Comparing Linear Regression and KNN with Feature Manipulation

Gwynith Godin
CS6140 Final Project
godin.g@northeastern.edu

Abstract

The purpose of this paper is to discuss general performance between Linear Regression and KNN Regression models in the context of supervised machine learning. Additionally, the number of features will be manipulated to understand the impact of model complexity on Linear Regression and KNN methods. The dataset being used is the Soccer Player Statistics from Kaggle[1], which includes descriptive statistics about current professional soccer players. The various machine learning models will predict a "player rating," which is a continuously valued label ranging 0 to 100.

1 Introduction

1.1 Dataset

The dataset comes from Kaggle and contains 53 descriptive statistics for 17341 unique players. The descriptive statistics (features) include, but are not limited to: height, weight, position, nationality, club, etc. Only field players were included in the assessment because goal keepers have their own unique features not applicable to field players. Only numeric features were chosen, those of which are valued on a scale of 0 to 100. Seven features were initially handpicked based on general player attributes that are applicable to each field position (defense, midfield, and forward). The seven initial features include "marking," "standing tackle," "aggression," "composure," "acceleration," "stamina," and "vision."

Aforementioned, each example contains an associated player rating, which will be used as the continuously valued label. The distribution of the data is unknown, so it is not confirmed if there is linearity between the features and label. In other words, we do not know for sure if the features in the dataset give direct insight into the player rating. Before constructing and implementing the models, the dataset was randomly split into a train set and a test set, consisting of 40% and 60% of the dataset, respectively.

1.2 Models and Test Methods

Linear Regression is a parametric supervised learning algorithm used for predicting continuous valued labels. This model establishes a linear relationship between features and labels and finds a best-fit line that minimizes the sum of the square errors between predicted and actual labels[2]. In this application, the model takes the general form:

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ...\theta_n x_n$$

[1]Soccer Player Statistics from Kaggle
[2]Sklearn Linear Regression

Where y is the known label, θ_n is the nth unknown parameter, and x_n is the nth feature- in this case, $n = 7$. The Linear Regression model is a handwritten model using Python's Numpy package, which works to fit X and y training data and estimates the unknown parameters by minimizing the sum of the square errors.

KNN is a non-parametric model that utilizes distances to measure similarity in points where the variable k determines the number of considered nearest neighbors[3]. A handwritten KNN function finds Euclidean distance (straight-line distance) between two points and then calculates the average of the target variables of the k neighbors. The paper will discuss training the Linear Regression and KNN models with the initial number of features $n = 7$, a small amount of features $n = 3$, and a large amount of features $n = 19$.

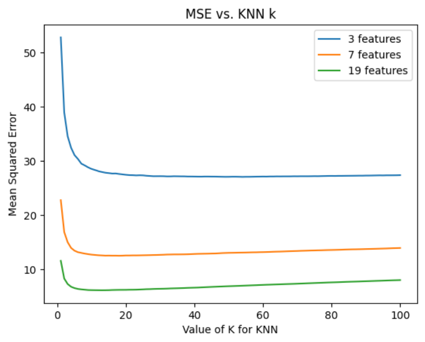
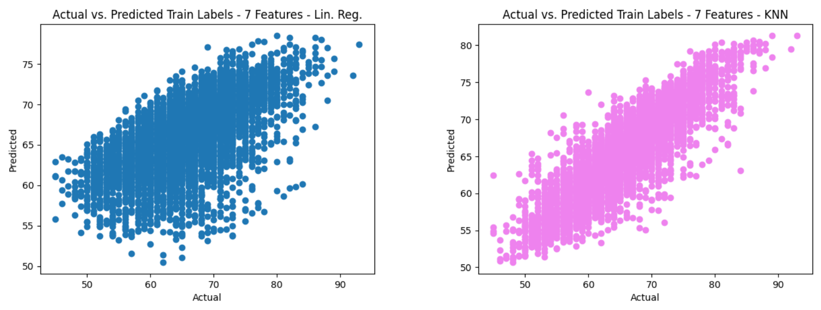


Figure 1: Finding choice of k based on MSE

The choice of k was determined by finding MSE for 100 values of k through the *sklearn* python package. In order to accommodate for such a wide-spread of features, the k value needs to generalize well enough to not cause overfitting, which can come from a k too small. To prevent underfitting or overfitting, $k = 31$ was selected, based on when the MSE stabilizes.

2 Results

Initially, a Linear Regression model and KNN model were trained and on $n = 7$ features.



(a) Linear Regression, $n = 7$ (b) KNN, $k = 31$, $n = 7$

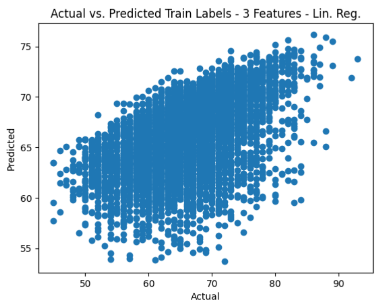
Figure 2: Linear Regression and KNN, n = 7 Features

Linear Regression and KNN models were then trained on $n = 3$ features including: "acceleration," "stamina," and "vision." They were then trained with $n = 19$ features, including all the features in $n = 7$. The Root Mean Squared Estimate $RMSE$, which measures the avg. of the squared differences between predicted and actual values and R-squared $R2$, which measures goodness of fit were found.

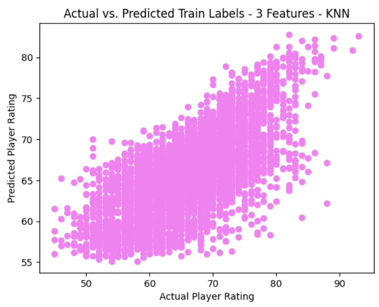
[3]How to find the optimal value of K in KNN?

Table 1: LR and KNN Values on $n = 3, 7, 19$ Features

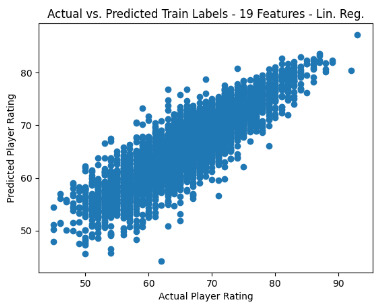
Model	Train/Test	n=7	RMSE	R^2	n=3	RMSE	R^2	n=19	RMSE	R^2
LR	Train		5.473	0.405		6.059	0.271		3.518	0.754
LR	Test		5.378	0.417		6.001	0.274		3.548	0.746
KNN	Train		2.339	0.812		2.462	0.531		1.876	0.9
KNN	Test		3.667	0.729		5.341	0.425		2.484	0.876



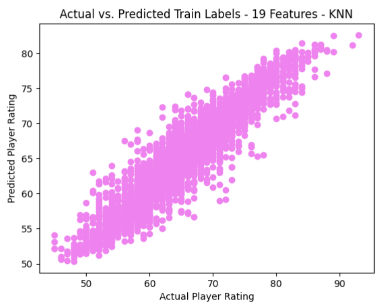
(a) Linear Regression, $n = 3$



(b) KNN, $k = 31$, $n = 3$



(c) Linear Regression, $n = 19$



(d) KNN, $k = 31$, $n = 3$

Figure 3: Linear Regression and KNN, n = 3, 19 Features

3 Discussion and Conclusion

Comparatively, KNN performed better on each model complexity. For $n = 3$, both KNN and LR train and test RMSE and R^2 are relatively low performing. With $n = 7$ features, we can see the RMSE and R^2 for both train and test shows significantly better performance. With $n = 19$, we start to see the performance level out, but KNN still yields lower RMSE and higherR^2 than LR.

It is easy to assume that the KNN model is better performing on this dataset, but the variation between test and train data indicate that KNN might be overfitting the dataset, which could be due to the k neighbors value causing flexible decision boundaries. Comparatively, the LR models show the exact opposite– the model generally performs better on test data. There is also a relationship between model complexity and performance. Seen in Figure *2a* and *2b*, the distribution of actual and predicted player ratings is more spread, indicating a higher error. Whereas *2c* and *2d* with more features show a more compact, linear distribution, indicating less error.

Overall, KNN Regression and Linear Regression are two inherently different models with the same objective. KNN excels with non-linear data and features of equal importance. Linear Regression is more suited for linear distributions and high dimensional spaces. Reducing feature correlation could ultimately be a better fit for LR. As we add features, the performance gap closes between KNN and LR, but KNN is able to capture provide better predictions for lower model complexity.

References

[1]Antione Krajnc, *https://www.kaggle.com/datasets/antoinekrajnc/soccer-players-statistics/data*, "Soccer Player Statistics".

[2] Avijeet Biswal, *https://www.simplilearn.com/tutorials/scikit-learn-tutorial/sklearn-linear-regression-with-examples*, "Sklearn Linear Regression".

[3] Rohan Kumar Bohara, *https://medium.com/@rkbohara097/how-to-find-the-optimal-value-of-k-in-knn-2d5177430f2a*, "How to find the optimal value of K in KNN?"

[4] Forestry, *https://academic.oup.com/forestry/article/94/2/311/5917597*, "An International Journal of Forest Research"