Joshua Croasdale
Spring 2024

Capstone Project

Predicting NHL Player Performance Using Linear Regression and Neural Networks

Capstone Overview

- Gain the upper hand in sports analysis
  - **Beat the markets!**
    - Sports betting is big business in North America
    - A reliable model that outperforms the ‘money lines’ is a highly valuable tool

- Use machine modelling to make predictions on <u>impactful</u> statistical categories in NHL hockey
  - Train on available data → **predict future data**

Dataset Overview

- All player data was provided by: https://www.hockey-reference.com
- Various statistics for all NHL players from 2005 to 2023 (2024 ongoing)
- Initial EDA:
  - Checked for null values, dropped irrelevant features, renamed some features
  - Feature engineering → Created some new feature columns by manipulating already present data
- Created different "versions" of my dataset
  - <u>Career numbers</u> (groupby 'Player' and sum)
  - <u>Career per Game Played</u> (career numbers for all feature categories divided by games played)
    - This "levels the playing field" by comparing every player on a per-game basis
  - <u>Three-year weighted average</u> (taking the most recent three seasons applying weights of relative importance to each)
    - This provides a dataset that is more indicative of future performance for a given player.

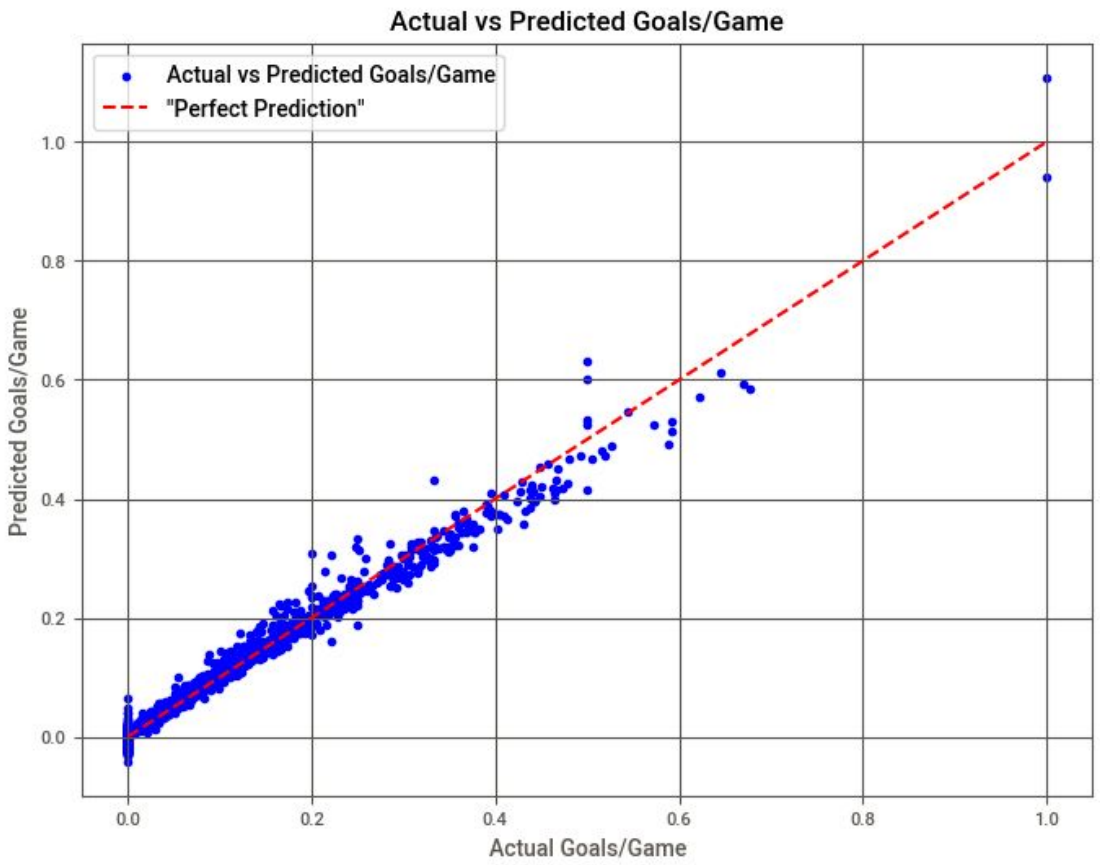
Initial Findings

- A lot of multicollinearity between independent features
  - PCA feature reduction more powerful than manual feature reduction
- Problem: **How best to arrange the data for modelling?**
  - Which data set “version” will best represent my data for modelling?
    - Career-long approach introduces the problem of age curves and experience
    - Three year weighted average dataset has the inherent advantage of placing the most importance on the most predictive samples

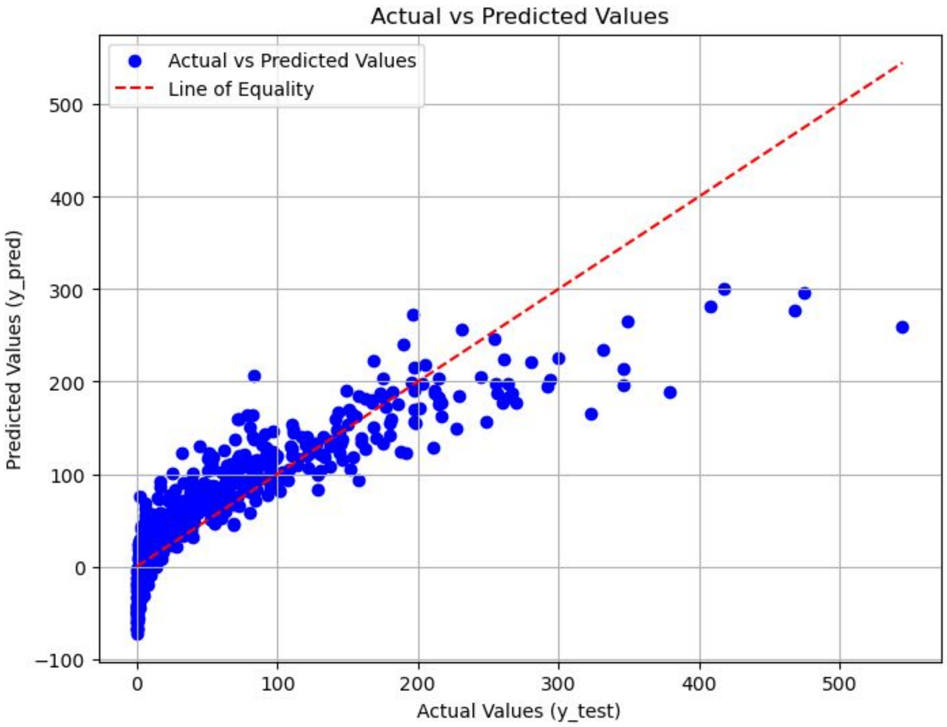
Three Year Weighted Average

- The algorithm introduced null values
  - Problem if I simply drop these nulls → small sample size
- Filled all the nulls by systematically replacing with the single season data for the given player’s most recent season
  - Sample size back up!
  - Caveat → these manually adjusted samples (players) have just one season’s worth of data
- After all this work…still decided to drop a certain amount of samples that had too few games played

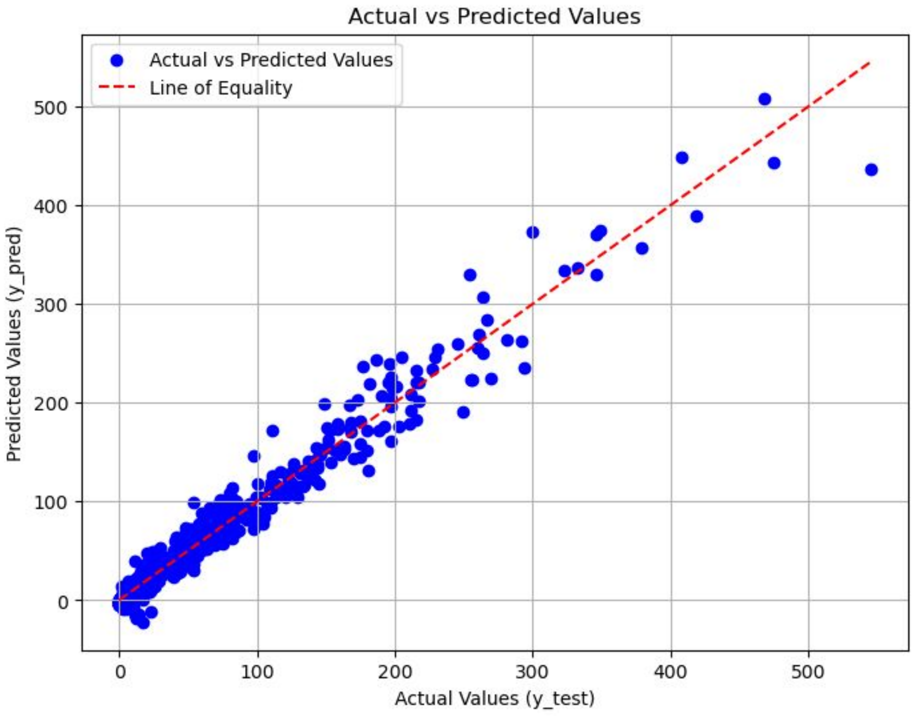
Context Matters!



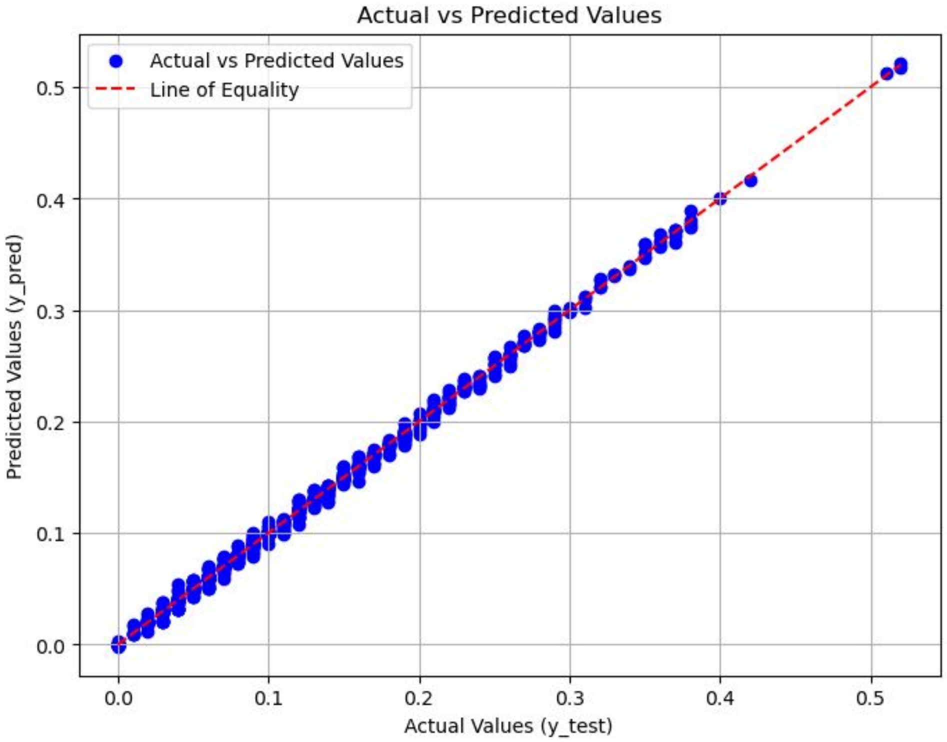
Manual Feature Reduction



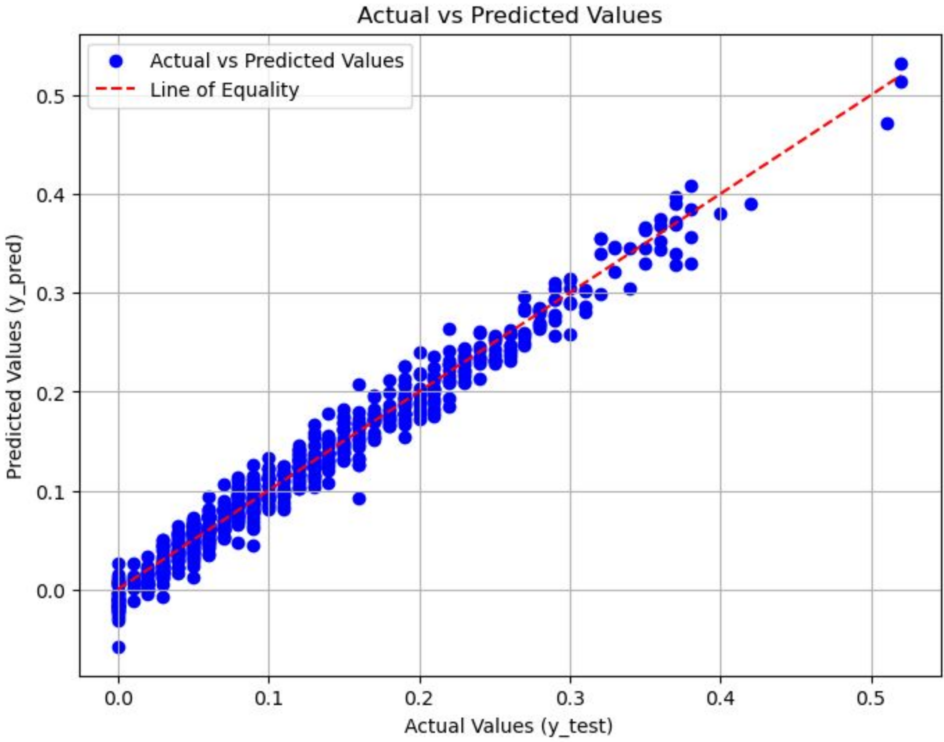
StandardScaler with PCA



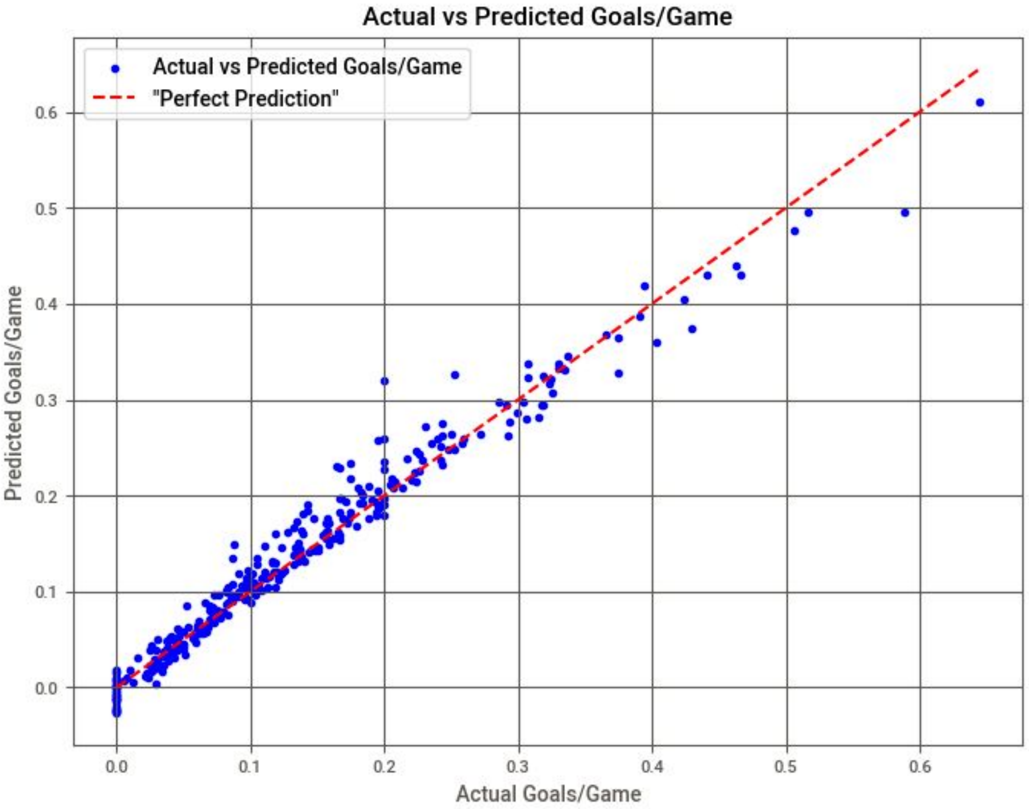
Career_per_game (fit on all features)



Career_per_game (fit with pipeline → StandardScaler, PCA)



Three_year_weighted
(fit with pipeline → StandardScaler, PCA)

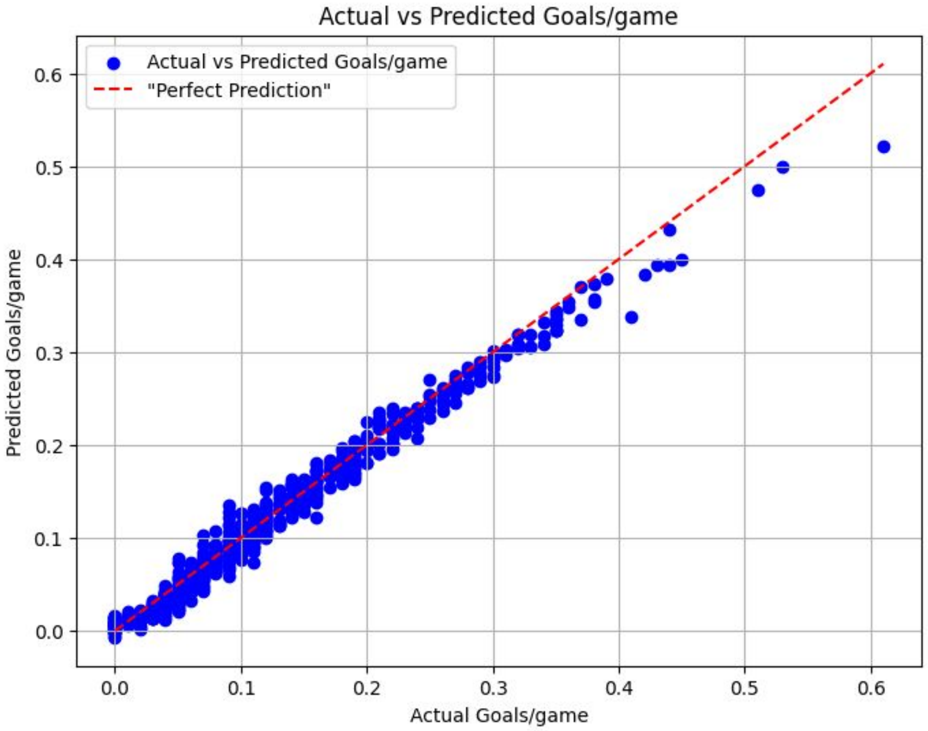


$$\mathrm{LWMA} = \frac{(P_n * W_1) + (P_{n-1} * W_2) + (P_{n-2} * W_3) \ldots}{\sum W}$$

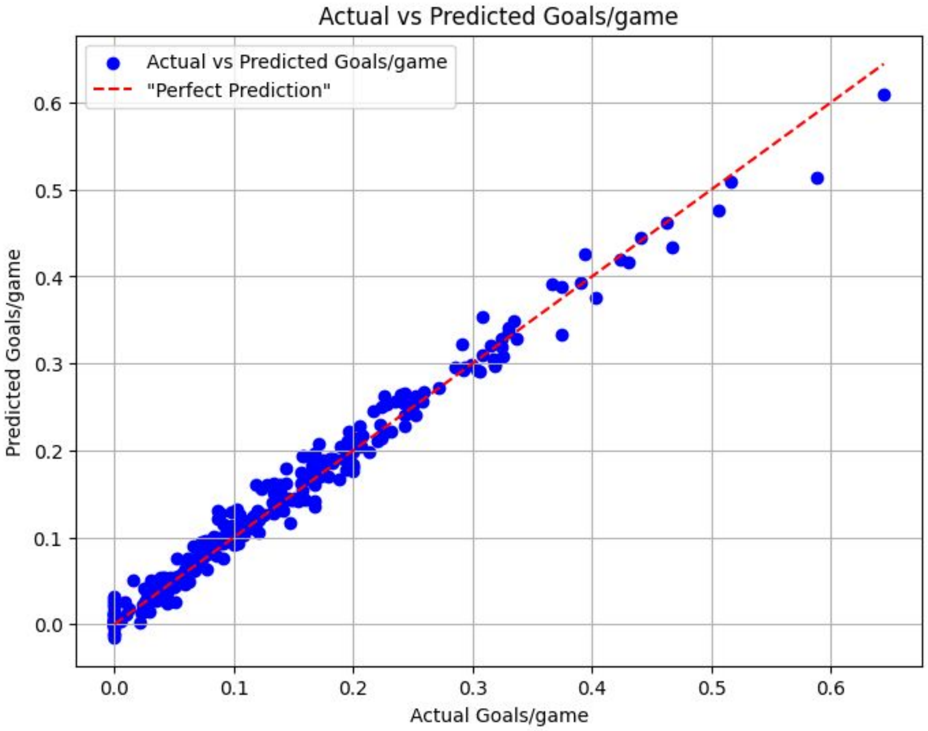
Where:

- P = Goals/game for that year
- n = The most recent year, n-1 is the prior year, and n-2 is two years prior
- W = The assigned weight to each year, with the highest weight assigned to the most recent year and descending from there

Neural Network fit to Per-Game Dataset



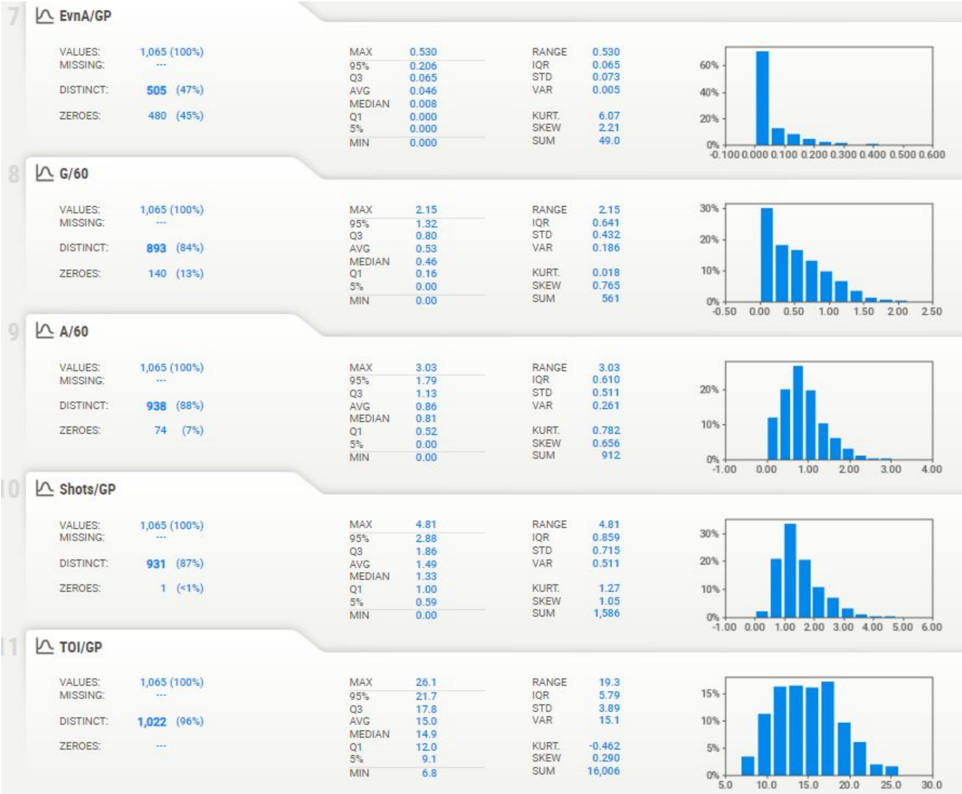
Neural Network fit to 3 Year Weighted Dataset



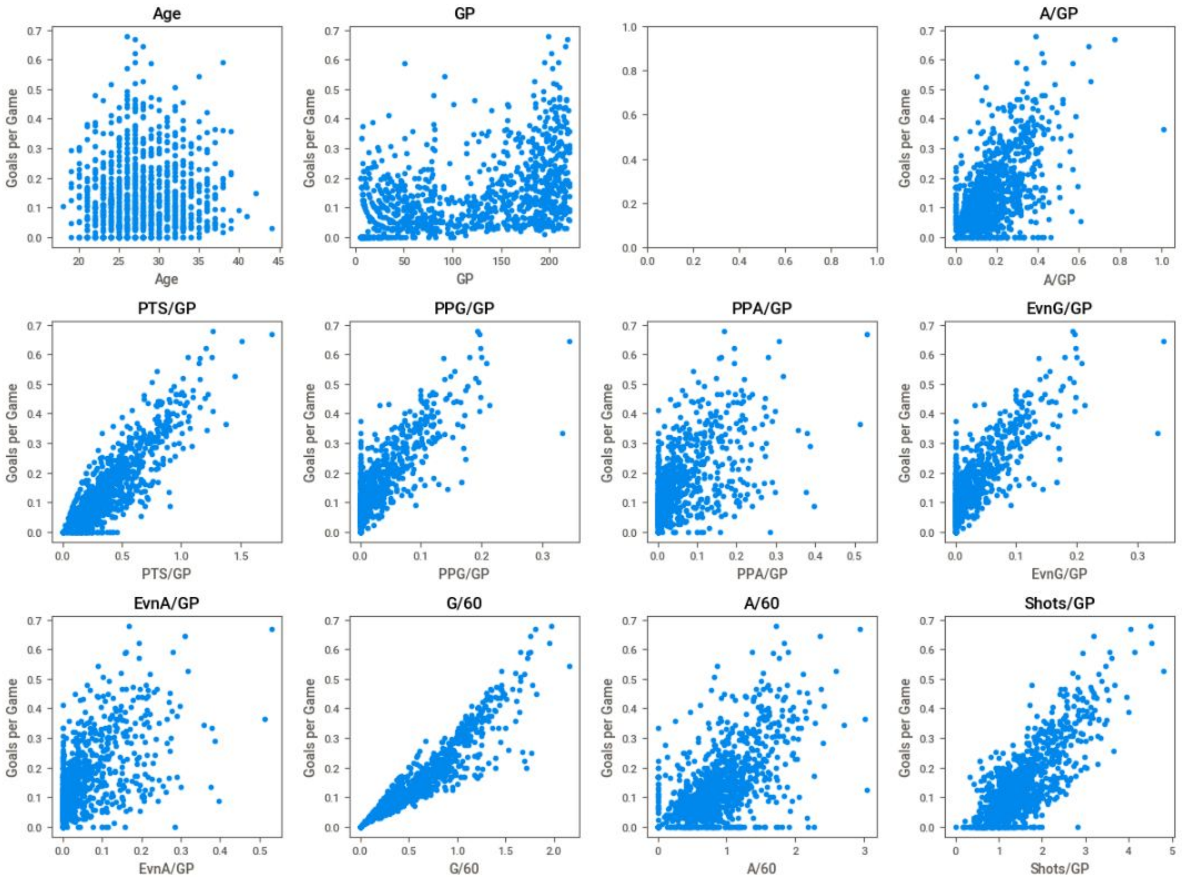
Updated Metrics Table

	Train Set Score	Test Set Score	Adjusted Rsquared	RMSE
Career Totals	0.990	0.990	0.990	7.299
Career Totals w/ manual feature reduction	0.862	0.813	0.812	32.192
Per Game w/ all features	0.998	0.998	0.998	0.004
Per Game w/ Pipeline & GridSearch	0.993	0.993	0.992	0.008
3 Year Weighted Avg w/ Pipeline & GridSearch	0.981	0.969	0.968	0.020
3 Year Weighted Avg w/ Pipeline & 0 G/GP filter	0.979	0.972	0.971	0.020
KNN model w/ StandardScaler (3 yr dataset)	1.000	0.931	0.929	0.029
Neural Network - Per Game Dataset	0.978	0.972	0.972	0.016
Neural Network - 3 Yr Weighted Dataset	0.981	0.943	0.941	0.027

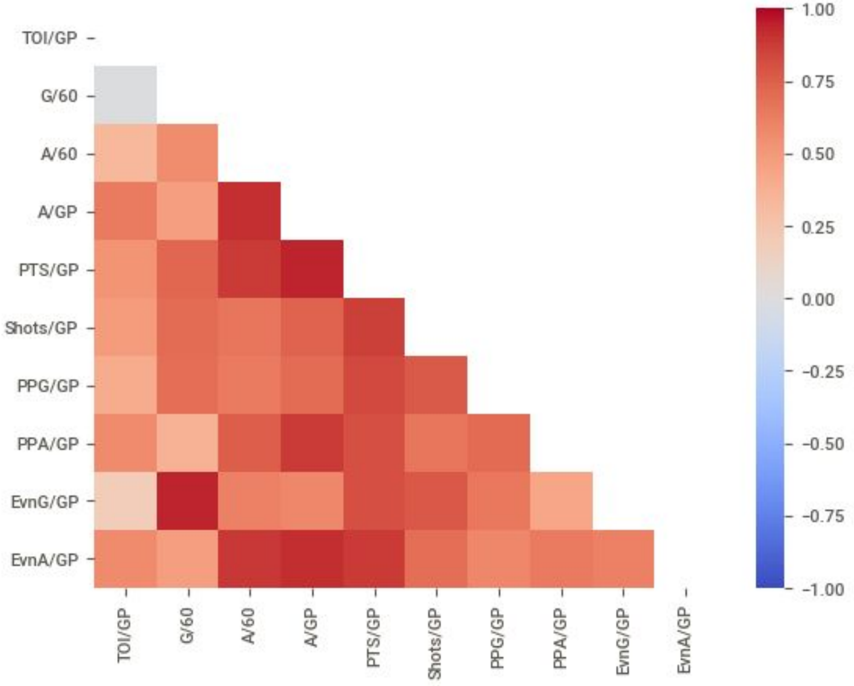
Assumptions of Linear Regression



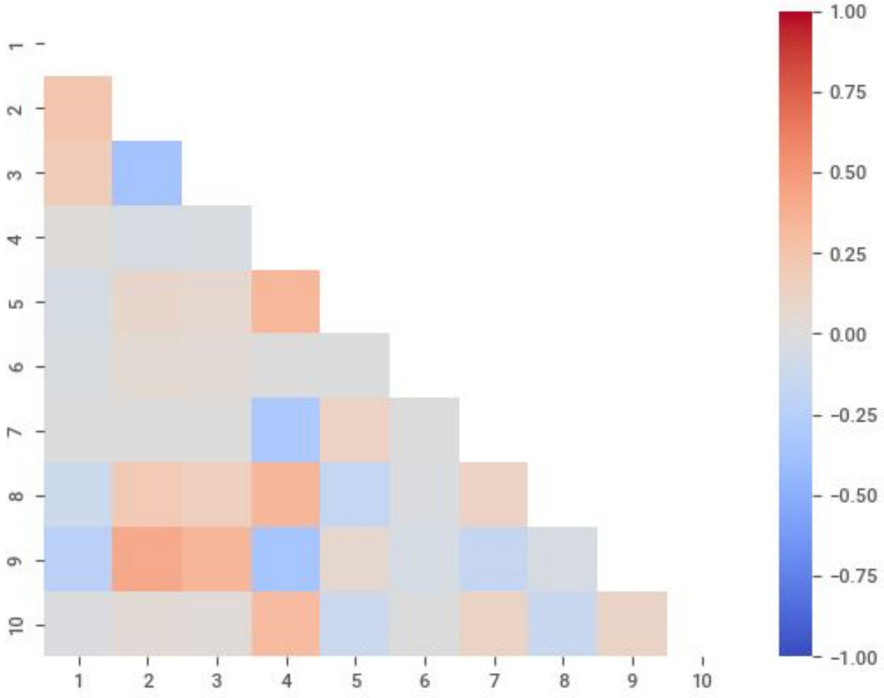
Linear Relationship Between Variables?



Multicollinearity



PCA feature reduction



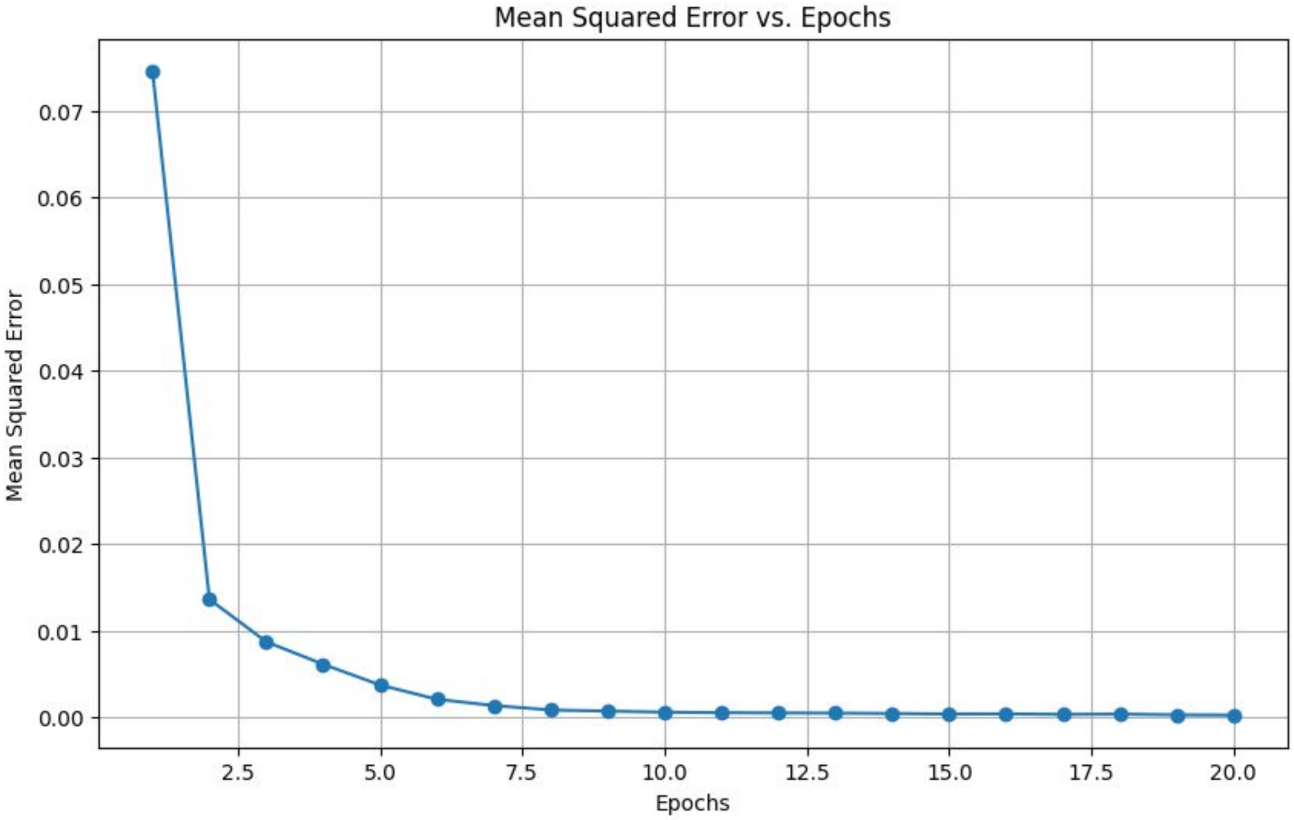
Which is the best model?

- Three Year Weighted Average fit with Linear Regression using PCA dimensionality reduction and GridSearch
- Adjusted R^2 of 0.968
- Root Mean Squared Error of 0.020
- Overall, this model seems to be the most robust
  - There is less evidence of overfitting to the training data
  - Low overall error
  - Strong performance on predictions

Next Steps…

- Repeat my entire modelling process (at least the best models) with Assists as the target
- To create a tangible, real-world result I want to train my model on data up to the 2023 season, and use the trained model to predict results for the 2024 season
  - Complications arise around curating the datasets so that the shapes match (X_train/y_train = 3_year_weighted_average & X_test/y_test = 2024 data)
- Productizing:
  - Create interactive tables
    - There are Python packages that do this…
  - Streamlit application for user input → 2024 prediction comparison

Thank You



Mean Squared Error vs. Epochs
Mean Squared Error
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs