Players of LoL may find the question odd in asking about assists rather than kills or wins. However, in this case it is warranted considering I primarily play the role of support. Players of this role will often quantify their skill by how many assists they get during a game. Therefore, I will be using assists as my dependent variable for this analysis.

Data Collection

For this analysis, I chose to use the Riot Games API which has a plethora of data that can be collected. Through this API, one may gather information on any of Riot's games, and for this case, the LoL API calls will be used. A python notebook will be used to do this analysis which will utilize Riot Watcher, a wrapper library for the Riot Games API.

The first thing I did was get my PUUID on the Riot Games API. This is a string that is used to identify players that is only used for the API. With that, I used the 'matchlist by puuid' call to get my last 100 matches. While 100 matches may not sound like enough, it should be plenty for this analysis. More matches would analyze my performance 100 matches ago which is vastly different than now and would hinder the reliability of the data. With those matches, I used the 'match by id' call to get the data from all of those matches. This resulted in 100 matches, however, it may include game modes that I am uninterested in investigating. This analysis focuses on the classic game mode as that is the most popular and important gamemode. Therefore, any matches that weren't classic were filtered out. This left 74 classic matches to analyze. There should be plenty of matches to represent my current playstyle. I then checked the top and bottom of the data and immediately noticed the incredible number of columns. After looking through the column values, it was no mistake, the API simply gave a large number of data points. To clean the data, I removed any of these columns that weren't important to understanding my gameplay. I then checked to ensure that the assists column was accurate and that the data made sense. After this, the data should be clean, accurate, and ready to be used. To prevent needing to call the API again, I exported this data to a Github Repository.

Analysis

To answer the primary question, my goal is to create a fairly accurate linear regression model based on the data provided. In order to do this, I will be analyzing the data and whittling down the data to the most helpful points for the regression model. While doing this, I will account for the various linear regression assumptions.

I began by plotting out all of the variables that seemed to be the most important on a histogram in order to get an idea of the data's distribution.

Figure 1

Histogram Distributions

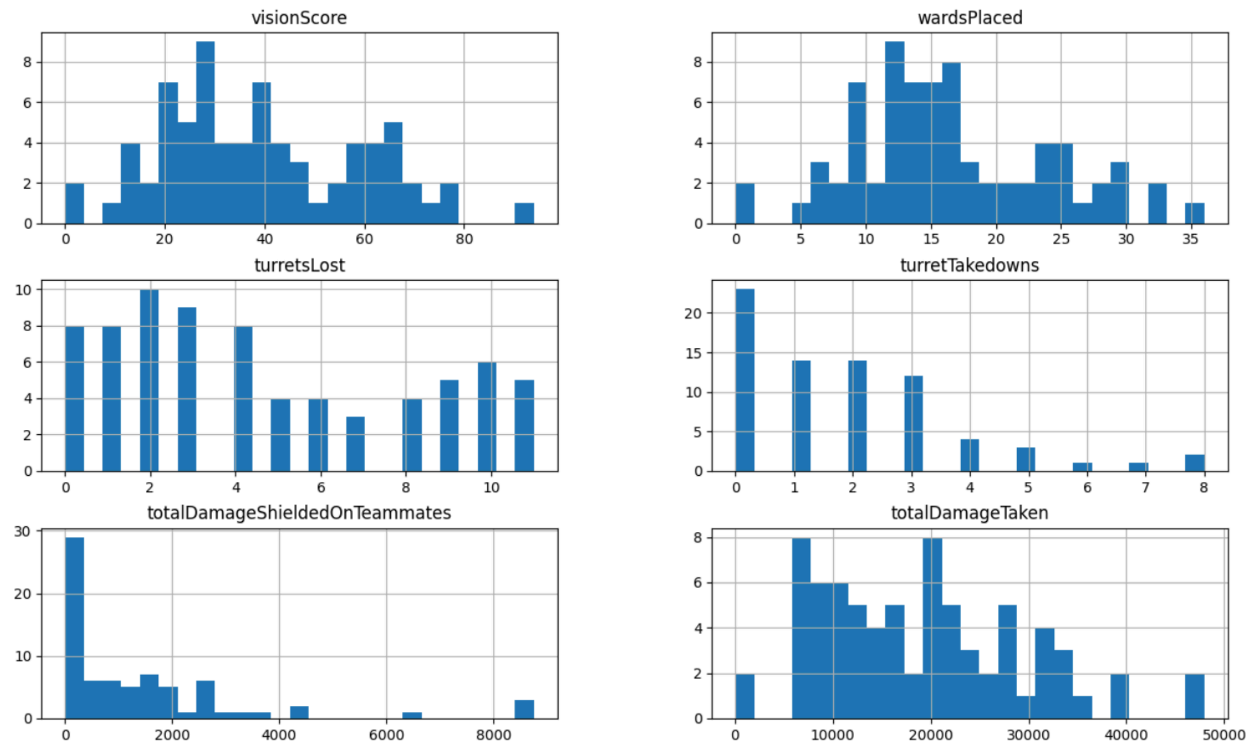
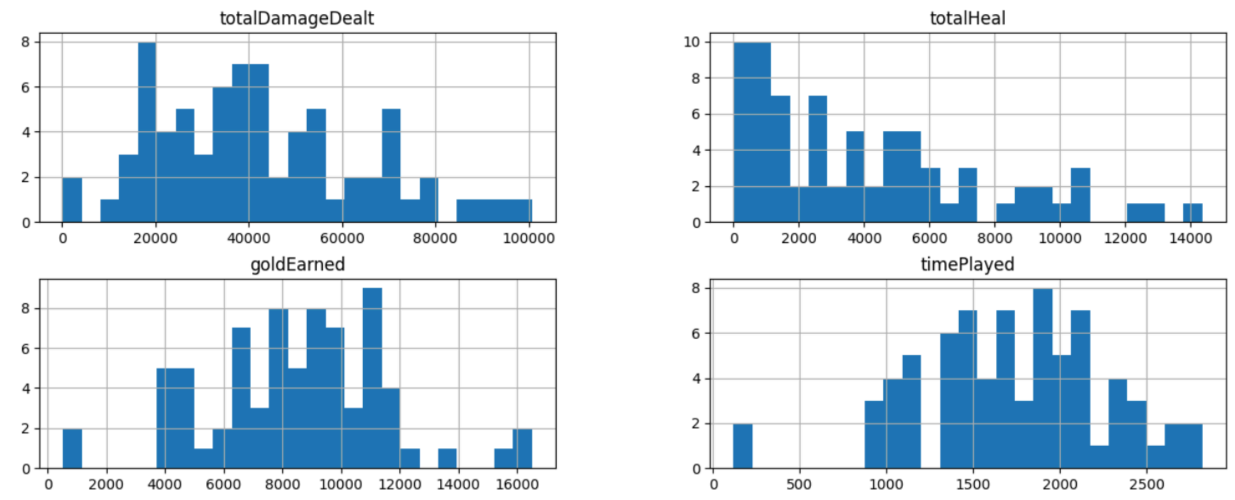


Figure 2

Histogram Distributions Cont.

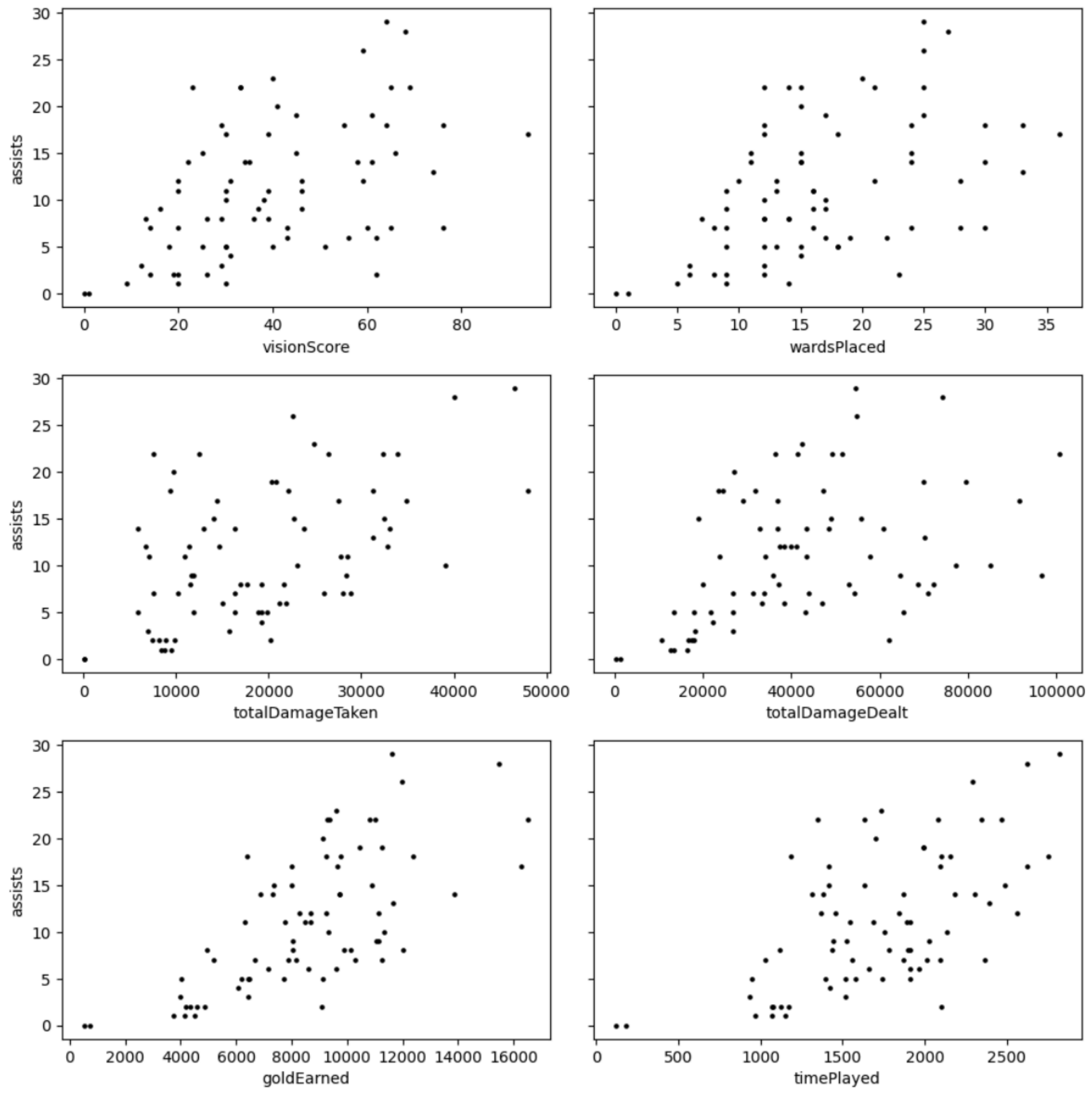


These 10 variables, through my understanding of the game, should have the most impact on the number of assists I get. For linear regression, one of the assumptions is that the data is evenly distributed. Only six variables out of the ten are relatively evenly distributed: vision score, wards placed, total damage taken, total damage dealt, goal earned, and time played. The variables turrets lost, and turret takedowns don't have any sort of distribution and should be omitted. The variables total heal and total shield on teammates are incredibly left leaning. This is reasonable as I do not play a champion that heals or shields every game. Therefore, only a few of those data points would be accurate so those variables will be omitted as well.

Then, I needed to check if the variables have a linear relationship with assists. This is important because linear regression assumes a linear relationship. I began plotting all of the variables on a scatter plot against assists (Fig 3). Luckily, upon initial inspection all of the variables seem to have a linear relationship. The relationships don't look incredibly strong, but there is moderate linearity. This figure does show that there may be a possibility for multicollinearity which will be further investigated later. Another one of the assumptions is no heteroskedasticity. From looking at the data in (Fig 3) there doesn't seem to be heteroskedasticity for any of the data points.

Figure 3

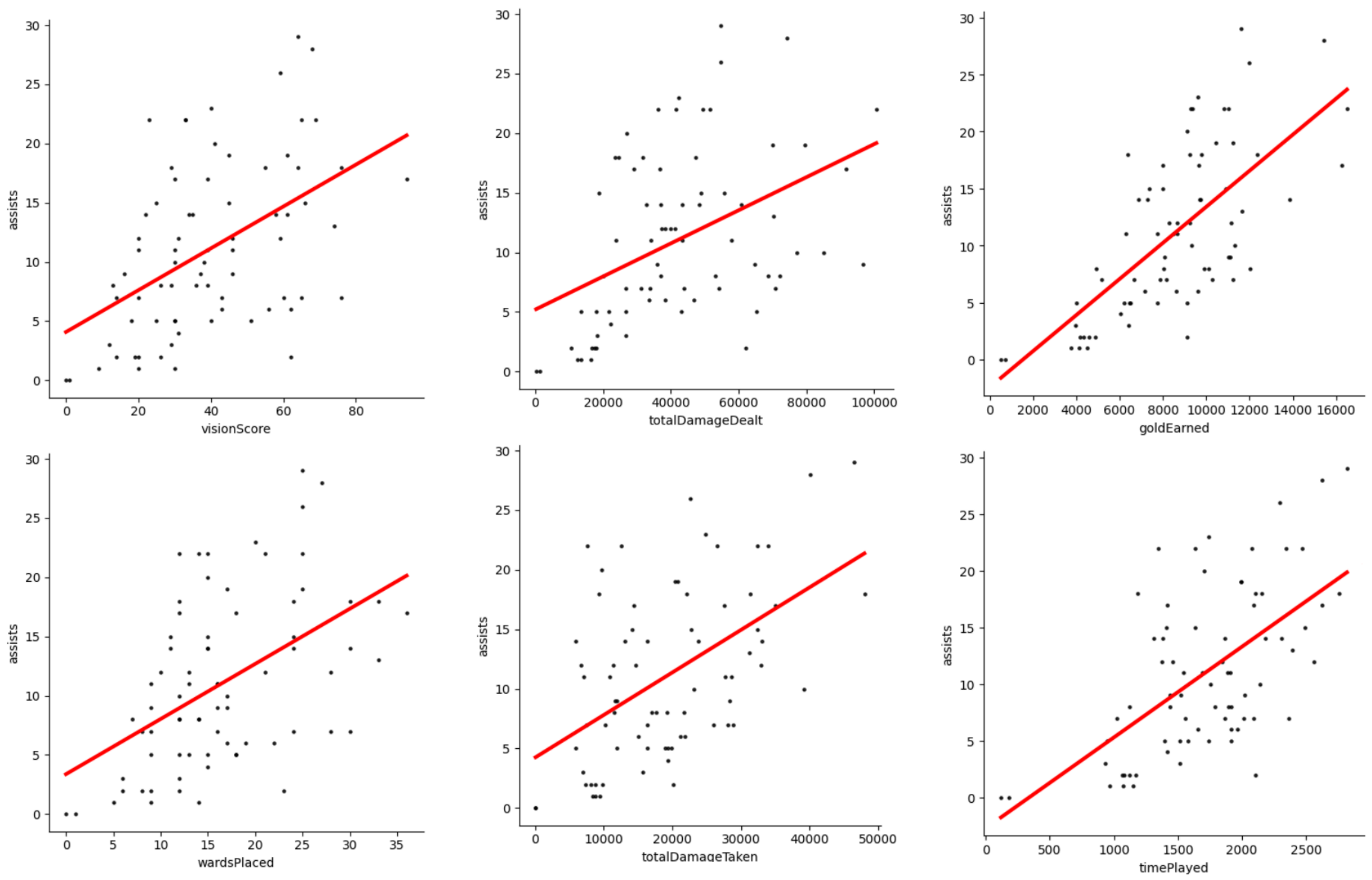
Evenly Distributed Variables Plotted Against Assists



These plots helped to validate some of the linear regression assumptions. None of these variables, so far, need to be omitted due to failing assumptions. To be sure, All of these variables were plotted using Seaborn's 'lmplots.' (Fig 4)

Figure 4

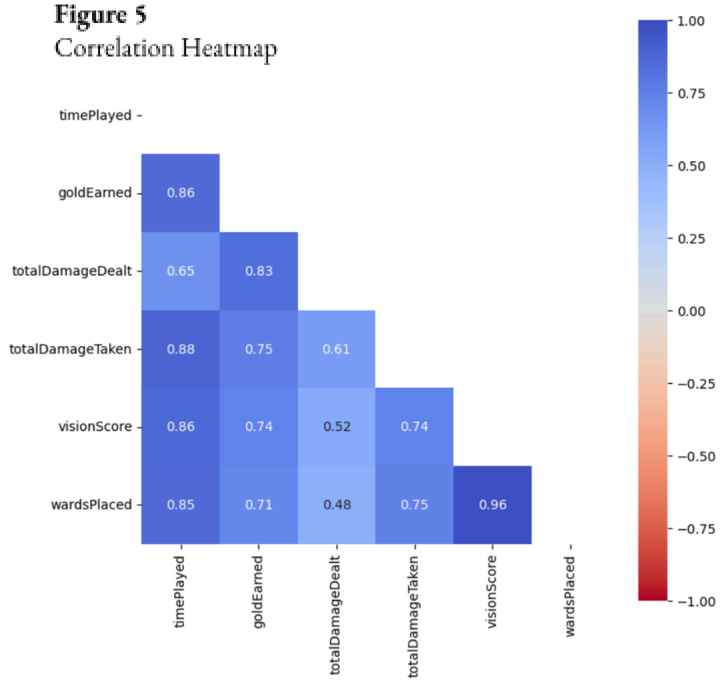
Seaborn lmplots



All of the variables look to have a strong enough linear relationship to be used in the linear regression model. Furthermore, most of the linear regression assumptions have been addressed. The next linear regression assumption to be addressed is multicollinearity. Most likely this dataset has some multicollinearity due to the nature of how close the games are time wise as well as how similar games tend to be. For this analysis I will allow multicollinearity as its impact shouldn't be too negative. Furthermore, it would be nearly impossible to avoid multicollinearity with this dataset. I will not use transformations on the dataset to address the multicollinearity but rather take it as a potential weakness to the conclusion.

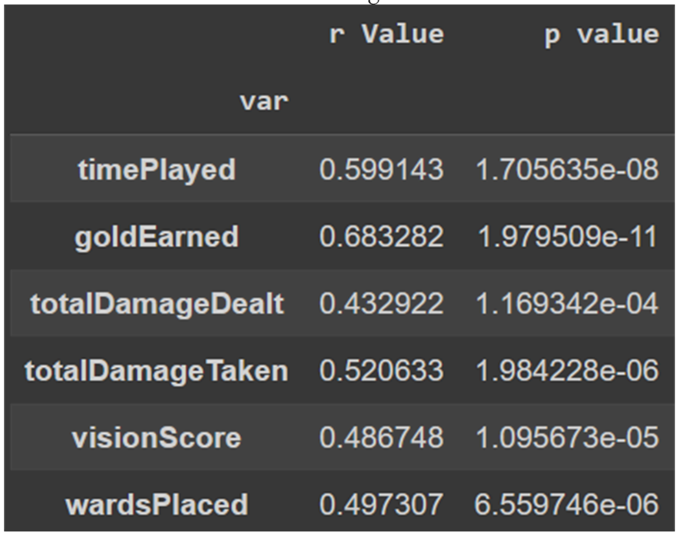
Figure 5

Correlation Heatmap



Now that the data has been reduced to the data points that could possibly work in a linear regression model, a Pearson Correlation Coefficient test will be applied to all of the data points against 'assists.' This is to understand if there is some statistical importance to the correlations. Given the correlation heatmap from earlier, there will most likely be a statistical importance, however, it is still important to verify. After the tests, I created a table with the r and p values of the variables against assists (Table 1)

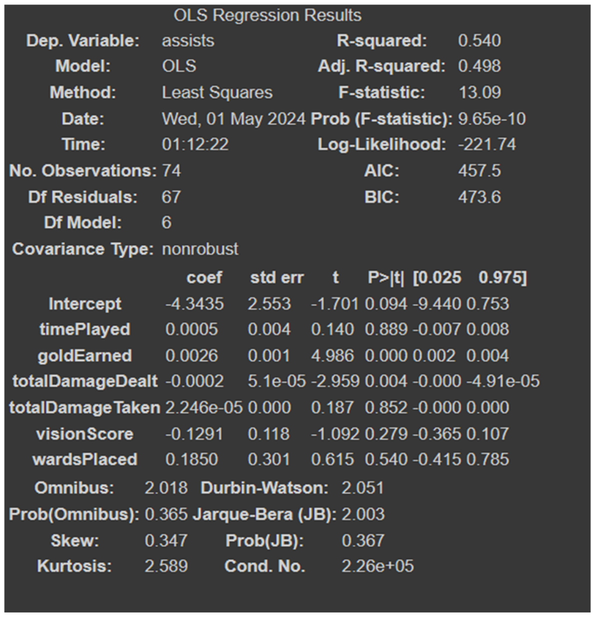
Table 1
Results of Pearson Correlation Test Against Assists



var	r Value	p value
timePlayed	0.599143	1.705635e-08
goldEarned	0.683282	1.979509e-11
totalDamageDealt	0.432922	1.169342e-04
totalDamageTaken	0.520633	1.984228e-06
visionScore	0.486748	1.095673e-05
wardsPlaced	0.497307	6.559746e-06

From this table we can gather some important information about these variables. First, looking at the r value (pearson correlation coefficient) there is some correlation. The closer to 1, the higher the correlation. In this case most variables are around 0.5 which is still significant. Gold earned is nearly 0.7 which means that it is more related than the rest. For the p values, they need to be less than 0.01 to be statistically significant. In this case, all of the p values are well below 0.01. From this test, it can be reasonably said that these variables are good candidates for the linear regression model.

Figure 6
Linear Regression Model With All 6 Variables

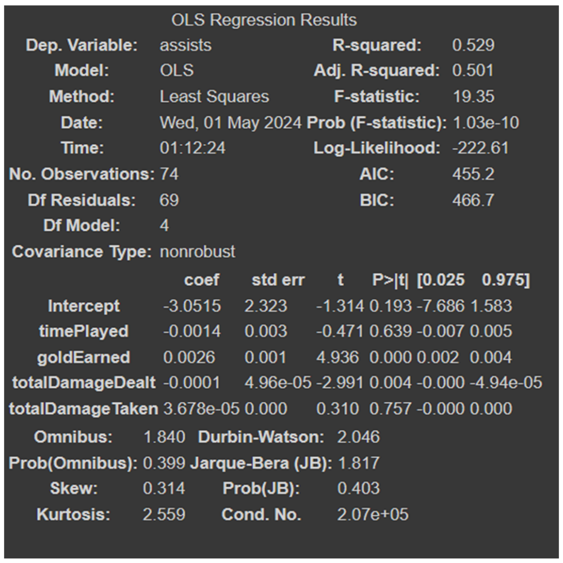


```
                        OLS Regression Results
  Dep. Variable:    assists                    R-squared:      0.540
         Model:     OLS                    Adj. R-squared:     0.498
        Method:     Least Squares              F-statistic:    13.09
          Date:     Wed, 01 May 2024 Prob (F-statistic):       9.65e-10
          Time:     01:12:22              Log-Likelihood:      -221.74
No. Observations:   74                             AIC:        457.5
  Df Residuals:     67                             BIC:        473.6
      Df Model:     6
Covariance Type:    nonrobust
                      coef     std err    t     P>|t|  [0.025   0.975]
      Intercept      -4.3435   2.553   -1.701  0.094  -9.440   0.753
     timePlayed       0.0005   0.004    0.140  0.889  -0.007   0.008
     goldEarned       0.0026   0.001    4.986  0.000   0.002   0.004
totalDamageDealt     -0.0002   5.1e-05 -2.959  0.004  -0.000  -4.91e-05
totalDamageTaken  2.246e-05    0.000    0.187  0.852  -0.000   0.000
    visionScore      -0.1291   0.118   -1.092  0.279  -0.365   0.107
    wardsPlaced       0.1850   0.301    0.615  0.540  -0.415   0.785
   Omnibus:       2.018  Durbin-Watson:     2.051
Prob(Omnibus):    0.365  Jarque-Bera (JB):  2.003
     Skew:        0.347     Prob(JB):       0.367
   Kurtosis:      2.589     Cond. No.       2.26e+05
```

For the linear regression model, I began by including all of the variables in the model where assists is a function of time played, gold earned, total damage dealt, total damage taken, vision score, and wards placed. This created a linear regression model with an adjusted r squared of 0.498 (Fig. 6). It is important to try other models to attempt for a better R squared. The next model is the same as the first but with vision score and wards placed omitted as these were the weakest from earlier testing. This resulted in an Adj. R squared of 0.501 (Fig. 7) which is an improvement over the earlier model.

Figure 7

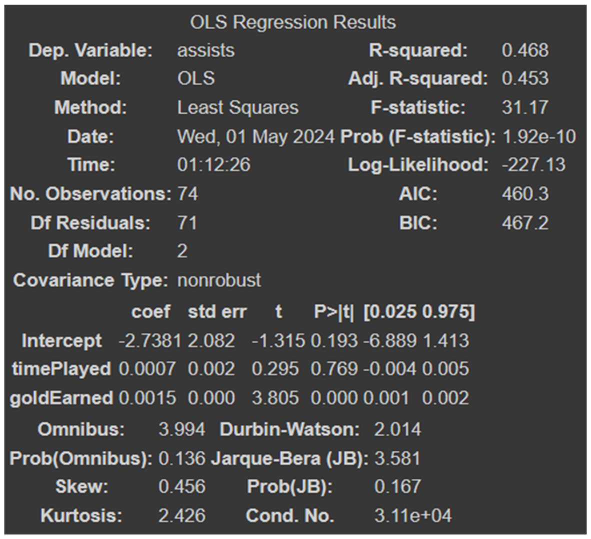
Linear Regression Model Without Vision Score and Wards Placed



```
                    OLS Regression Results
  Dep. Variable:    assists           R-squared:          0.529
      Model:        OLS               Adj. R-squared:     0.501
     Method:        Least Squares     F-statistic:        19.35
      Date:         Wed, 01 May 2024  Prob (F-statistic): 1.03e-10
      Time:         01:12:24          Log-Likelihood:     -222.61
No. Observations:   74                AIC:                455.2
  Df Residuals:     69                BIC:                466.7
    Df Model:       4
Covariance Type:    nonrobust
                    coef       std err   t       P>|t| [0.025  0.975]
    Intercept       -3.0515    2.323     -1.314  0.193 -7.686  1.583
   timePlayed       -0.0014    0.003     -0.471  0.639 -0.007  0.005
   goldEarned       0.0026     0.001     4.936   0.000 0.002   0.004
totalDamageDealt    -0.0001    4.96e-05  -2.991  0.004 -0.000  -4.94e-05
totalDamageTaken    3.678e-05  0.000     0.310   0.757 -0.000  0.000
   Omnibus:        1.840  Durbin-Watson:     2.046
Prob(Omnibus):     0.399  Jarque-Bera (JB):  1.817
     Skew:         0.314  Prob(JB):          0.403
   Kurtosis:       2.559  Cond. No.          2.07e+05
```

Figure 8

Linear Regression Model With Only Time Played and Gold Earned



```
                    OLS Regression Results
  Dep. Variable:    assists           R-squared:          0.468
      Model:        OLS               Adj. R-squared:     0.453
     Method:        Least Squares     F-statistic:        31.17
      Date:         Wed, 01 May 2024  Prob (F-statistic): 1.92e-10
      Time:         01:12:26          Log-Likelihood:     -227.13
No. Observations:   74                AIC:                460.3
  Df Residuals:     71                BIC:                467.2
    Df Model:       2
Covariance Type:    nonrobust
              coef     std err  t       P>|t|  [0.025  0.975]
  Intercept   -2.7381  2.082    -1.315  0.193  -6.889  1.413
 timePlayed   0.0007   0.002    0.295   0.769  -0.004  0.005
 goldEarned   0.0015   0.000    3.805   0.000  0.001   0.002
   Omnibus:        3.994  Durbin-Watson:     2.014
Prob(Omnibus):     0.136  Jarque-Bera (JB):  3.581
     Skew:         0.456  Prob(JB):          0.167
   Kurtosis:       2.426  Cond. No.          3.11e+04
```

For the next model, I tried removing the next least strong variables which were total damage dealt and total damage taken. This left a model with only time played and gold earned (Fig. 8). This model had a much worse Adj. R squared of only 0.453. This means that the strongest model was the model where assists were a function of time played, gold earned, total damage dealt, and total damage taken.

Potential Counter Arguments and Weaknesses

From a Linear Regression standpoint, there are two weaknesses. The first, mentioned earlier, is there is some multicollinearity. This is due to the nature of the data being so similar because of how similar League of Legends matches are. This could potentially be solved with more data but most likely not.

For the next weakness, linear regression assumes no autocorrelation. However, this data is most likely auto correlated. That is because all of the observations for each match come from the same game and therefore the data for each game is correlated with each other. One of the most problematic variables for autocorrelation is how long each game is. In general, a longer game will increase all other variables. In this data set, all of the games were a different length. However, this autocorrelation was acceptable given that most games remain within the range of twenty to forty minutes long.

Since the linear regression model fails these two assumptions, it isn't incredibly strong. However, these assumptions aren't nearly as important as the other assumptions and therefore the model is still statistically valuable.

One of the other weaknesses to this analysis is how few matches were analyzed. There were only 74 matches that were analyzed after the data was cleaned. While more matches could have been pulled from the API, it wasn't beneficial to the data. This is because 100 matches ago, my playstyle was different than it currently is. This is because I was in a different skill bracket than I am now. Therefore, the relationships between the data may not apply to my current playstyle.

Conclusion

The goal of this analysis was to answer the question: what components of my League of Legends gameplay contribute to getting more assists? Answering this will help me better my game play so that I may join the ranks of the best players. These answers may also help other players who play support like me to better their own gameplay. League of Legends is a globally recognized esports title that brings in millions of dollars of revenue for many players. It is not just another game and by getting good, one can unlock potential doors for success.

After pulling data from the Riot API, cleaning the data, analyzing the data, and creating a linear regression model, the original question has been answered. The answer to the question is: an increase in time played, gold earned, total damage dealt, and total damage taken has a positive impact on the number of assists I get. Therefore, if I want to improve my games, I should look to increase these variables. Prolonging the game makes sense as the longer I play, the more assists I should get. As for dealing more damage, I should try to play support champions with a high damage output and poke in lane as much as possible. As for earning more gold, I can prioritize poking in lane, getting objectives, and farming when appropriate. Taking more damage may seem hindering but the data shows that it helps with getting more assists. This may be because being aggressive usually causes one to take more damage. Therefore, I should try to engage the enemy and poke more often even if it causes me to take more damage.

Overall, after this data analysis, I have a better understanding of what components of my gameplay help me to get more assists. Improving these components will in turn make me a better support. I will implement these changes into my gameplay and could possibly do a follow up analysis to see if the results were accurate.

Sources

Python Notebook:
https://colab.research.google.com/drive/18lz_yy9jeBVWx6mBMp_tzWgs5nPGuYvN?usp=sharing
Riot API:
https://developer.riotgames.com/apis
Riot Watcher (Riot API Python Wrapper):
https://github.com/pseudonym117/Riot-Watcher
Data Collected from Riot API:
https://github.com/Dpere22/Spring2024DataScience