**Python Simulation for Predicting NFL Scores vs. Knime Predictive Regression Model**

Lorenzo Muranelli | INLS 625 | Professor Rajasekar | UNC Chapel Hill

November 19, 2020

**Background**:

The goal of this project is to create a multi-linear regression model for predicting NFL scores that can be run to make more accurate predictions about future NFL games and can potentially be applied to other sports. I was initially going to make a Monte Carlo simulation to predict future games, which relies on random draws and accounts for risk factors that a historical simulation cannot do.[[1]](#footnote-1) However, this process is both time consuming and costly as it takes thousands of runs to achieve accurate (or close to accurate) predictions. In the future, I would like to expand on the current project to incorporate the Monte Carlo simulation.[[2]](#footnote-2) This specific example runs a simulation to predict NBA player lines and can potentially be adapted for NFL scores, but requires adjustments. I may also utilize a project that uses a Monte Carlo simulation to predict NFL outcomes in R that I found on GitHub to help me build a future model.[[3]](#footnote-3) The final example uses an extensive predictive NFL betting model I found on Github and could also be adjusted to my current model.[[4]](#footnote-4)

In order to complete this project, I relied heavily on a prior version of this project with the same goal, but I altered the variables and team (Buffalo Bills instead of New England Patriots), as well as the focus of the results.[[5]](#footnote-5) The prior version of this project primarily focused on weather as a variable that impacts NFL scores, but found that there was not a positive correlation. Thus, I removed this variable and instead focused on betting variables such as the over/under line and the spread favorite, as well as stadium neutrality, which tracks if there was home field advantage. I utilized a dataset I found on Kaggle, which incorporate NFL score and betting odds data from 1966-2020.[[6]](#footnote-6) I used Python for the first part of the project and then used Knime, an open source platform for data analytics, providing a user-friendly graphical workbench for the entire analysis process: data access, data transformation, initial investigation, powerful predictive analytics, visualization, and reporting. I then drew conclusions based on the results of each of these tools and compared them. I have outlined the steps I took as follows:

**Goal**: Create a model that predicts how may points will be scored in a game for each NFL team (Buffalo Bills as use case example in this scenario).

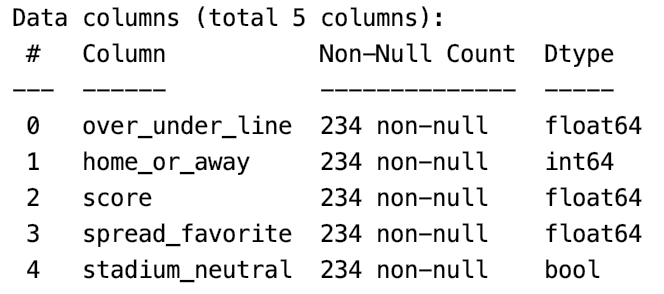
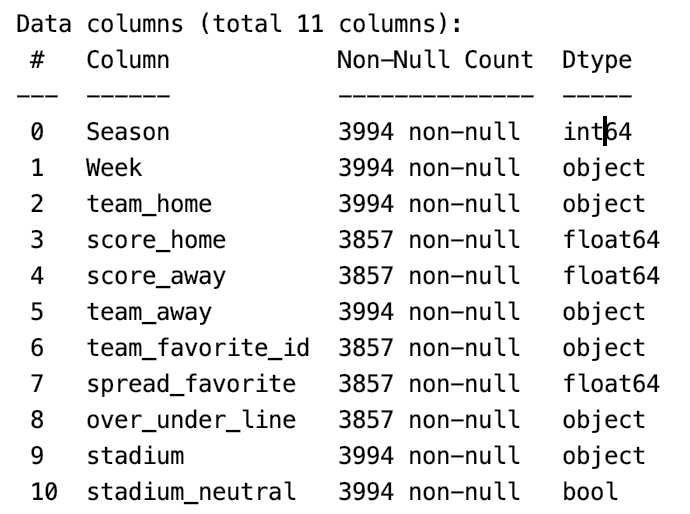
**Steps**:

1. Aggregate NFL scores and betting data (over/under line, spread) dataset from Kaggle
   1. The data shows NFL game results from 1966-2020 and betting odds data since 1979. The dataset was created from numerous sources including ESPN, NFL.com, and Pro Football Reference.
2. Develop Python program that compiles data from the dataset to examine correlation between different variables on the final scores
   1. Python libraries used: Import pandas (data manipulation), numpy (data manipulation), matplotlib (plot data), scikit-learn (machine learning, modelling), statsmodels (regression), and seaborn (pairplot)
   2. Load the dataset (after the year 2005, games before 2005 are not of interest/not relevant for this project)
   3. Drop null values (dropna)
   4. Use the Buffalo Bills as a use case (favorite team). Can be altered in the future for other NFL teams
   5. Select variables to index: **'over\_under\_line'**,**'home\_or\_away'**, **‘spread\_favorite’**, **‘stadium\_neutral’** as explanatory variables(X) to predict target variable (Y) **‘score’**
   6. Change to a float value data type (astype(‘Float64’))
   7. Create a pairplot to examine correlations/conduct analysis
      1. Then print the correlation metrics (stronger correlation for values closer to 1; weaker correlation for values closer to 0)
   8. Create training/test set using pd.dataframe and the appropriate independent (X) and dependent (Y) variables
   9. Build linear regression model (obtain slope and line of best fit), R-squared, Root mean squared error (RMSE)
   10. Generate residual plot of actual vs. predicted values
   11. Produce a correlation matrix for indexed values
3. Draw conclusions
4. Use Knime: Simple regression tree to gauge whether the spread favorite accurately predicted the winner of NFL games: two trees (one for home team score and one for away team score since I could not combine the score value as I did in Python)
   1. Load the data using the Excel Reader (XLS). I had to use the original CSV file because the altered Excel dataset would not load. I excluded columns I did not need (same variable columns as Python). However, I had to include NFL games from all years (1966-2020). For the final results, I only looked at the results for recent data.
   2. Split into two partitions
   3. Partition both the data: 80% training and 20% test data using stratified sampling on the team favorite id variable
   4. Simple regression tree learner: included variables schedule season, score away, team favorite id, spread favorite, over/under line, and stadium neutral for the home score prediction. The same variables were included for the score away prediction set except the score away variable was swapped with the score home variable
   5. Simple regression tree predictor: one for each expected score outcome
   6. Column filter: For the actual and predicted outcomes
   7. Line plot: to visualize the performance of the simple regression tree
   8. Numeric scorer: to score the prediction using the spread favorite variable
5. Draw conclusions
6. Compare Python and Knime results and draw final conclusions

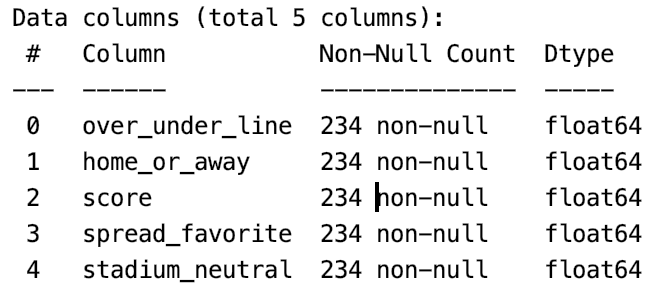
**Conclusions**:

After using Python for the linear regression model, I generated the following results:

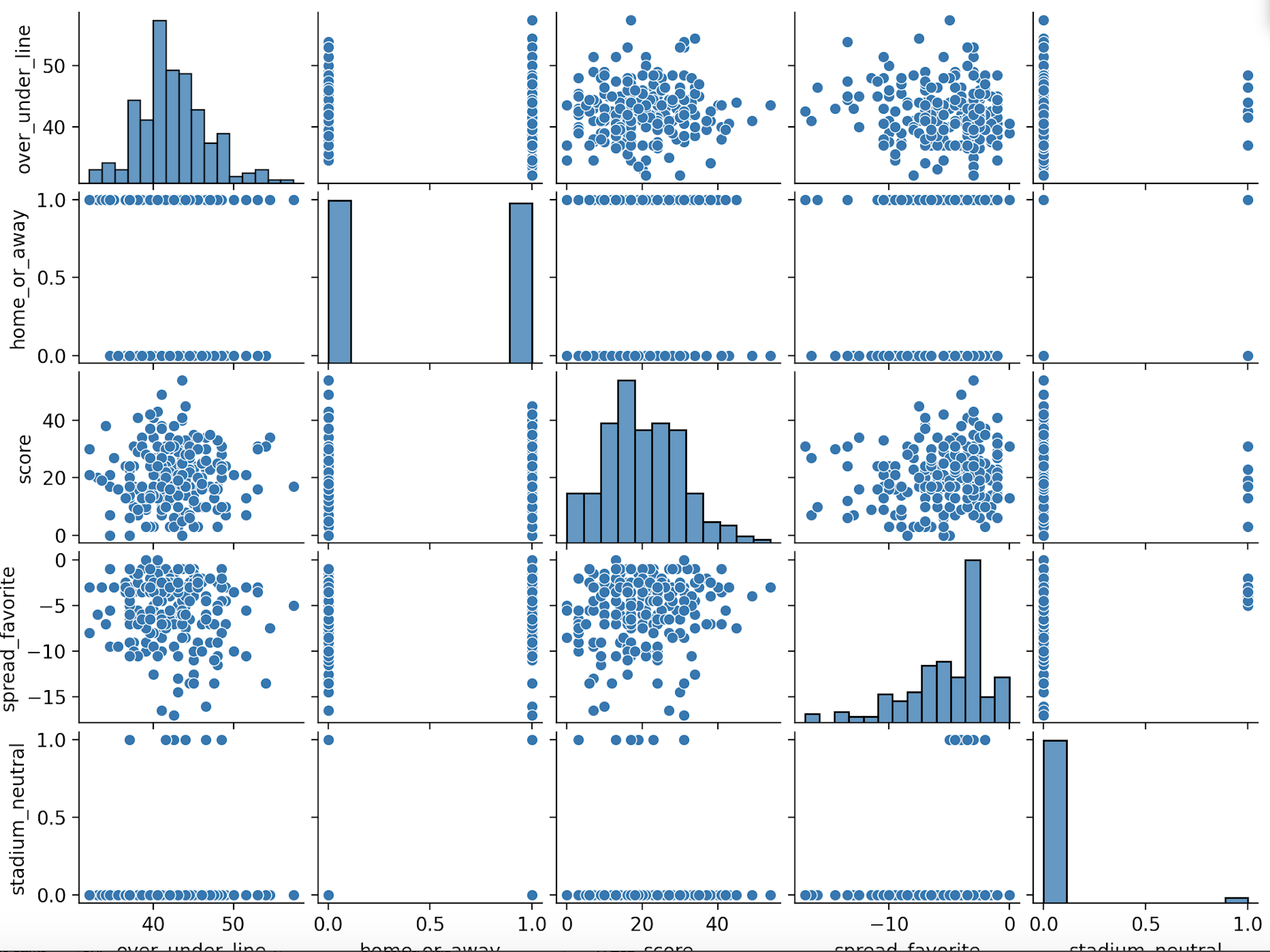
**Figure 1**: Data columns prior to data type conversion to ‘Float64’ for indexed variables of focus.



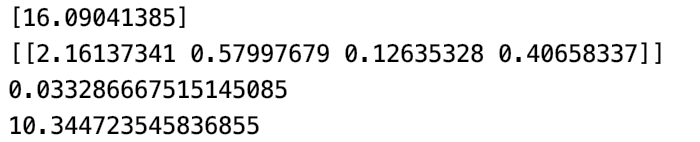
**Figure 2**: Data columns after adjusting data types to ‘Float64’ for indexed variables of focus (consistency).



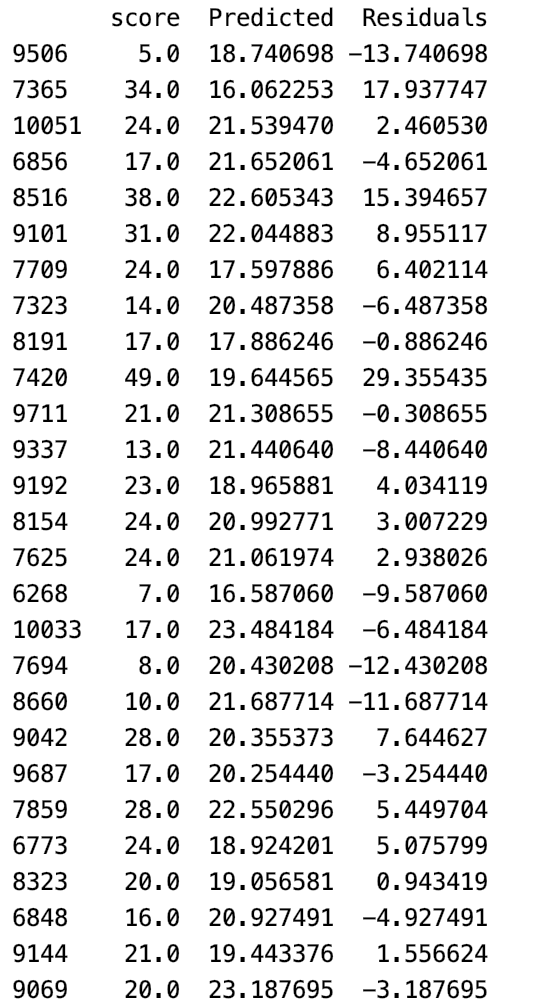
**Figure 3**: Pairplot: over/under, home or away, score, spread favorite, stadium neutral. A pairplot creates a grid of axes such that each variable in the data will be shared in the y-axis across a single row and in the x-axis across a single column. The pairplot shows the relationship for (n,2) combination of variables in a dataFrame as a matrix of plots and the diagonal plots are the univariate plots. From the pairplot, it appears as though score and spread favorite are positively correlated, over/under line and spread favorite are positively correlated, whereas over/under line and score are negatively correlated. However, more statistical tests must be run to know for certain.



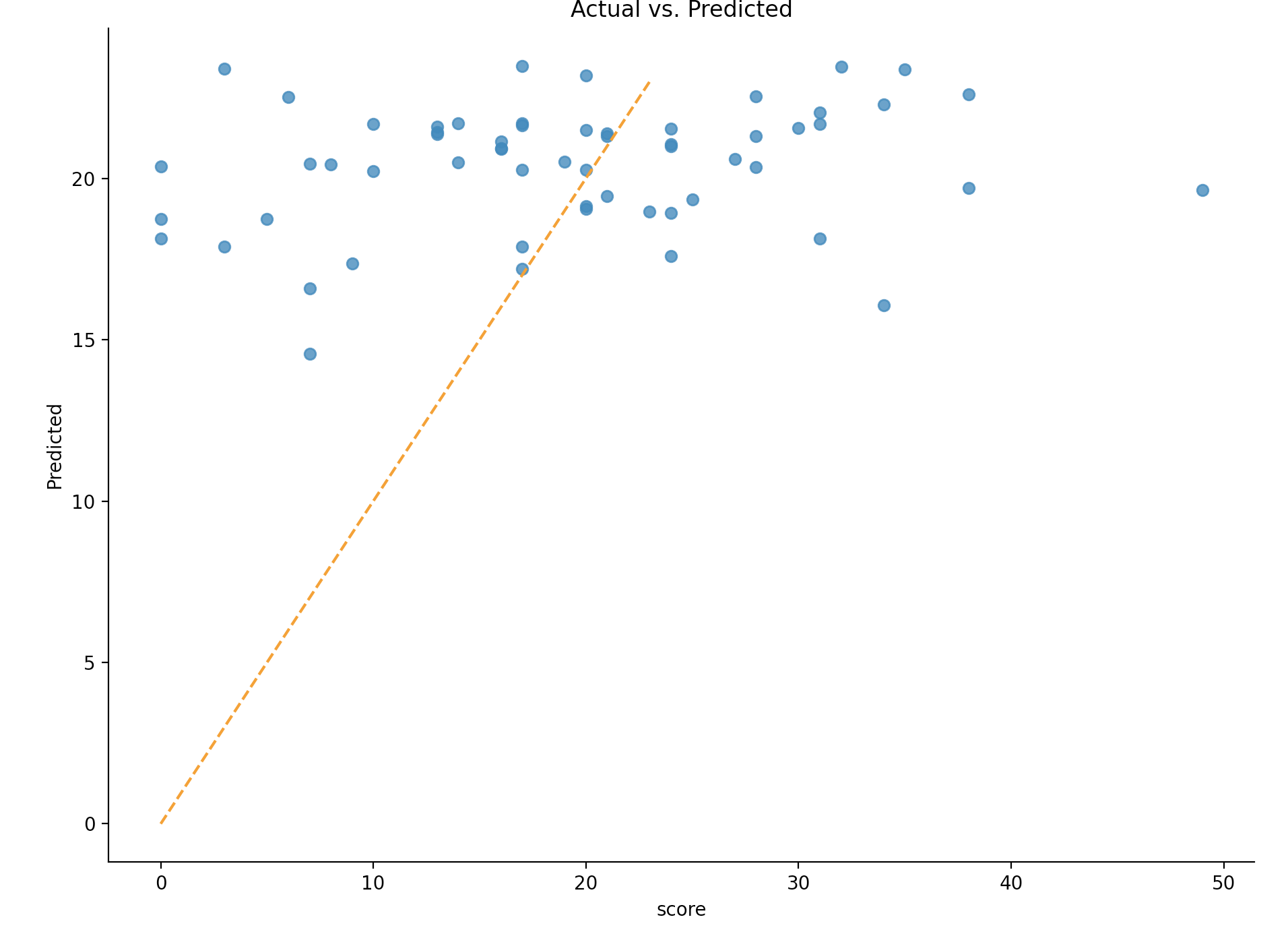
**Figure 4**: Correlation metrics. Based on these results, the estimated intercept is 16.09 and the regression coefficients are ^β1 = 2.2, ^β2 = 0.58, ^β3 = 0.13, and ^β4 = 0.41. This suggests that when holding other variables constant, for every additional increase in over/under line, spread, home/away, and stadium neutrality, there is an associated point increase in the mean score for each respective variable. Next, the R-squared value, which predicts the percent of variance explained by the model on a 0-100% scale indicates that the variance is 3%, which means that the model explains little to none of the variability of the response data around the mean. Lastly, the RMSE, or the square root of the variance of the residuals, which indicates absolute fit of the model to the data, is 10.345.

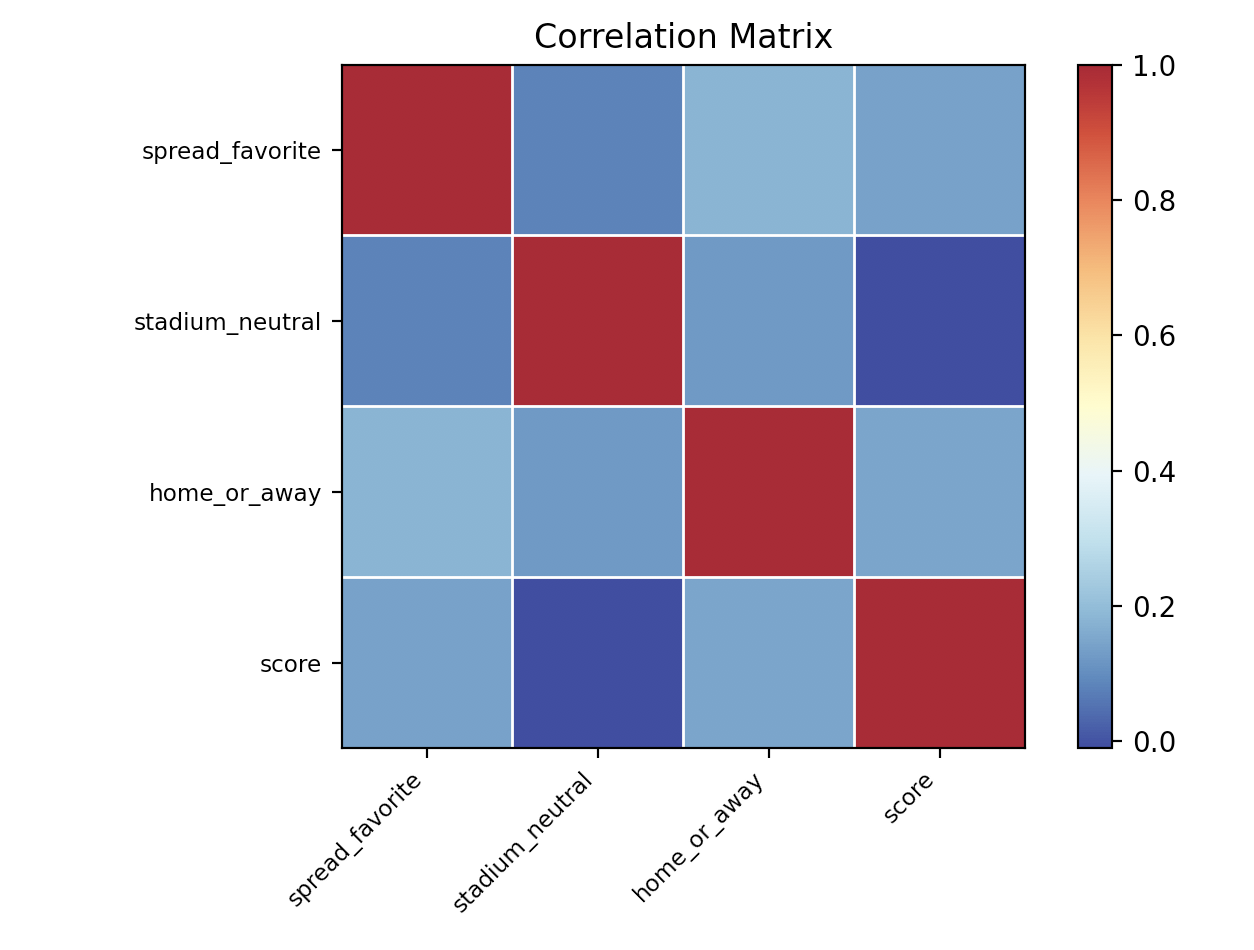


**Figure 5**: Residuals table with the Score, Residuals, and Predicted data points. The data indicates that there is a linear relationship between the explanatory and target variables and assumes that the predictors are additive.



**Figure 6**: Residual plot. The results indicate that there is not an even spread around the diagonal line—a cluster around the diagonal line would indicate that our assumptions were correct.

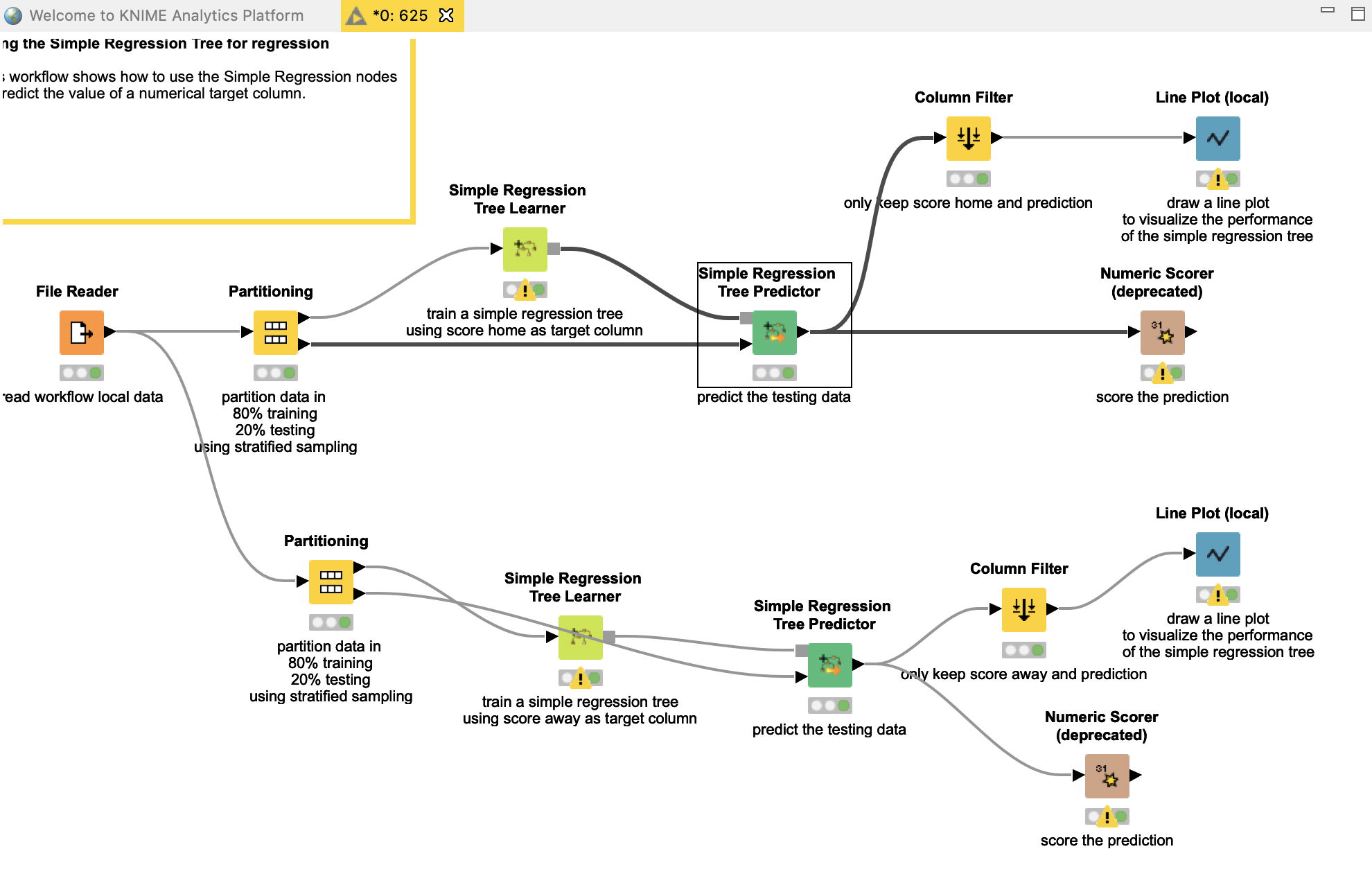


**Figure 7**: Our final figure shows a correlation matrix for the variables of interest. As you can see, there is not a strong correlation between any of the variables. According to the matrix, stadium neutrality and score are not correlated and spread favorite and home or away are a more correlated, but still do not exhibit a strong correlation.

**Takeaway**: Perhaps or simulation and model was not as accurate at predicting future NFL games/scores as I had anticipated, but there is more work to be done. Let’s see if Knime is more effective than my model.

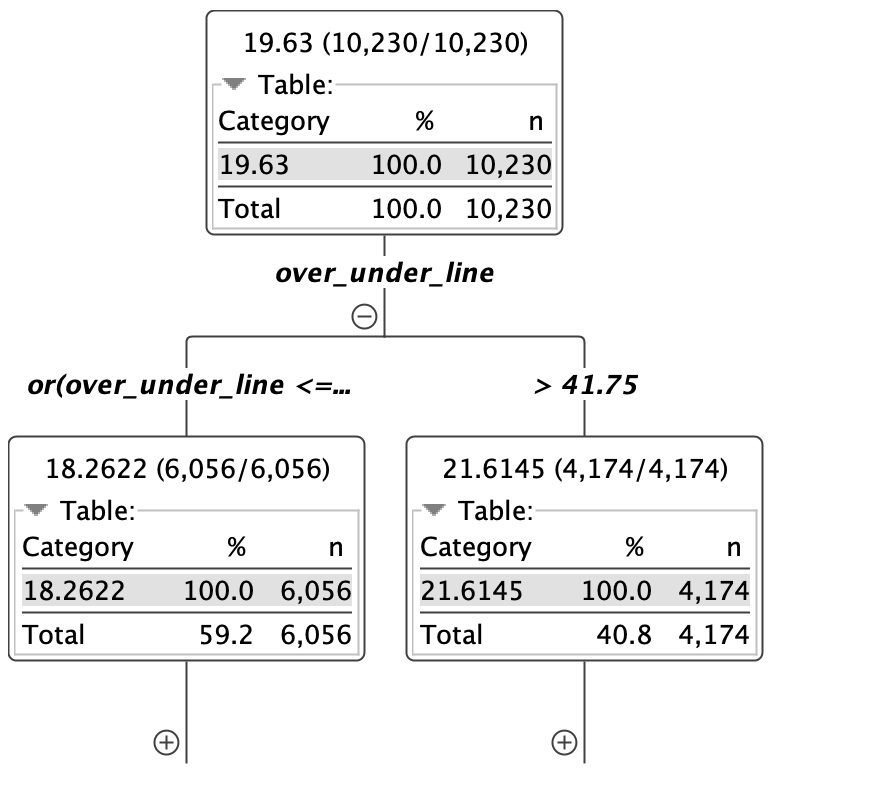
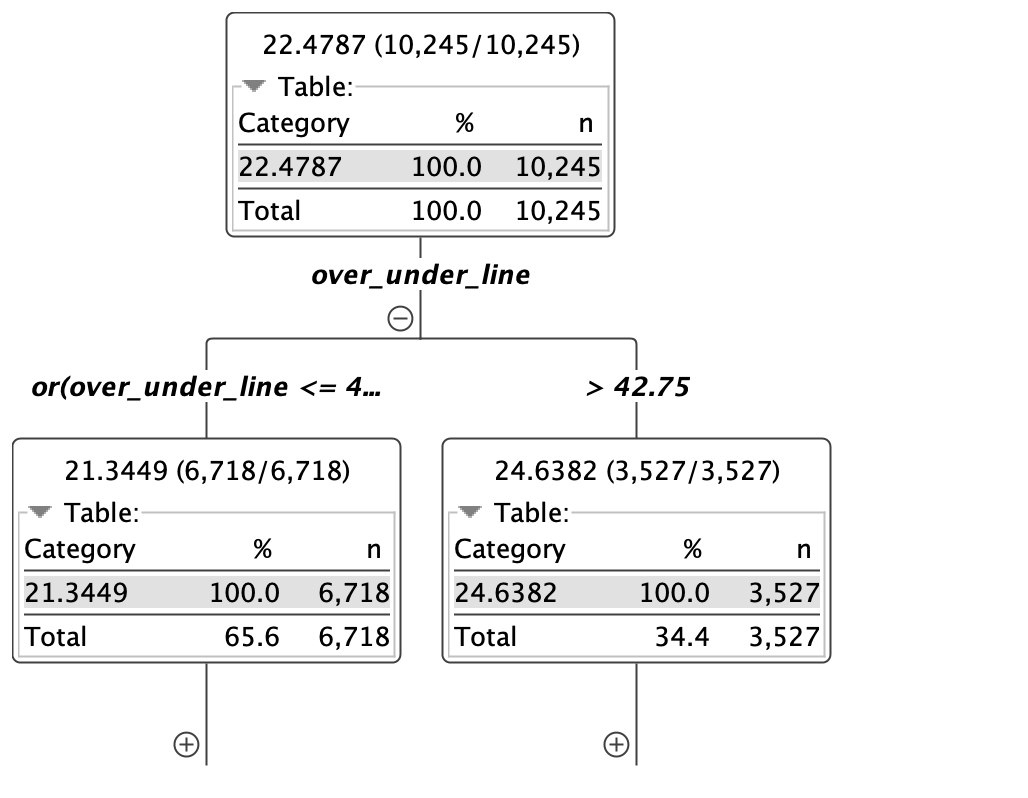
After using Knime, I generated the following results:

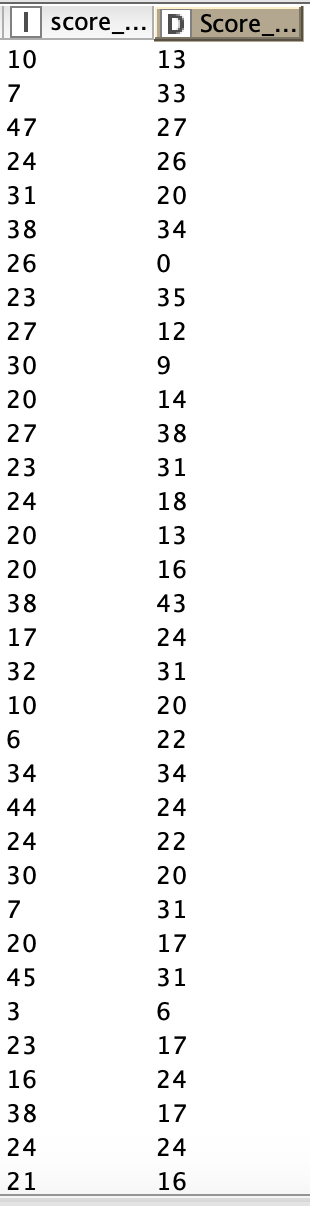
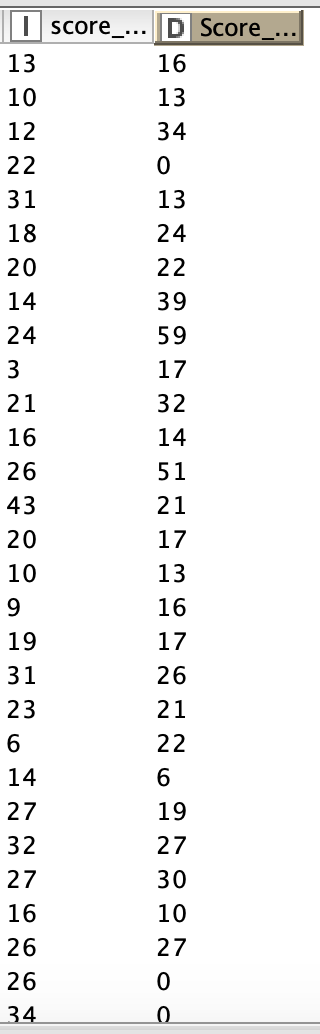
**Figure 1**: Workflow schema



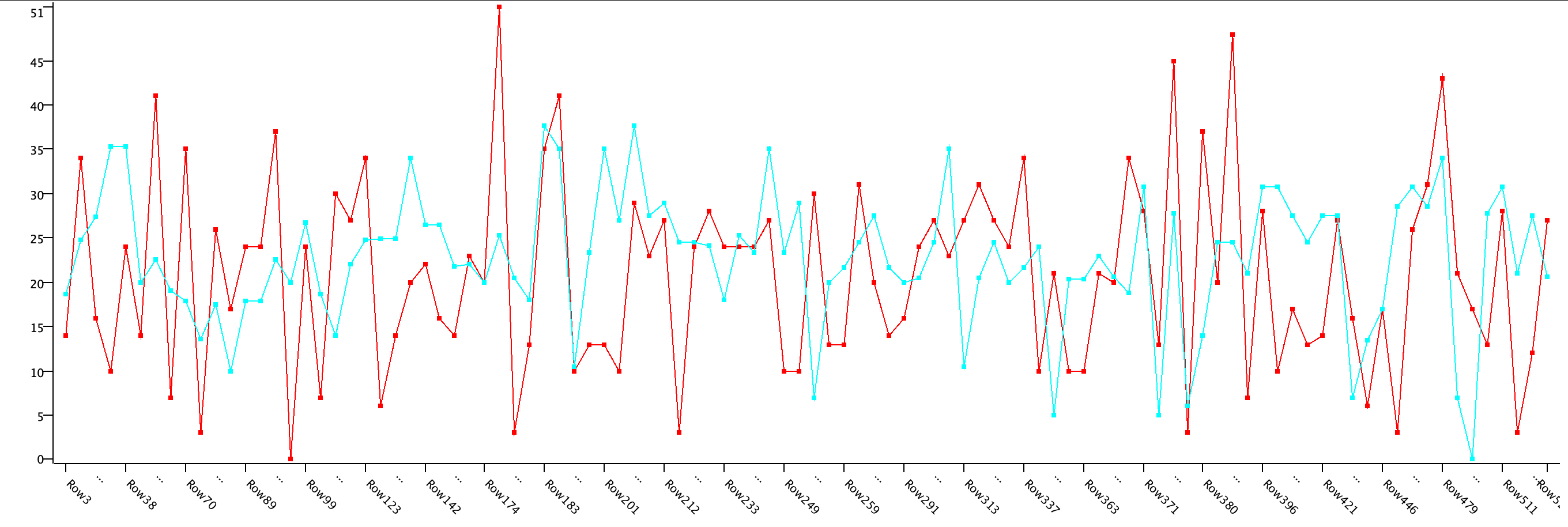
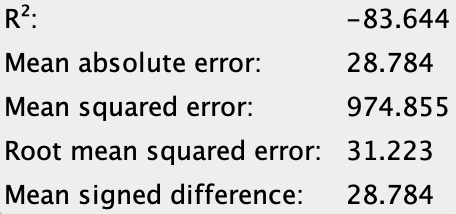
**Figure 2**: Regression Tree View

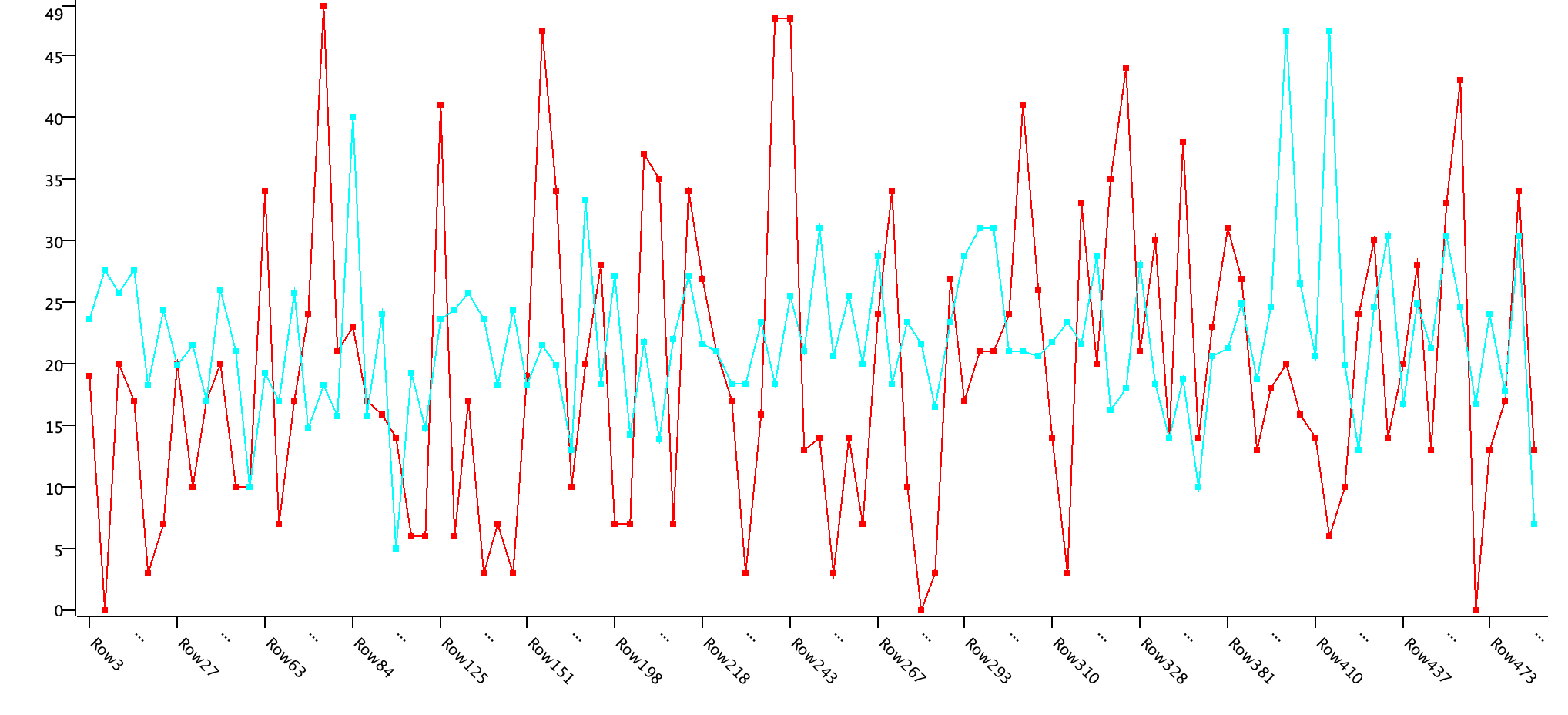
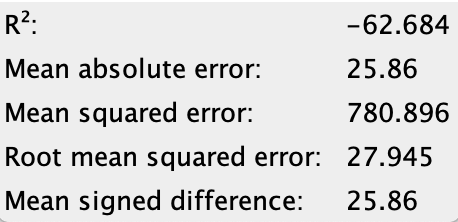
Home score away score



**Figure 3**: Predicted output for scores. Doesn’t appear to be any correlation between the two, only some of the scores are close to accurate.

**Figure 4**: Line plot data and statistics data. The scoring data appears to follow a general trend, but it is not very accurate at predicting the actual score. As you can see, the R2 value is ~83% indicating that the data was a great fit for the model (home). Likewise, the R2 value was ~63% (away). However, for both tests, the mean squared error was very high, which is not ideal.

Home

Away

**Takeaway**: Again, it is hard to reach a firm conclusion and say that Knime is an accurate predictor of NFL scores. But it was good to see that the R2 value was very high, even though the error was also high. In summary, predicting an NFL score ahead of time is a difficult endeavor—no wonder people lose so much money.

1. https://pythonprogramming.net/simple-bettor [↑](#footnote-ref-1)
2. https://towardsdatascience.com/predict-nba-player-lines-with-monte-carlo-simulation-58a1c006a6e2 [↑](#footnote-ref-2)
3. https://github.com/papagorgio23/NFL-Playoff-Sim/blob/master/NFL%20Playoff%20Monte%20Carlo.R. [↑](#footnote-ref-3)
4. https://github.com/TyWalters/NFL-Prediction-Model/blob/master/NFLBettingModelTraining.py [↑](#footnote-ref-4)
5. *https://medium.com/swlh/predicting-nfl-scores-in-python-3560ccd58cb1* [↑](#footnote-ref-5)
6. *https://www.kaggle.com/tobycrabtree/nfl-scores-and-betting-data* [↑](#footnote-ref-6)