Predicting the Full-time Goal Count and Interest of Football Matches Using Linear and Quadratic Regressions, k-NN Classification and Logistical Regression

Aalto University, December 2019

ABSTRACT – given a dataset containing records of three seasons of football matches, we want to determine if it is possible to predict the full-time goal count of a match and classify it as being or not interesting. On an early stage, an analysis of the dataset is done, and relations between the variables are studied and explored. PCA analysis is then done over the dataset in order to determine how its dimension can be reduced. For the score prediction task, two regression methods, linear and quadratic, are used over the dataset, 2 PCA components and 9 PCA components. For the interest classification, k-NN and logistic regression models are used for the same datasets. The best results for the prediction task came from the linear model using the standard dataset, with RMSE = 1.48, while for the classification the logistical model using the standard dataset was the best one, with an accuracy of 67%.

1. Introduction

Football matches are the source of a lot of speculation over what will happen in the future based on a team's current track record. Being able to predict, for example, the likely number of goals that a hypothetical or ongoing match is going to have can be very valuable to many different people, from the team itself to gambling platforms. It is, then, necessary to explore existing football records in order to see if it is possible to make any predictions based on the existing data, or if there are any patterns that may be useful to the parties involved.

From this necessity, a data analysis experiment has been performed over data collected from Premier League matches from three distinct seasons. This experiment attempts to create models that can help classify a match as being or not Interesting, and to try to predict how many goals a match will have at full time based on the match's statistics at half-time.

2. Data Analysis

The data for the study has been collected from Football-Data.co.uk, and holds the entire records of the seasons from 2016 through 2018 of the Premier League, England's top football league. Each datapoint has 13 features, and there are a total of 1140 datapoints. These datapoints have been split into training and testing sets with a ratio of approximately 70%/30%, respectively (the training set has 798 datapoints and the testing set has 342). All the features are integers (either positive, negative or zero), and all datapoints have their features initialized. Therefore, there is no need to pre-process the raw data from the files.

For each datapoint, there are two labels: one, named "Interest", classifies the datapoint as being interesting or not (encoded as 1 and 0). The other label, "FTG", shows what the goal count is at full time for a given datapoint.

	HomeTeam	AwayTeam	HTHG	HTAG	HTR	HS	AS	HST	AST	HF	AF	HC	AC
0	19	8	2	1	1	18	6	7	2	8	10	5	4
1	24	16	2	0	1	12	8	6	3	12	13	5	1
2	3	12	0	0	0	13	8	4	4	8	8	0	11
3	2	20	0	0	0	7	6	2	2	10	10	2	6
4	24	9	0	1	-1	14	11	6	3	13	16	4	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...
793	25	20	0	0	0	10	16	3	6	11	14	4	7
794	6	16	1	0	1	16	15	5	4	6	12	5	4
795	9	2	0	2	-1	6	15	1	7	9	13	2	3
796	0	7	1	0	1	15	11	6	8	5	17	5	4
797	12	18	1	1	0	23	6	10	4	7	13	9	2

798 rows × 13 columns

Figure 1: Visualization of the features of the training dataset

2.1. Measuring Correlation

In order to determine how the multiple features are related to each other, a correlation matrix was calculated for both the training and testing datasets. A correlation matrix is a diagonal matrix that shows, for each pair of features, how related they are to each other. The correlation value varies from -1 to 1, where 0 stands for two features that are completely uncorrelated, and -1 or 1 for two completely identical features (the most common example of the latter is when we calculate the correlation of a feature with itself, which can be seen on the main diagonal of the matrix).

	HomeTeam	AwayTeam	HTHG	HTAG	HTR	HS	AS	HST	AST	HF	AF	HC	AC
HomeTeam	1	-0.047	0.00082	-0.041	0.024	0.06	0.026	0.026	0.011	0.015	0.056	0.039	0.037
AwayTeam	-0.047	1	-0.027	-0.014	0.0027	-0.028	0.03	-0.024	0.016	0.0049	-0.013	0.022	0.072
HTHG	0.00082	-0.027	1	-0.072	0.67	0.16	-0.048	0.41	-0.081	-0.067	0.049	-0.041	0.018
HTAG	-0.041	-0.014	-0.072	1	-0.66	-0.12	0.19	-0.12	0.4	0.066	-0.00024	-0.1	-0.064
HTR	0.024	0.0027	0.67	-0.66	1	0.17	-0.16	0.33	-0.31	-0.086	0.03	0.039	0.055
HS	0.06	-0.028	0.16	-0.12	0.17	1	-0.49	0.71	-0.32	-0.12	0.016	0.58	-0.4
AS	0.026	0.03	-0.048	0.19	-0.16	-0.49	1	-0.34	0.68	0.088	-0.052	-0.38	0.55
HST	0.026	-0.024	0.41	-0.12	0.33	0.71	-0.34	1	-0.24	-0.04	0.0045	0.37	-0.26
AST	0.011	0.016	-0.081	0.4	-0.31	-0.32	0.68	-0.24	1	0.099	-0.059	-0.24	0.29
HF	0.015	0.0049	-0.067	0.066	-0.086	-0.12	0.088	-0.04	0.099	1	0.11	-0.074	0.0019
AF	0.056	-0.013	0.049	-0.00024	0.03	0.016	-0.052	0.0045	-0.059	0.11	1	0.023	-0.05
HC	0.039	0.022	-0.041	-0.1	0.039	0.58	-0.38	0.37	-0.24	-0.074	0.023	1	-0.34
AC	0.037	0.072	0.018	-0.064	0.055	-0.4	0.55	-0.26	0.29	0.0019	-0.05	-0.34	1

Figure 2: Correlation Matrix for the training dataset

	HomeTeam	AwayTeam	HTHG	HTAG	HTR	HS	AS	HST	AST	HF	AF	HC	AC
HomeTeam	1	-0.06	-0.057	0.038	-0.044	-0.016	0.042	0.0057	0.046	-0.058	-0.047	-0.047	-0.0027
AwayTeam	-0.06	1	-0.0092	0.055	-0.028	0.014	0.1	-0.026	0.0066	0.0059	0.054	0.062	0.11
HTHG	-0.057	-0.0092	1	-0.072	0.71	0.14	-0.12	0.34	-0.16	-0.077	0.017	-0.043	0.074
HTAG	0.038	0.055	-0.072	1	-0.62	-0.061	0.17	-0.094	0.39	0.088	0.067	-0.05	0.04
HTR	-0.044	-0.028	0.71	-0.62	1	0.13	-0.15	0.32	-0.31	-0.099	-0.051	0.0099	0.0074
HS	-0.016	0.014	0.14	-0.061	0.13	1	-0.49	0.7	-0.33	-0.11	0.033	0.58	-0.41
AS	0.042	0.1	-0.12	0.17	-0.15	-0.49	1	-0.38	0.66	0.11	-0.091	-0.41	0.53
HST	0.0057	-0.026	0.34	-0.094	0.32	0.7	-0.38	1	-0.24	-0.11	-0.029	0.37	-0.27
AST	0.046	0.0066	-0.16	0.39	-0.31	-0.33	0.66	-0.24	1	0.11	-0.052	-0.23	0.3
HF	-0.058	0.0059	-0.077	0.088	-0.099	-0.11	0.11	-0.11	0.11	1	0.093	-0.14	0.029
AF	-0.047	0.054	0.017	0.067	-0.051	0.033	-0.091	-0.029	-0.052	0.093	1	0.086	-0.074
HC	-0.047	0.062	-0.043	-0.05	0.0099	0.58	-0.41	0.37	-0.23	-0.14	0.086	1	-0.35
AC	-0.0027	0.11	0.074	0.04	0.0074	-0.41	0.53	-0.27	0.3	0.029	-0.074	-0.35	1

Figure 3: Correlation Matrix for the testing dataset

Before we start to examine the feature pairs, it is useful to first determine if the two datasets are statistically similar. If this is true, then it will simplify the analysis, as we will only need to do it on the largest dataset (i.e. the training one) instead of doing it on both.

Given that the correlation matrices of both datasets are very similar to each other, we will assume that the datasets are statistically similar, and will, from now on, use the training dataset only for the rest of this preliminary analysis.

2.2. Pairwise Plots

From the correlation matrix we can distinguish some feature pairs with high correlation values. We will choose 5 feature pairs with high correlation and plot the pair-wise plots for each of them in order to determine if there is any tangible reason for the high correlation, and to attempt to offer an explanation for that reason.

2.2.1. HTR and HTHG

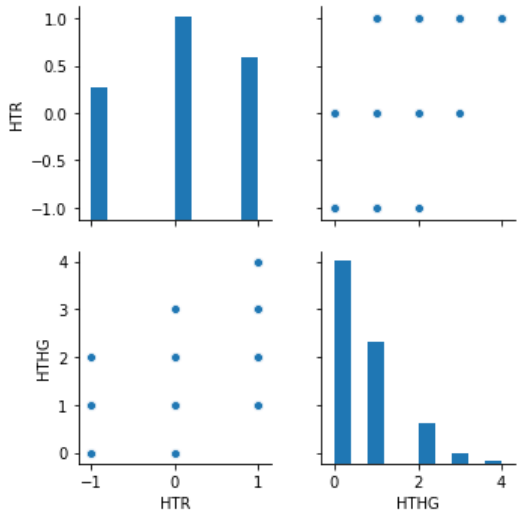

Figure 4: Pairwise plots for the half-time results and the half-time results of the home team

These plots show the relation between the number of goals of the home team at half-time (HTHG) and whether it is losing (-1), drawing (0) or winning the game (1) at half-time (HTR). We can see that, if the team is winning at half-time, then it most likely has a lot of goals by then, and vice-versa: a small number of goals at half-time may mean it is drawing or losing.

2.2.2. HTR and HTAG

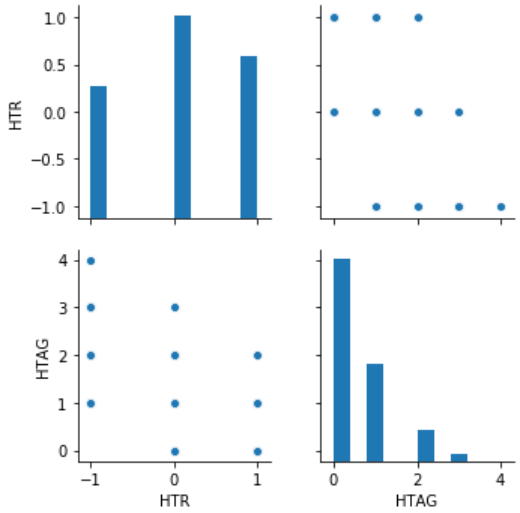

Figure Pairwise plots for the half-time results and the half-time results of the away team

As we can see, the same trend as the previous example applies, too, to the away team (in this case, -1 and 1 switch, as -1 is a victory for the away team).

2.2.3. HST and HS

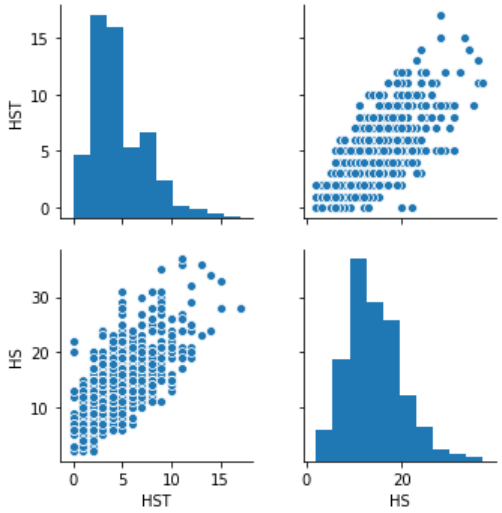

Figure 5: pairwise plots for the number of shots and the number of shots on target by the home team

Here we can see how the number of shots made by the home team (HS) relates to the number of shots on target by the home team (HST). It follows that every shot on target is a shot, but not all shots are on target. We can see that, generally, the more shots a team has, the more shots on target it has too.

2.2.4. HC and HS

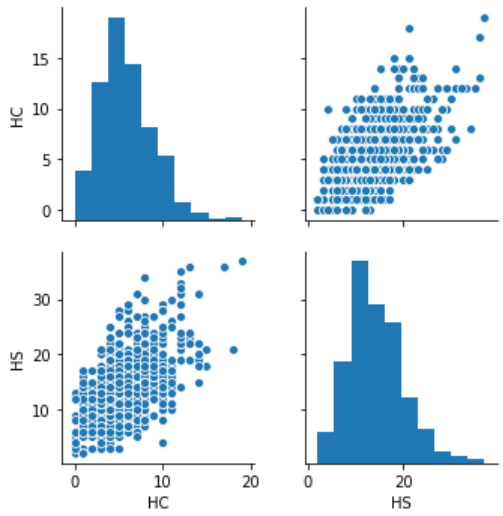

Figure 6: Pairwise plot for number of corners and number of shots by the home team

Here we can see the relation between shots (HS) and corners (HC) made by the home team. Generally, a higher number of one means a higher number of the other, and vice-versa.

2.2.5. AC and AS

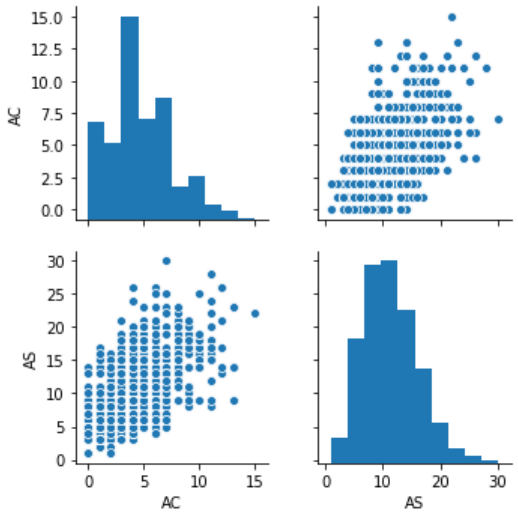

Figure 7: Pairwise plot for number of corners and number of shots by the away team

Similarly to the previous plot, the same trend regarding corners (AC) and shots (AS) is valid for the away team too, although the relation is not as strong – bigger number of shots is possible with a lower number of corners when compared to the previous plot.

2.3. Principal Component Analysis

Having 13 total features, the dataset is subject to the curse of dimensionality. Principal Component Analysis (PCA) is a transformation of the dataset that allows for it to be expressed using a smaller number of features, while still retaining most of its original variance. These artificial features, called components, are linear combinations of the original features, and are always orthogonal to each other in order to maximize the variance in-between them. The number of principal components can be chosen, and it follows that the more principal components we use, the closer to the original variance we will get.

In order to calculate a PCA transform, the data needed to be standardized first so that each feature is expressed as having mean = 0 and variance = 1.

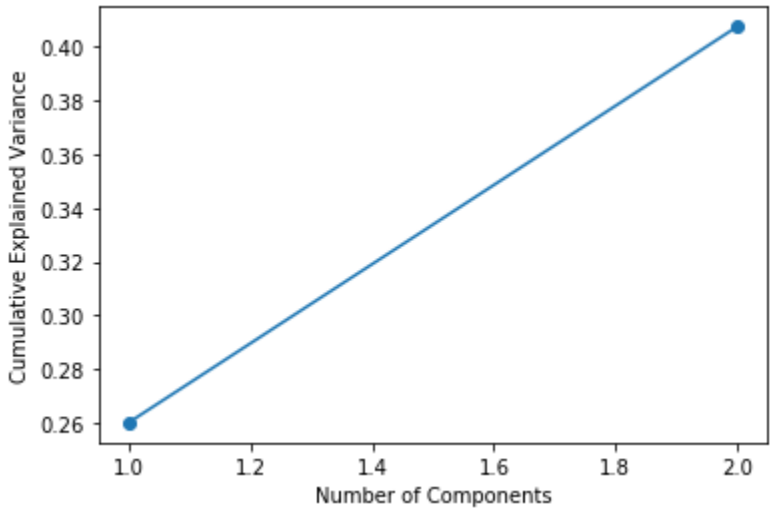

Figure 8: Cumulative Explained Variance for the first 2 PCA components

We proceeded to calculate the PCA transform for the training dataset, and plotted the cumulative explained variance of the first two components on a plot. As it can be seen, by only using 2 components we can only express around 40.7% of the original variance.

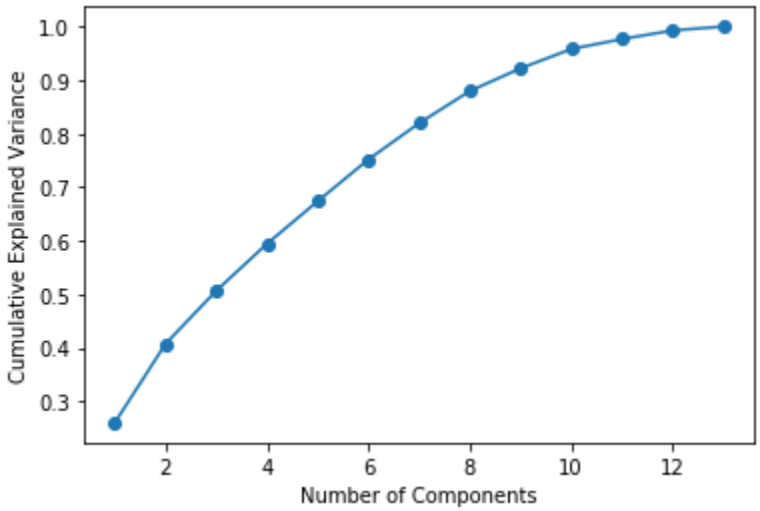

Figure 9: Cumulative Explained Variance for all PCA components

If we plot the cumulative explained variance for all 13 PCA components, we can see that it takes many components before it starts nearing 100%. For a cumulative explained variance value superior to 90%, we'd have to use at least 9 PCA components.

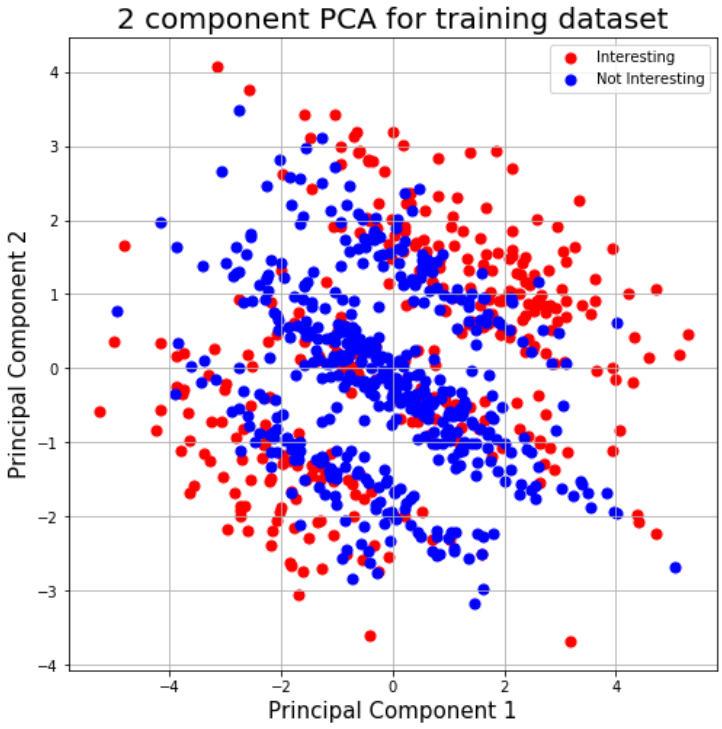

Figure 10: 2 component PCA for training dataset and the "Interest" label

By using 2 components, we are able to map, in 2D, the datapoints and their respective labels. Since the "FTG" label is continuous, we can only map a plot for the "Interest" label.

3. Methods

This section describes the methods applied in order to solve the two proposed tasks, including the mathematical foundations of the models to be used.

3.1. Predicting the full-time score

The prediction of the full-time goals (sum of the number of goals of both teams at the end of the match) must be approached as a regression problem, since FTG is a continuous value in $\mathbb{N}^0$.

The high dimensionality of the data, as previously explained, can pose a problem, and as such we will build 3 different models: one using the original dataset, another using 2 PCA components and finally another 9 PCA components (as that was the number of components previously identified that assured at least 90% of cumulative variance).

Some of the features that may be of particular relevance to predict the full-time score are the HTHG and HTAG (half-time goals of the home and away teams) and the HST and AST (shots on target by the home and away teams). We plan to evaluate the model later on to see if this hypothesis holds.

Since we cannot, at first glance, decide if there is a linear trend to the data, we will adopt two regression models: one Linear, and the other Quadratic (second degree polynomial).

Thus, we shall have a total of 6 models: 3 Linear and 3 Quadratic. All will be validated using the testing datasets, and evaluated using the Root Mean Squared Error.

3.1.1. Multiple Linear Regression

A linear regression with multiple independent variables, as is the case with our data, attempts to build a linear function that outputs the target variable y by assigning weights β to each feature x of the datapoint:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_k x_k$$

The weight vector β can be estimated through the Ordinary Least Square Method, or some of its other variants. The generic formula for this method is the following, where X is the matrix of datapoints and y the target vector:

$$\beta = (X^T X)^{-1} X^T y$$

3.1.2. Quadratic Regression

A quadratic regression can be considered to be a special case of linear regression, as we can transform the input datapoints to then be applied on a linear regression model. This was the approach chosen for this study. The transformation creates polynomial combinations of features with degree less or equal than 2. For example, the vector $[a, b]$ would be transformed onto $[1, a, b, a^2, ab, b^2]$. These transformed datapoints are then fed to a multiple linear regression model (e.g. the same as used for the linear model).

3.1.3. Root Mean Square Error (RMSE)

The RMSE metric gives an idea of, on average, how far from the predicted value a datapoint is expected to be. The lower this value is, the more perfect is the model. It is calculated by taking the square root of the Mean Square Error (MSE), which is given by:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

3.2. Classifying a match as "interesting"

The classification of a match as being "interesting" is a binary classification problem. As with the previous task, we will also adopt 3 datasets (original, PCA with 2 components and PCA with 9 components), and two distinct classification models.

The first chosen model is k-Nearest Neighbors (k-NN). This model is adequate for binary classification for odd k values, and the value to be chosen is 5. From figure 6 we can see that there is significant overlap between the two kinds of datapoints, and as such a higher k value can help reduce the risk of misclassification due to the influence of neighbors with opposite labels.

The second chosen model is a Logistic Regression, which is used for binary classification by predicting a probability to the label.

Finally, the confusion matrices for each model will be calculated using the testing datasets, as well as each model's accuracy.

3.2.1. k-NN

The k-NN model, unlike most models, does not have a training phase. All it does is keep a record of all the training datapoints. Then, for each new datapoint x we wish to classify, we calculate the k datapoints that are nearest to x. This measurement is done using, for example, the Euclidean distance between the points. After the k closest datapoints have been determined, we assign to x the same label as the majority of its neighbors. In the case of a tie between labels, it is chosen randomly between those labels. Therefore, in the case of binary classification, it is important to specify an odd value for k, as that avoids ties between binary categories.

3.2.2. Logistic Regression

Given the Sigmoid function $g(z) = \frac{1}{1+e^{-z}}$ and the value of z, a probability between 0 and 1 shall be produced. The Logistic Regression model tries, then, to create a function that outputs a value for z based on the training dataset, to then be fed to the Sigmoid function. This function, $z = f(X)$, is usually a

linear combination of the different features, and can be estimated using Maximum Likelihood Estimation and the training dataset. To predict the label of a datapoint x, we first calculate the z value of x and then feed it to the Sigmoid function, which shall output the probability p of the datapoint being classified with the positive label (and, naturally, the probability $1 - p$ of it being classified with the negative label). In other words, if $p > 0.5$ we classify it as having the positive label, and if $p < 0.5$, we classify it with the negative one.

3.2.3. Confusion Matrix and accuracy

A confusion matrix is a $2x2$ matrix that allows for the comparison of the actual and predicted values outputted by a binary classifier when tested using the testing dataset, and can be used to determine the quality of a model. The sum of all 4 elements of the matrix is the total number of datapoints on the testing dataset, and it matches the predicted positives and negatives with the actual positives and negatives.

The accuracy metric can be expressed as a percentage and is calculated by the formula, where TP = true positives, TN = true negatives, FP = false positives and FN = false negatives:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

4. Experiments and Results

This section describes the results of the experiments performed using the previously described methods. The software used for the experiments was the scikit-learn Python library.

4.1. Predicting the full-time score

Here are the Root Mean Square Error (RMSE) results for each model:

	Default Dataset	2 PCA Components	9 PCA Components
Linear Regression	1.48	3.16	2.11
Quadratic Regression	1.51	3.83	1.48

We also include the obtained Linear Regression model for the Default Dataset:

$$FTG = 0.01 * HomeTeam - 0.00 * AwayTeam + 0.87 * HTHG + 0.74 * HTAG - 0.09 * HTR + -0.01 * HS - 0.01 * AS + 0.17 * HST + 0.18 * AST + 0.00 * HF - 0.03 * AF - 0.04 * HC - 0.01 * AC$$

4.2. Classifying a match as "interesting"

Here are the 6 confusion matrices for the created models, as well as the accuracy measurements. "AP" and "AN" stand for "Actual Positives" and "Actual Negatives", respectively, and "PP" and "PN" stand for "Predicted Positives" and "Predicted Negatives".

	PP	PN
AP	60	88
AN	48	146

Figure 11: k-NN + default dataset

	PP	PN
AP	44	104
AN	71	123

Figure 12: k-NN + 2 PCA components

	PP	PN
AP	31	117
AN	64	130

Figure 13: k-NN + 9 PCA components

	PP	PN
AP	82	66
AN	47	147

Figure 14: Logistic Regression + default dataset

	PP	PN
AP	21	127
AN	37	157

Figure 15: Logistic Regression + 2 PCA components

	PP	PN
AP	55	93
AN	62	132

Figure 16: Logistic Regression + 9 PCA components

Model	Accuracy (%)
k-NN + default dataset	60
k-NN + 2 PCA components	49
k-NN + 9 PCA components	47
Logistic Regression + default dataset	67
Logistic Regression + 2 PCA components	52
Logistic Regression + 9 PCA components	54

Figure 17: Accuracy of each model

5. Conclusion and discussion

5.1. Predicting the full-time score

One of the two models with the lowest RMSE value is the one that uses the default dataset and a linear regression. We can notice that the 4 features we originally hypothesized as being the most important for the prediction (HTHG, HTAG, HST and AST) are the ones with the biggest weight coefficient on the model, and therefore we can confirm that they are the ones that contribute the most to the final result.

Regarding the PCA versions, on the linear model the 2 PCA component dataset was the worst, with the 9 PCA version in the middle. The fitness of the linear model, then, directly correlates with the variance of the data, with lower variance giving a worse fitness.

On the quadratic version, we can see that the results for the default dataset and the 2 PCA dataset were worse than the linear, but for the 9 PCA dataset we got the same RMSE value as the linear model trained with the default dataset. We can't take any conclusions as to which features are being the most relevant to the final result in this case due to both the quadratic combinations and PCA transformations undergone by the dataset, and as such a direct comparison between this model and the linear one with the default dataset is limited. However, if we were to choose one, the linear with the default dataset would be better, as it does not require the input to be transformed.

5.2. Classifying a match as "interesting"

As for the classification task, some observations can be made. When it comes to using the default dataset versus the PCA ones, on both the k-NN and the Logistic Regression the default dataset had a better accuracy. This can be attributed to the reduced variance of the PCA datasets, which causes information to be lost.

Comparing the 2 PCA components dataset to the 9 PCA components, we can also see that there is little difference between both of them. On the k-NN models, the 2 PCA version was 2% more accurate, while on the logistical regression model the opposite happened, with the 9 PCA version being 2% more accurate. Furthermore, 47% and 49% accuracy are worse than what we would expect from the accuracy of a random guess of a binary label (50%), therefore making these two models (k-NN with PCA 2 and PCA 9) useless.

Finally, comparing the k-NN models against the logistic regression ones, regardless of the dataset used, the logistic regression models had better accuracy. This can partly be explained by the significant overlap of differently-labeled datapoints on Figure 6 (e.g. a positive datapoint may have mostly negative datapoints as its closest neighbors, and vice-versa), as that will inevitably cause misclassifications. Further work would be required in order to determine if increasing the number of neighbors k could improve the accuracy of the k-NN classifier when compared to the logistic one. Additionally, other classification models could be applied to the data, particularly ones that are more suited to overlapping and high-dimensional data.

6. References

[1] https://en.wikipedia.org/wiki/Mean_squared_error

[2] Preprocessing data for a quadratic transformation on sklearn

[3] http://football-data.co.uk/

[4] https://en.wikipedia.org/wiki/Polynomial_regression

[5] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[6] https://en.wikipedia.org/wiki/Logistic_regression

[7] https://en.wikipedia.org/wiki/Accuracy_and_precision

[8] https://en.wikipedia.org/wiki/Ordinary_least_squares