Practical Business Analytics

UFC Betting Prediction and Pay-Per-View Analysis

Group Project Report

Group Name: The Deepest Learners

Module Lecturer: Prof Nick Ryman-Tubb

Table of contents

Project Scope

The high processing power of computers, and large volumes of data availability nowadays, enables humans to make data-driven decisions.
Data-driven decision making (DDDM) means producing decisions that are supported by enough data rather than decisions that are intuitive or based on perception alone.
This sparked the idea of using past sports data, specifically all Ultimate Fighting Championship (UFC) events, to discover recurring patterns to predict the outcome of a sporting event in the future.

Introduction to UFC

The Ultimate Fighting Championship (UFC) is an American based company which is the largest promoter of mixed martial arts events (Wikipedia, 2019).
The UFC is currently one of the fastest-growing sports in the world (Telegraph, 2017) and organises events weekly; Events consist of a few multiple-rounded fights. There are more than 40 events every year with the main event taking place once every month. The generated bar chart below illustrates the increase in the number of UFC matches over the years.

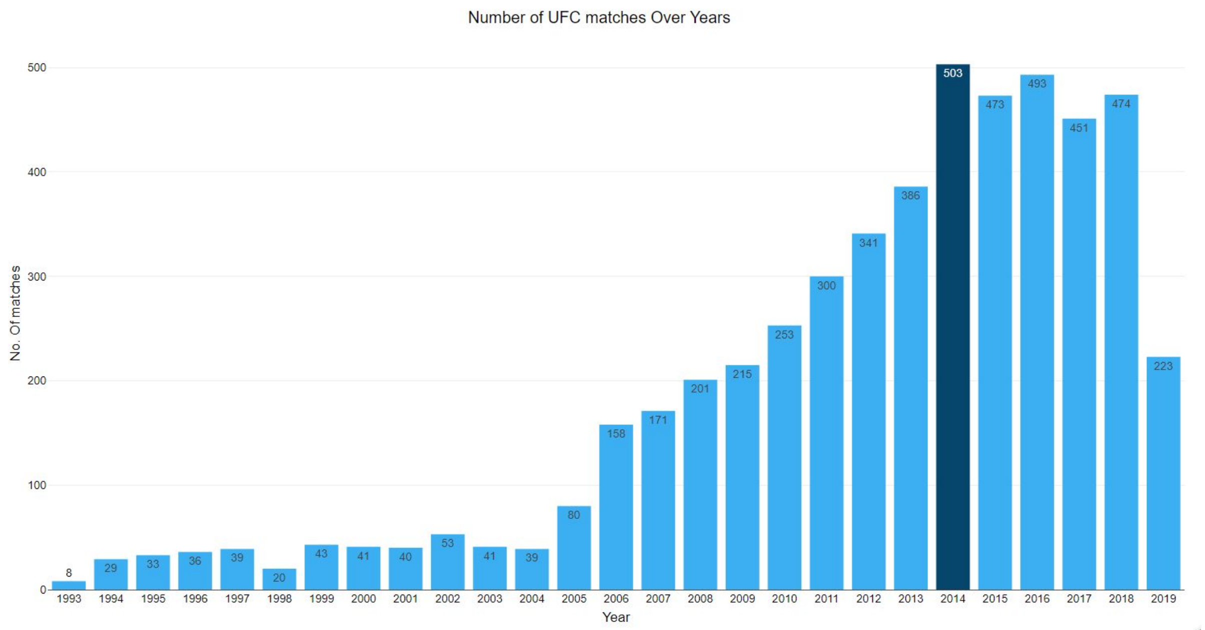


Figure 1) Number of events held yearly, since 1993 with 2014 holding maximum number of events so far .

Initial Hypothesis

The initial hypothesis was that the rank of an athlete and their win-loss ratio can be the most imperative fields in winning probability determination. Hence, the objective is to determine the most correlated factors with winning probability to challenge the initial hypothesis.

Business Case Overview

Similar to the stock exchanges, Betting exchanges are a marketplace for clients to gamble on the result of events; Customers can "back" or "lay" the outcome, and they can trade in real-time during the event to stop the loss or take a profit. Betting exchanges may have lucrative potentials if one can have an advantage over other players/gamblers by having a robust prediction model

Next, The Pay-Per-View (PPV) is defined as a broadcast method where customers purchase the right of viewing that specific event on their private telecast. At first glance, there seems to be a relationship between the number of PPV sales and specification of an event (e.g: Location, Weight Class). Analysis of this relationship highlights the business prospects clustering the UFC universe.

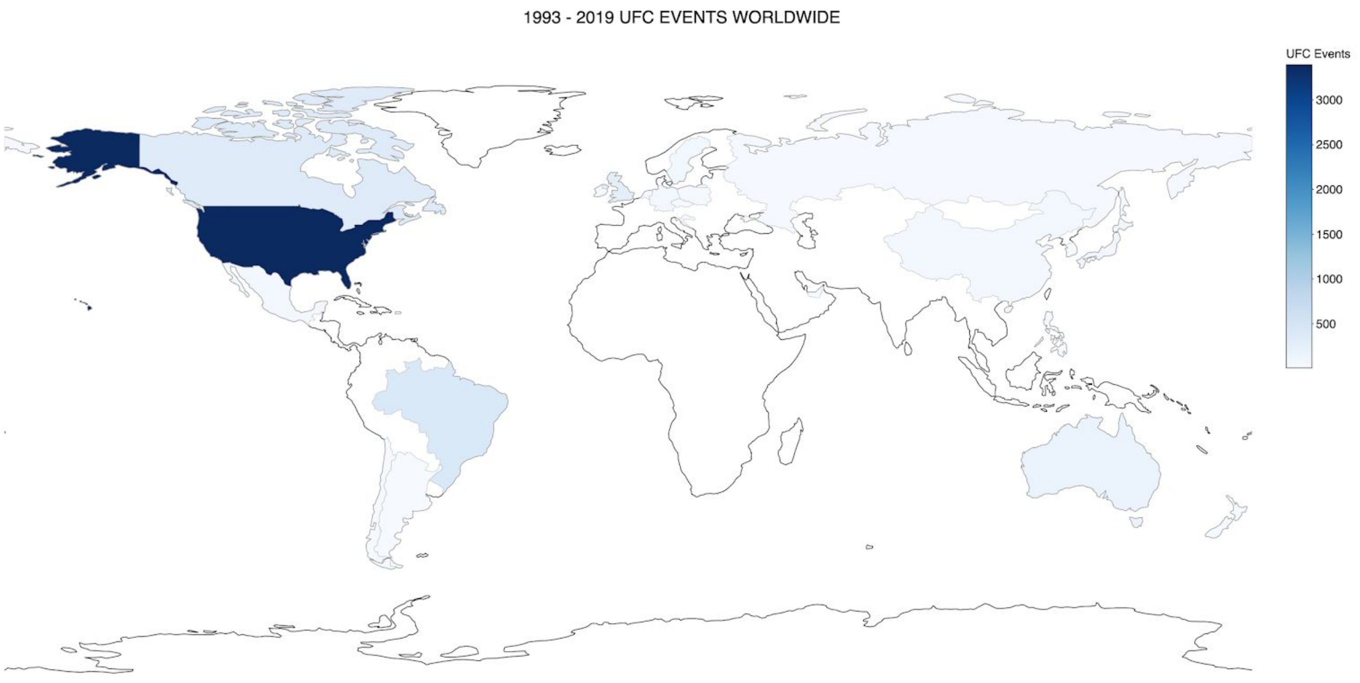


Figure 2) 1993 - 2019 UFC Events Worldwide[1]

The produced choropleth map displays the density of events held worldwide with the United States, Brazil and Canada ranking 1 to 3

[1] Interactive figure available in R scripts.

Data Understanding: UFC-Fight historical data from 1993 to 2019

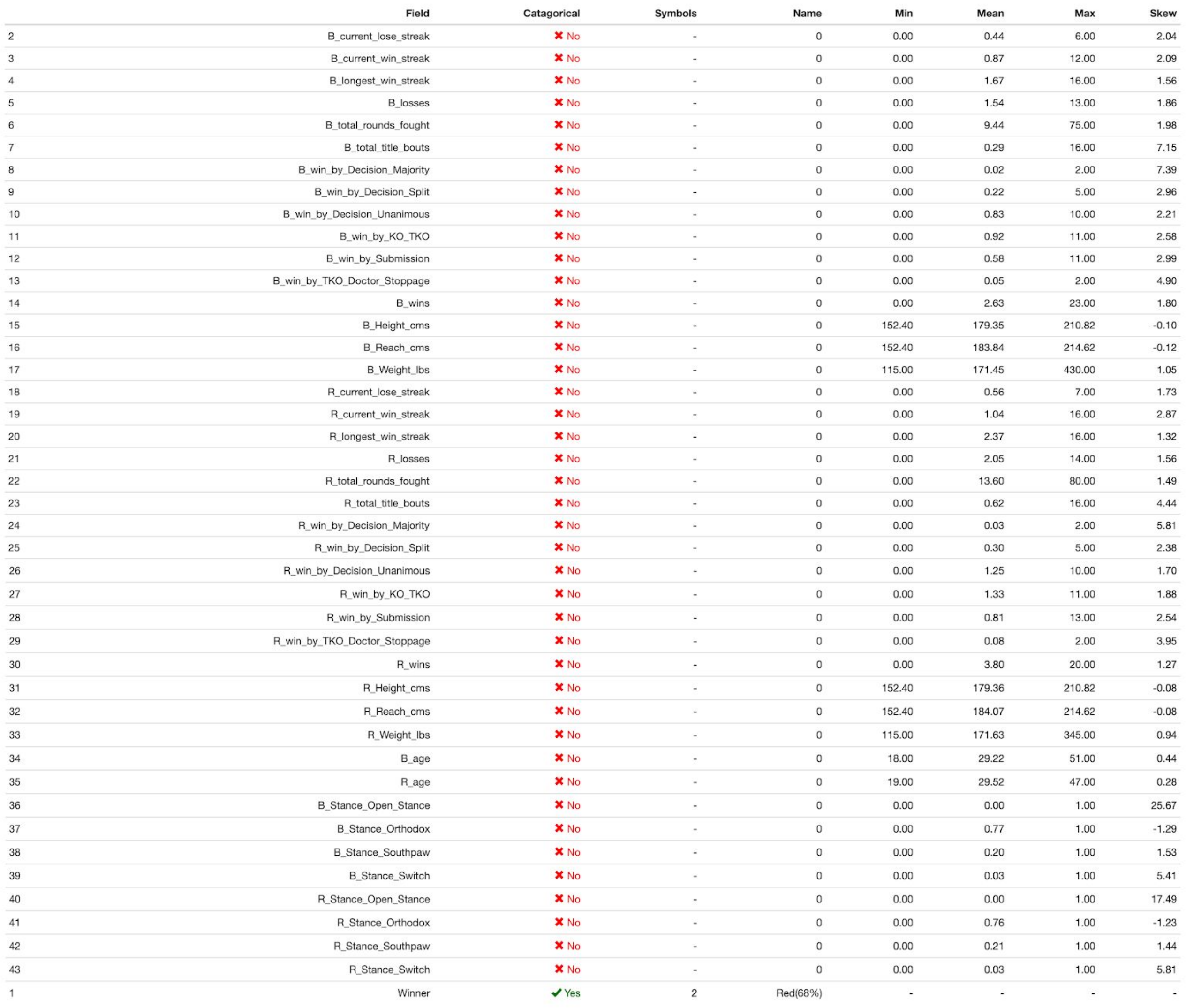
The original dataset, `data.csv`, contains the list of all UFC fights in the history of the organisation. Each row represents information on match details, two fighters (blue and red), and the winner.
The aforementioned dataset has a dimension of 5144 rows by 145 columns and includes Factors and Numeric data types.

The data dictionary below can be used to describe the fields:

Column Name	Description
R_ and B_ prefix	signifies red and blue corner fighter stats respectively
opp	containing columns is the average damage done by the opponent on the fighter
KD	is the number of knockdowns
SIG_STR	Number of significant strikes 'landed of attempted'
SIG_STR_pct	is significant strikes percentage
TOTAL_STR	is total strikes 'landed of attempted'
TD	is Number of takedowns
TD_pct	is takedown percentages
SUB_ATT	Number of submission attempts
PASS	Number of times the guard was passed
HEAD	Number of significant strikes to the head 'landed of attempted'
BODY	Number of significant strikes to the body 'landed of attempted'
CLINCH	Number of significant strikes in the clinch 'landed of attempted'
GROUND	Number of significant strikes on the ground 'landed of attempted'
win_by	is a method of win
last_round	is the last round of the fight (ex. if it was a KO in 1st; then this will be 1)
last_round_time	is when the fight ended in the last round
Format	is the format of the fight (3 rounds/ 5 rounds etc.)
Referee	is the name of the Ref
Date	is the date of the fight
Location	is the location in which the event took place
Fight_type	is which weight class and whether it is a title bout or not
Winner	is the winner of the fight
Stance	is the stance of the fighter (orthodox/ southpaw/ etc.)
Height_cms	is the height in centimetres
Reach_cms	is the reach of the fighter (arm span) in centimetres.
Weight_lbs	is the weight of the fighter in pounds (lbs)
age	is the age of the fighter
title_bout	A boolean value of whether it is a title fight or not
weight_class	is which weight class the fight is in (Bantamweight/ heavyweight/ Women's flyweight/ etc.)
no_of_rounds	is the number of rounds the fight was scheduled for

current_lose_streak	is the count of current concurrent losses of the fighter
current_win_streak	is the count of current concurrent wins of the fighter
draw	is the number of draws in the fighter's UFC career
wins	is the number of wins in the fighter's UFC career
losses	is the number of losses in the fighter's UFC career
total_rounds_fought	is the average of total rounds fought by the fighter
total_time_fought(seconds)	is the count of total time spent fighting in seconds
total_title_bouts	is the total number of title bouts taken part in by the fighter
win_by_Decision_Majority	is the number of wins by majority judges decision in the fighter's UFC career
win_by_Decision_Split	is the number of wins by split judges decision in the fighter's UFC career
win_by_Decision_Unanimous	is the number of wins by unanimous judges decision in the fighter's UFC career
win_by_KO/TKO	is the number of wins by knockout in the fighter's UFC career
win_by_Submission	is the number of wins by submission in the fighter's UFC career
win_by_TKO_Doctor_Stoppage	is the number of wins by doctor stoppage in the fighter's UFC career

Table 1) Data dictionary.



	Field	Catagorical	Symbols	Name	Min	Mean	Max	Skew
2	B_current_lose_streak	✗ No	-	0	0.00	0.44	6.00	2.04
3	B_current_win_streak	✗ No	-	0	0.00	0.87	12.00	2.09
4	B_longest_win_streak	✗ No	-	0	0.00	1.67	16.00	1.56
5	B_losses	✗ No	-	0	0.00	1.54	13.00	1.86
6	B_total_rounds_fought	✗ No	-	0	0.00	9.44	75.00	1.98
7	B_total_title_bouts	✗ No	-	0	0.00	0.29	16.00	7.15
8	B_win_by_Decision_Majority	✗ No	-	0	0.00	0.02	2.00	7.39
9	B_win_by_Decision_Split	✗ No	-	0	0.00	0.22	5.00	2.96
10	B_win_by_Decision_Unanimous	✗ No	-	0	0.00	0.83	10.00	2.21
11	B_win_by_KO_TKO	✗ No	-	0	0.00	0.92	11.00	2.58
12	B_win_by_Submission	✗ No	-	0	0.00	0.58	11.00	2.99
13	B_win_by_TKO_Doctor_Stoppage	✗ No	-	0	0.00	0.05	2.00	4.90
14	B_wins	✗ No	-	0	0.00	2.63	23.00	1.80
15	B_Height_cms	✗ No	-	0	152.40	179.35	210.82	-0.10
16	B_Reach_cms	✗ No	-	0	152.40	183.84	214.62	-0.12
17	B_Weight_lbs	✗ No	-	0	115.00	171.45	430.00	1.05
18	R_current_lose_streak	✗ No	-	0	0.00	0.56	7.00	1.73
19	R_current_win_streak	✗ No	-	0	0.00	1.04	16.00	2.87
20	R_longest_win_streak	✗ No	-	0	0.00	2.37	16.00	1.32
21	R_losses	✗ No	-	0	0.00	2.05	14.00	1.56
22	R_total_rounds_fought	✗ No	-	0	0.00	13.60	80.00	1.49
23	R_total_title_bouts	✗ No	-	0	0.00	0.62	16.00	4.44
24	R_win_by_Decision_Majority	✗ No	-	0	0.00	0.03	2.00	5.81
25	R_win_by_Decision_Split	✗ No	-	0	0.00	0.30	5.00	2.38
26	R_win_by_Decision_Unanimous	✗ No	-	0	0.00	1.25	10.00	1.70
27	R_win_by_KO_TKO	✗ No	-	0	0.00	1.33	11.00	1.88
28	R_win_by_Submission	✗ No	-	0	0.00	0.81	13.00	2.54
29	R_win_by_TKO_Doctor_Stoppage	✗ No	-	0	0.00	0.08	2.00	3.95
30	R_wins	✗ No	-	0	0.00	3.80	20.00	1.27
31	R_Height_cms	✗ No	-	0	152.40	179.36	210.82	-0.08
32	R_Reach_cms	✗ No	-	0	152.40	184.07	214.62	-0.08
33	R_Weight_lbs	✗ No	-	0	115.00	171.63	345.00	0.94
34	B_age	✗ No	-	0	18.00	29.22	51.00	0.44
35	R_age	✗ No	-	0	19.00	29.52	47.00	0.28
36	B_Stance_Open_Stance	✗ No	-	0	0.00	0.00	1.00	25.67
37	B_Stance_Orthodox	✗ No	-	0	0.00	0.77	1.00	-1.29
38	B_Stance_Southpaw	✗ No	-	0	0.00	0.20	1.00	1.53
39	B_Stance_Switch	✗ No	-	0	0.00	0.03	1.00	5.41
40	R_Stance_Open_Stance	✗ No	-	0	0.00	0.00	1.00	17.49
41	R_Stance_Orthodox	✗ No	-	0	0.00	0.76	1.00	-1.23
42	R_Stance_Southpaw	✗ No	-	0	0.00	0.21	1.00	1.44
43	R_Stance_Switch	✗ No	-	0	0.00	0.03	1.00	5.81
1	Winner	✓ Yes	2	Red(68%)	-	-	-	-

Table 2) Statistics of columns.

The intention was to predict the results of the matches, without any knowledge of the internal data of the match. Hence, columns consisting of match level data (e.g: Location, Data, ...) were dropped. This resulted in `UFC_Preprocessed` dataset containing 39 columns.

In our approach we utilised CRISP-DM methodology to explore data, train and evaluate ML models and deploy the business objective.

Data Preprocessing

`UFC_Preprocessed` has an overall sparsity of ~12% which may lead to 35% data loss if rows containing NA were to be dropped without a missing value treatment.
The following list, briefly describes the initial processes taken place:

- Replacing empty string with NA
- Removing 'Draw' matches (The problem is considered to be a binary classification, win/loss only)
- Distinguishing numeric & symbolic fields
- Removing constant columns (due to no variation in them)
- Formating data to 3 Decimal Points
- 1-hot-encoding categorical fields, `B_Stance` and `R_Stance`

Missing Value Treatment

Next was to have a missing value treatment plan to avoid large data loss. For this, physical aspects of fighters, `Height`, `Weight`, and `Reach` were considered. Firstly, as there was only 0.12% missing values in Height, the NAs were replaced with the median of the column.

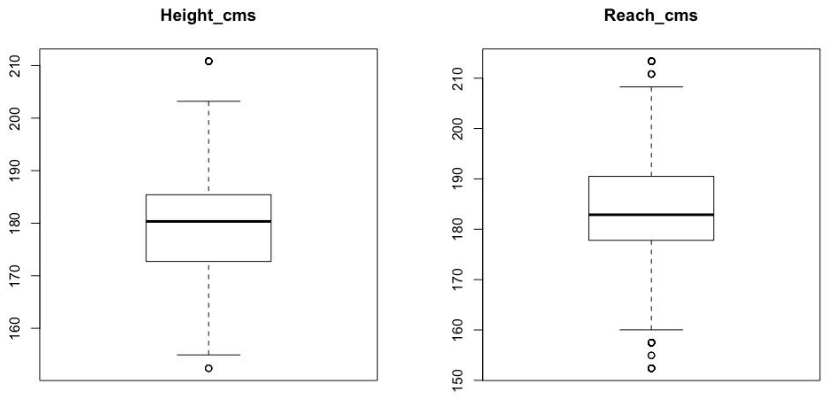


Figure 3) Height and Reach Box plots.

Furthermore, Correlation analysis on these 3 attributes, suggested that Height and Reach have an 89% correlation.

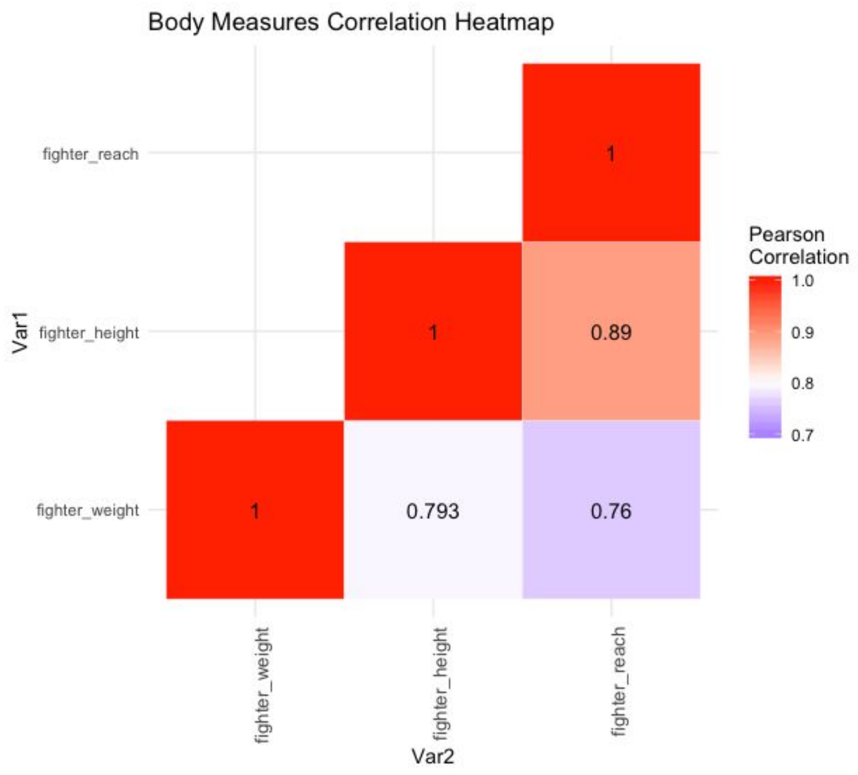


Figure 4) Correlation Heatmap.

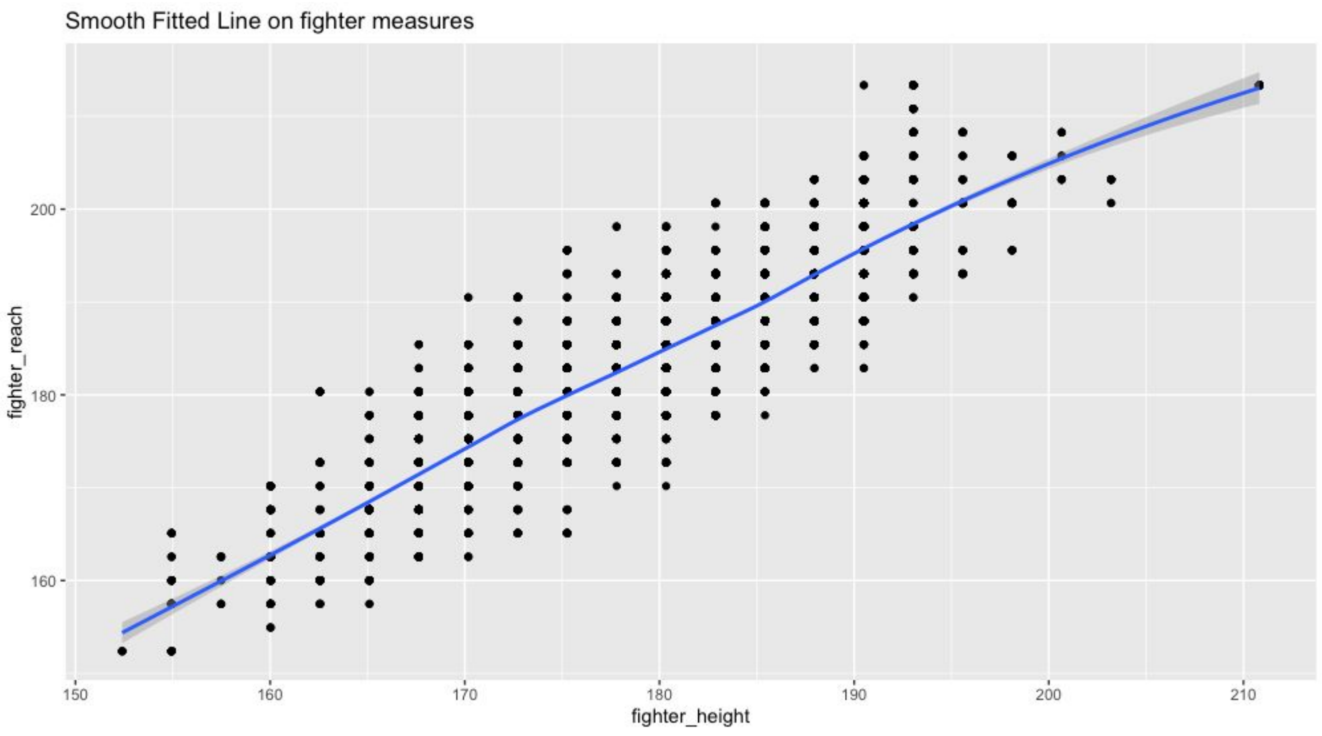


Figure 5) Smooth fitted line on fighter Measures.

The above diagrams indicated a Linear Regression model may be used to predict the ~9% missing values of `Reach` based on the `Height`.

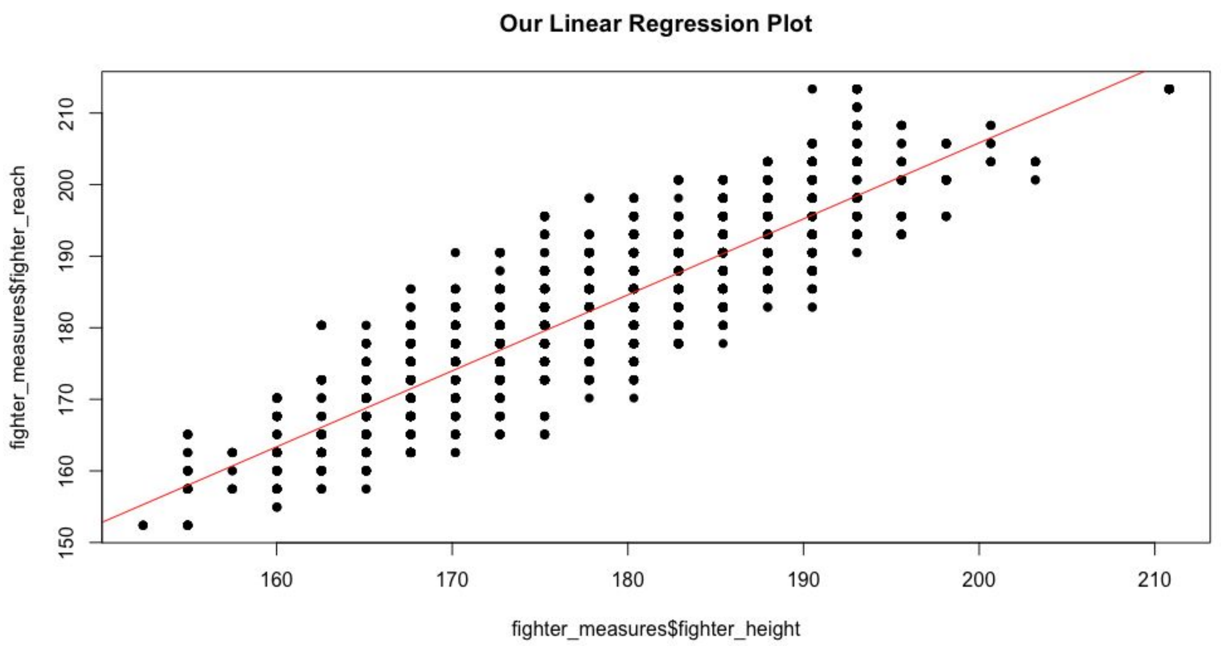


Figure 6) Linear Regression (LR) plot.

Finally, any remaining NAs were to be dropped. Consequently, the team managed to minimise loss of data to 9.9%.

The figures below summarise the remaining data:

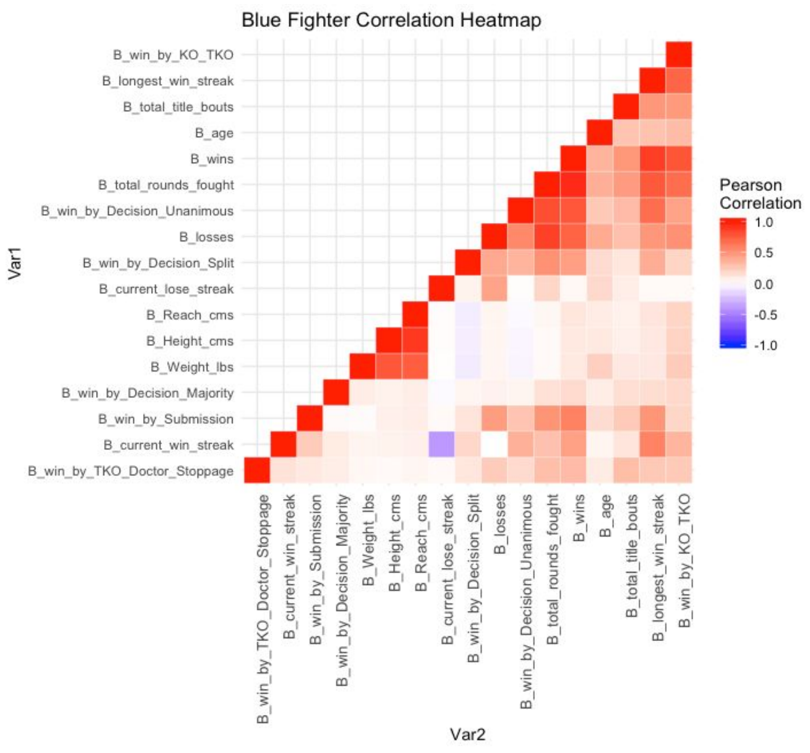


Figure 7) Blue fighter correlation Heatmap.

Dimensionality Reduction with automated PCA

In the long run, large dimensions of data will lead to computational expenses; dimensionality reduction techniques may be used to tackle this problem. In order to analyse the performance of the models on a more reduced dataset in terms of dimension, the `perform_PCA()` function was introduced.

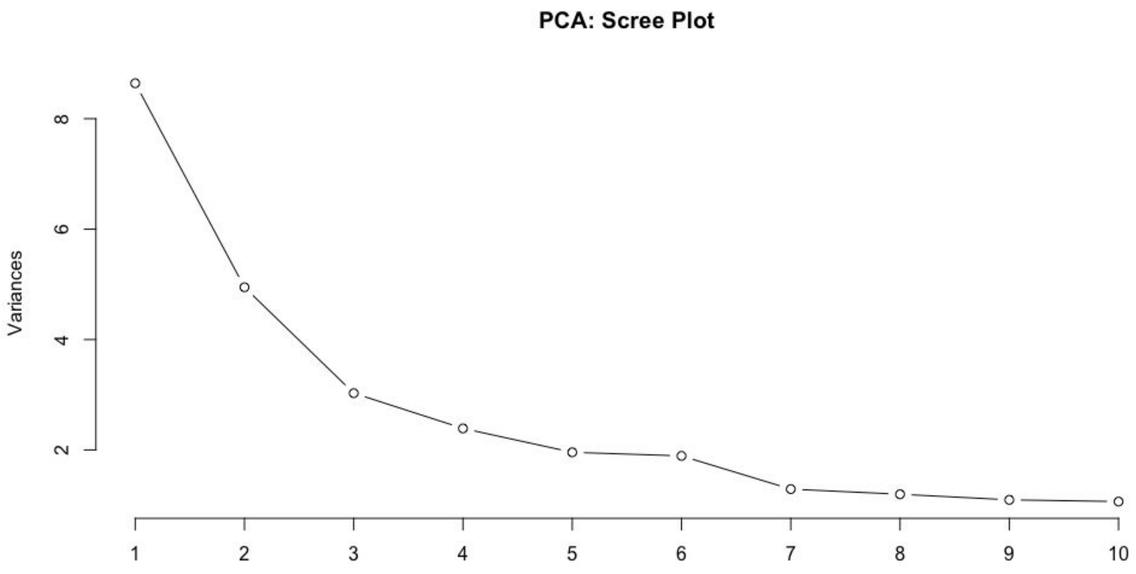


Figure 8) PCA scree plot.

Scree plot and Kaiser criterion are two methods for selecting the number of principal components, since Scree plots are interpreted visually we considered Kaiser criterion (variance larger than 1) as our method for selecting principal components.

The function is generalised and can automate the PC analysis on a given dataset. Here is the output of the function on `UFC_Preprocessed` dataset:

```
B_current_win_streak                    R_Reach_cms
R_losses                                B_current_lose_streak
R_Stance_Southpaw                       B_Stance_Southpaw
R_Stance_Switch                         R_win_by_Submission
B_win_by_Decision_Split                 B_Stance_Switch
R_Stance_Open_Stance
```

(11 Most important Fields)

Scaling and Normalisation

Finally, `normalised()` and `scale()` functions were introduced in order to execute necessary procedures prior to any ML modelling.

In summary, `ufc_PREP.R` script executes all above preprocessing requirements, displays some visualisations and produces a few CSV files including 2 main ones, `UFC_FINAL.csv` and `UFC_PCA.csv`.

Data Modelling and Training

Throughout the project, the group worked on different machine learning algorithms in order to enhance the output performance of the overall outcome. Built models along with their methodology and performance are stated here.

K-Nearest Neighbours

The K-Nearest Neighbours (KNN) algorithm is a great approach when dealing with numeric variables. The algorithm is applied to the training data, and then on the testing. We have designed function `perform_knn()` in our code; this function performs the KNN algorithm on any inputted data frame. In order to unify all the values in the data frame, values had to be normalised prior to the KNN performance. Hence, by creating function `normalised()` , based on the $zi = \frac{xi - min(x)}{max(x) - min(x)}$ equation, all values are normalised, where "zi" is the outputted number between 0 to 1.

The challenge with the KNN algorithm is determining the "k" value with the highest performance. Some references suggest $k = \sim\sqrt{(n)}$ where n is the number of instances in the training set. However, in order to actuate the "k" value with the highest accuracy, we evaluated all values in the range of 1-100, each running for 30 times; the average accuracy percentage of each value was then plotted, as in the graph below. In fact, the experiment was administered twice, initially on the original data set, and further when PCA was applied. The confusion matrix for Pre-PCA application and post-PCA application is also available in the appendix section. The graph below, also, clearly illustrates the KNN average accuracies for each "k" value. Hence, it is derivable that the accuracy of the KNN algorithm is higher on the original dataset, with peak accuracy value of $k = 73$.

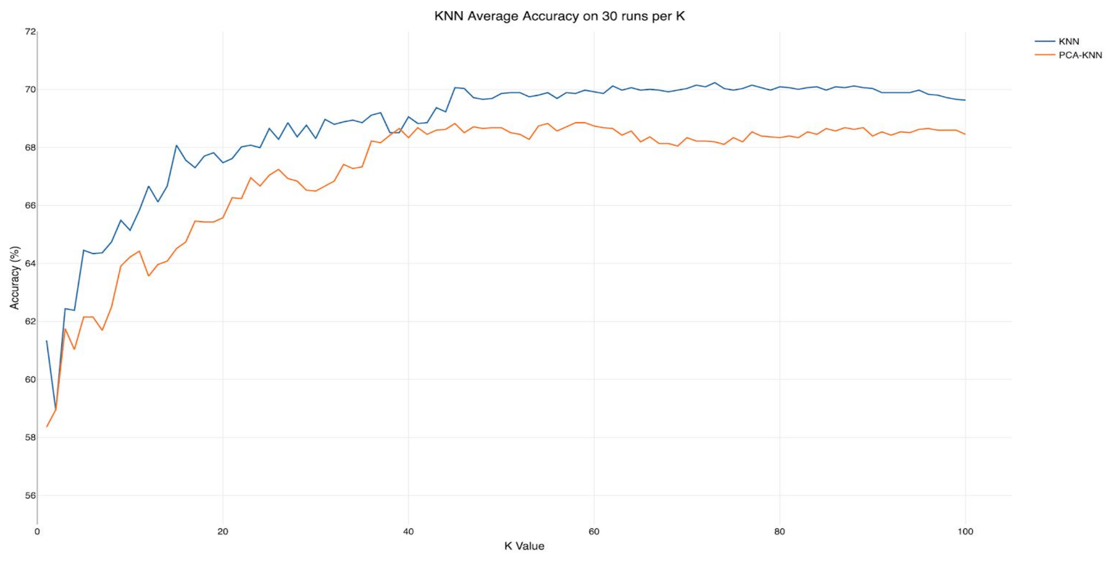
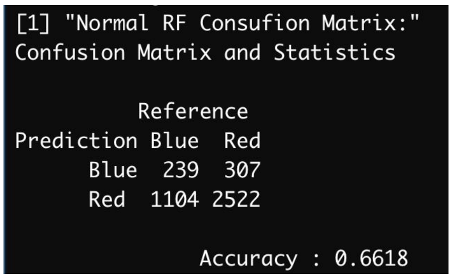


Figure 9) KNN accuracy output graph.

Random Forest

Another approach of modeling the data was considered using random forest which works well with both continuous and categorical data. It is made up of multiple decision trees (which alone tends to overfit) thus ensuring robustness against overfitting.
We used the function `randomForest()` in the *randomForest* R library to train the model on our training data set. Contrary to previous approaches, we did not normalise the data this time because random forests don't require data to be rescaled or transformed. We trained the model on both, PCA(principal component analysis) fields and without PCA fields.

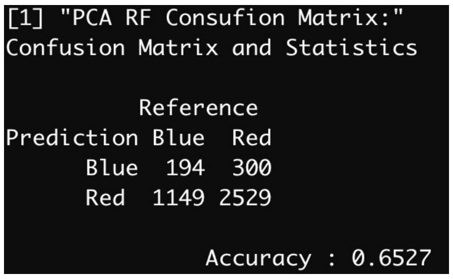
The confusion matrix for both the approaches is mentioned below:



```
[1] "Normal RF Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction Blue  Red
      Blue  239  307
      Red  1104 2522

               Accuracy : 0.6618
```

With PCA

Without PCA



```
[1] "PCA RF Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction Blue  Red
      Blue  194  300
      Red  1149 2529

               Accuracy : 0.6527
```

Here we concluded that using the data set with PCA variables did not have a significant change on accuracy so we decided to deploy the model with full variables for better precision.

The formula used for calculating the confusion matrix is mentioned below :

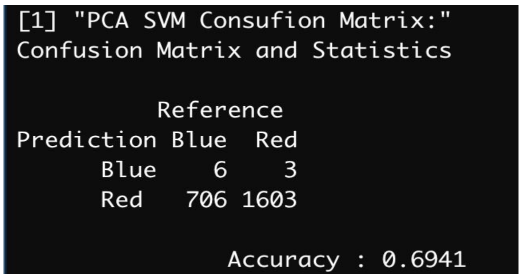
$$Accuracy = \frac{TP + TN}{TP+TN+FP+FN}$$

Support Vector Machine

SVM is another nonlinear modelling approach to solve classification problems. SVM uses a technique named as kernel trick to transform data and find optimal boundaries. We are using the *E1071* R library to perform SVM classification. In our model Radial Basis Functions was used as the kernel technique.

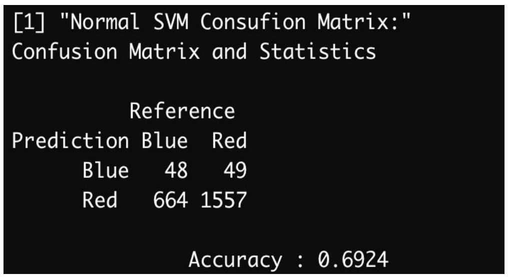
The confusion matrix for SVM implementation with reduced dimensions and the original dataset is displayed below.

With PCA



```
[1] "PCA SVM Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction Blue  Red
      Blue    6    3
      Red   706 1603

               Accuracy : 0.6941
```

Without PCA



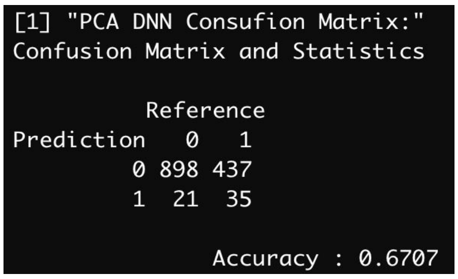
```
[1] "Normal SVM Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction Blue  Red
      Blue   48   49
      Red   664 1557

               Accuracy : 0.6924
```

Deep Neural Network

Our final approach was to use the Deep Neural Network. We used a *multi-layer perceptron* as the type of our neural network. The Neural Network contains 2 hidden layers with 3 and 2 nodes respectively. The R function we used to train this model is `neuralnet()`. We set the *threshold* value to 0.01 so the model continues to train until the loss function reaches a point that it does not go below the threshold value. The *step-max variable* was set to *1e6* to define the time interval for the model to converge.

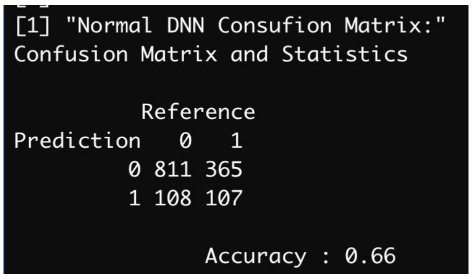
The confusion matrix for both approaches are listed below :

With PCA



```
[1] "PCA DNN Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 898 437
         1  21  35

               Accuracy : 0.6707
```

Without PCA



```
[1] "Normal DNN Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 811 365
         1 108 107

               Accuracy : 0.66
```

The training was again repeated with datasets with and without PCA variables. Our training models are displayed below.

There are, however, computational limitations for scaling the layers of the Neural network. That is specifically due to lack of resources (Stronger processing systems, such as GPU's) and time. Hence, in order to determine the number of neurons for algorithm, we used the following equation (Stackexchange, 2019):

$$Nh = \frac{Ns}{(\alpha \times (Ni + No))}$$

Where:
Ni = number of input neurons.
No = number of output neurons.
Ns = number of samples in the training data set.
α = an arbitrary scaling factor usually 2-10.

A better deep learning architecture could be defined by the calculations below.
Hence:
Ni = 42
No = 1
Ns = 3244
α = 5 (an arbitrary scaling factor usually 2-10).

→ Number of neurons in the first layer = ~ 15

DNN with PCA

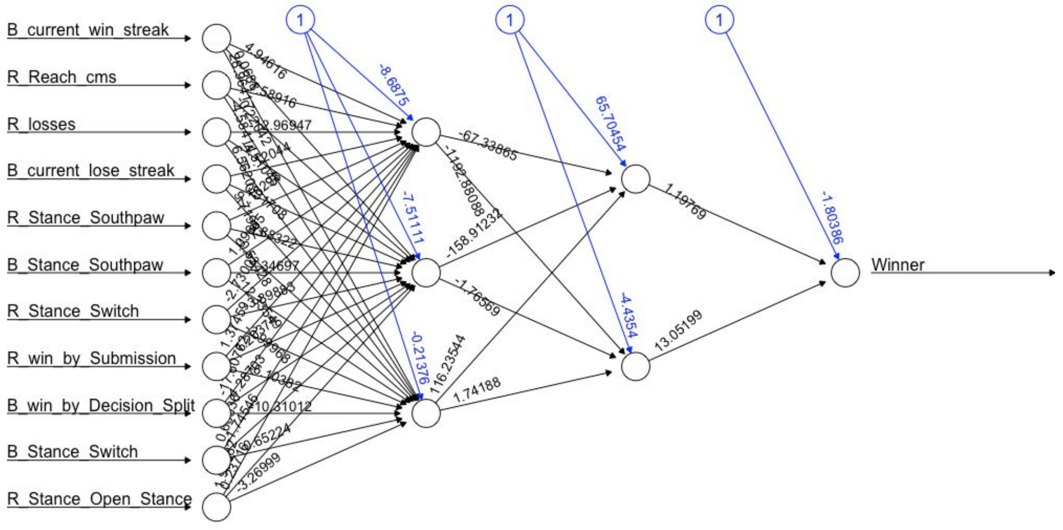


Figure 10) DNN with PCA.

DNN without PCA

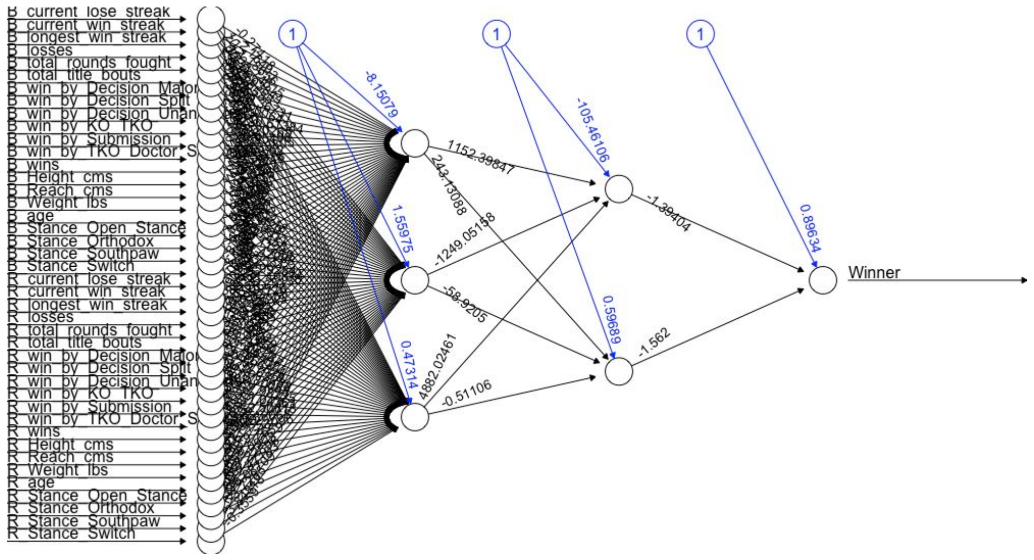


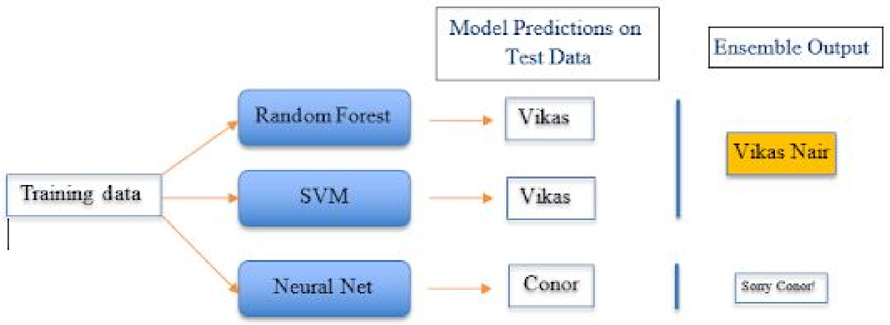
Figure 11) DNN without PCA.

Moving towards ensemble modeling

Taking inspiration from the architecture of Random forest, a set of weak classifiers could be turned into a model with comparatively high accuracy.

We thought of creating a simple ensemble architecture of combining three models and taking the majority response of the model predictions as the predicted value for each row of our testing dataset.

A diagrammatic representation of the architecture could be shown as below:



By adding all the three classifiers together the model performance remained the same (Mean accuracy of classification ~68%) suggesting the predictions by the majority of models do not vary much.

This could be studied further in detail as the situation could arise due to multiple reasons. A few points have been mentioned in the future scope section defining the scope of the ensemble analysis.

Deployment

To deploy our models we created a *Rshiny* web application. The web application has a simple interface that allows you to choose two fighters and predict the winner.

The layout of the app is as follows :

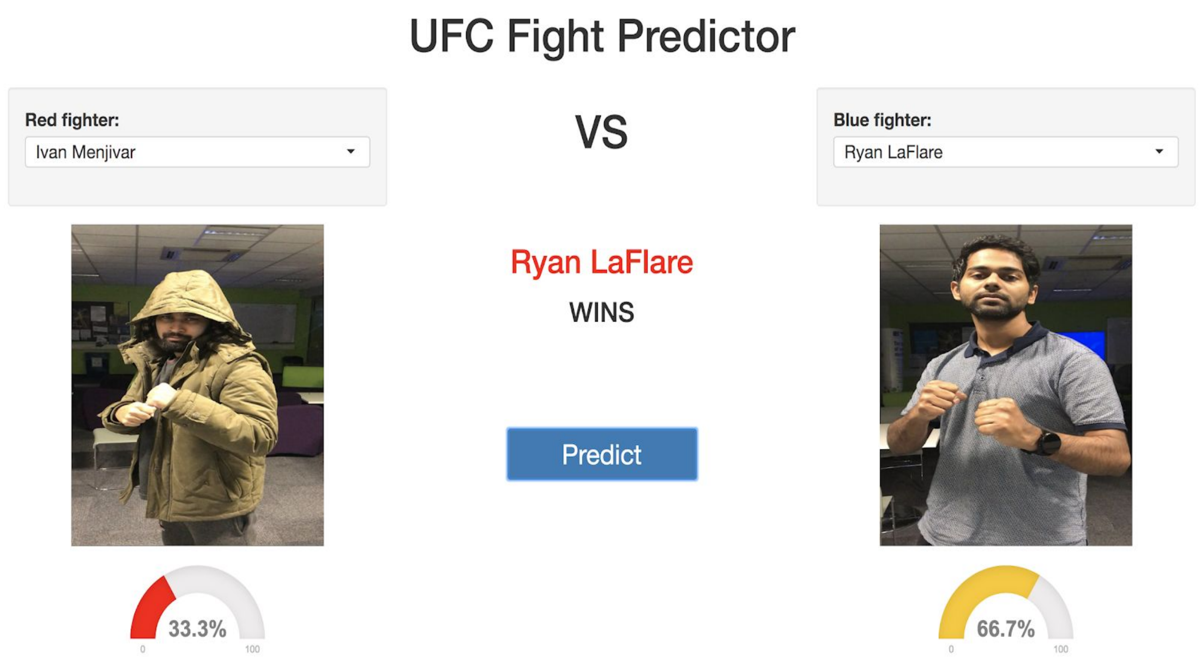


Figure 12) UFC Fight predictor dashboard.

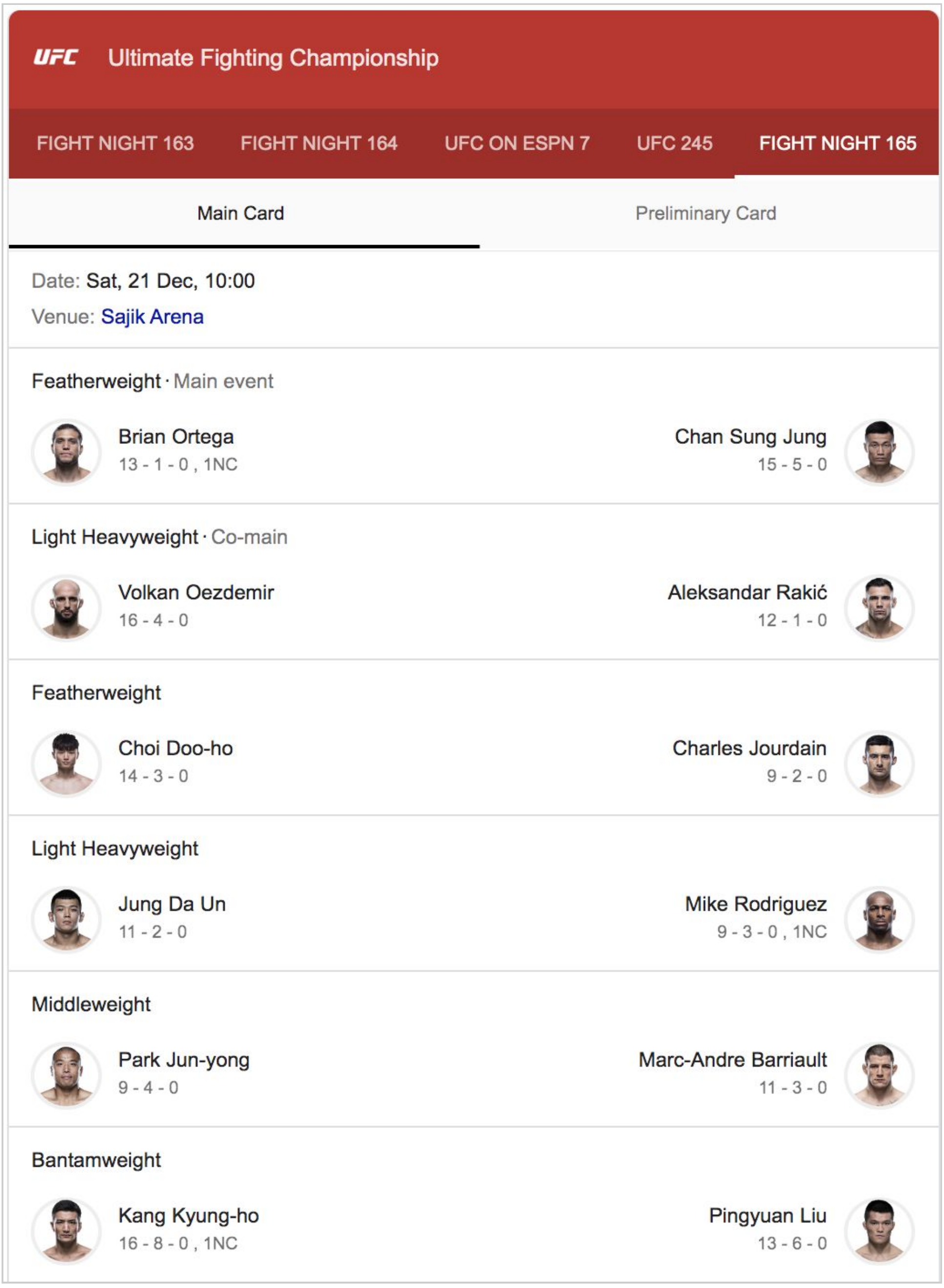
The output shows the winning probabilities of two fighters and predicts the winner based on the candidate that has a greater winning probability.

How The App Works

- The models are saved as .rdf files in the root folder.
- The web app passes the IDs of fighters when we click predict.
- The fighter data related to that ID is passed on to the trained model and the output is displayed on the web app.
- The app uses ensemble methods to combine the results of three models SVM, neural networks and random forest to produce a single output based on the majority vote.

Future UFC Fight Predictions

Using the aforementioned predicting tool, we have used it to predict the upcoming UFC event, taking place on the 21st of December 2019. Below are the results obtained from the app which are listed as follows:



UFC Ultimate Fighting Championship

FIGHT NIGHT 163 | FIGHT NIGHT 164 | UFC ON ESPN 7 | UFC 245 | FIGHT NIGHT 165

Main Card | Preliminary Card

Date: Sat, 21 Dec, 10:00
Venue: Sajik Arena

Featherweight · Main event
Brian Ortega 13 - 1 - 0 , 1NC — Chan Sung Jung 15 - 5 - 0

Light Heavyweight · Co-main
Volkan Oezdemir 16 - 4 - 0 — Aleksandar Rakić 12 - 1 - 0

Featherweight
Choi Doo-ho 14 - 3 - 0 — Charles Jourdain 9 - 2 - 0

Light Heavyweight
Jung Da Un 11 - 2 - 0 — Mike Rodriguez 9 - 3 - 0 , 1NC

Middleweight
Park Jun-yong 9 - 4 - 0 — Marc-Andre Barriault 11 - 3 - 0

Bantamweight
Kang Kyung-ho 16 - 8 - 0 , 1NC — Pingyuan Liu 13 - 6 - 0

Predicted Winner by our tool UFC
→ Brian Ortega
→ Volkan Oezdemir
→ Charles Jourdain
→ Mike Rodriguez
→ Park Jun-yong
→ Kang Kyung-ho

Figure 13) UFC events of 21st December 2019.

Table 3) Event winners.

Conclusion

To summarise, the initial dataset was preprocessed, NAs were replaced with suitable missing value treatment and categorical fields were 1-hot-encoded. Afterwards, PCA was performed to generate a separate dataset with principal components only.

Two processed datasets (with and without PCA) were fed in the Machine Learning models, namely KNN, RF, SVM, Logistics Regression and DNN, as part of the training. An average accuracy of ~68% was achieved whereas the performance and confusion matrix of each model stated specifically in the report.

The business objective of the project was achieved by developing an R-shiny application which may be used to predict future events through applying an ensemble method.

The initial hypothesis suggested that the win-loss ratio is an important factor in determining the winner. Although this is not declined in the analysis, PCA obtained factors such as 'current_win_streak', 'losses' and 'current_lose_streak' that correlate with the initial hypothesis.
Moreover, the generated figure from the single-layer perceptron neural network displays the most important factors from its perspective:

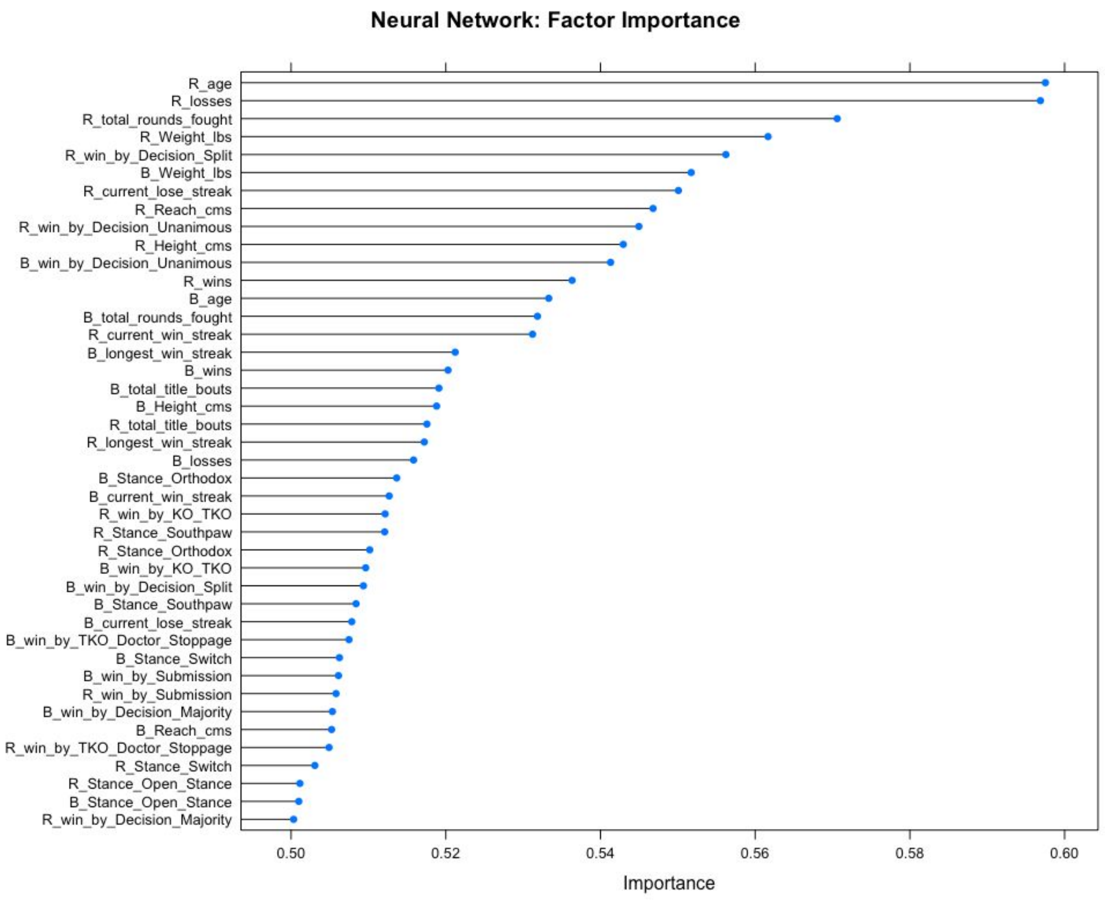


Figure 16) Neural Network

Limitations & Future Scope

The current ensemble method takes into account 3 predictions from SVM, DNN and RF models. Although an individual model provides a continuous [0,1] ranged output showing the probability of one class being true, the combination of these always outputs a final certainty in a 33% - 66% format. This is due to the current aggregation method used. To further enhance the ensemble method in such a way that an accurate probability is produced, the probability of probability needs to be calculated which introduces the concept of Dirichlet distribution. A business objective which can be achieved from this is to build systems to maximise profit and minimise risk.

Class Imbalance is a well-known obstacle in machine learning classification where one class dominates the other by having a larger ratio of observations. Two solutions are over-sampling occurrences of the minority class or under-sampling occurrences of the majority class. The disadvantage of the first is that by creating an exact copy of the minority instance, it increases the chance of overfitting. Whereas drawback of the latter is shrinkage of accuracy due to dropped majority samples. Nevertheless, under-sampling may be applied as more data is available for this problem.

Currently, the average accuracy of models is ~68%. However, as more data is available, by having larger portions of train/test sets, models can improve their performance.

Lastly, each model has a number of hyperparameters which can alter the accuracy. Hyperparameter tuning can take place in the form of a grid or random search to try to optimise the output of an individual model.

References

Hyndman, R. (2019). How to choose the number of hidden layers and nodes in a feedforward neural network?. [online] Cross Validated. Available at: https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw [Accessed 3 Dec. 2019].

Sirohi, K. (2018). K-nearest Neighbors Algorithm with Examples in R (Simply Explained knn). [online] Medium. Available at: https://towardsdatascience.com/k-nearest-neighbors-algorithm-with-examples-in-r-simply-explained-knn-1f2c88da405c [Accessed 16 Nov. 2019].

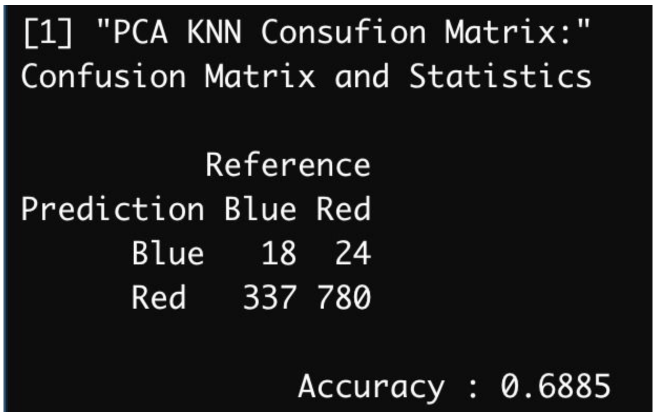
En.wikipedia.org. (2018). Ultimate Fighting Championship. [online] Available at: https://en.wikipedia.org/wiki/Ultimate_Fighting_Championship [Accessed 1 Dec. 2019].

The Telegraph. (2017). *What is UFC, what is MMA and what are the rules?*. [online] Available at: https://www.telegraph.co.uk/mma/0/ufc-mma-fighting-rules/ [Accessed 29 Nov. 2019].

Prof. Nick Ryman-Tubb (2019). *Lab sheets 1-7.* [documents and code] Available at: https://surreylearn.surrey.ac.uk/d2l/le/content/188750/Home [Accessed Nov. 2019]

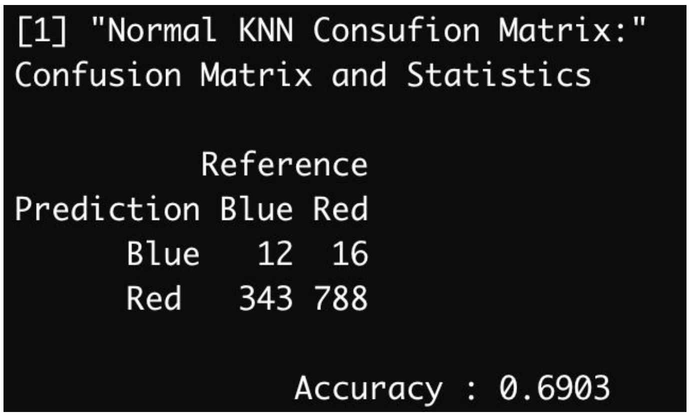
Google trends, (2019). [online] Available at: https://trends.google.com/trends/explore?date=all&q=ufc,%2Fm%2F01h7lh [Accessed 2 Dec. 2019].

Appendix



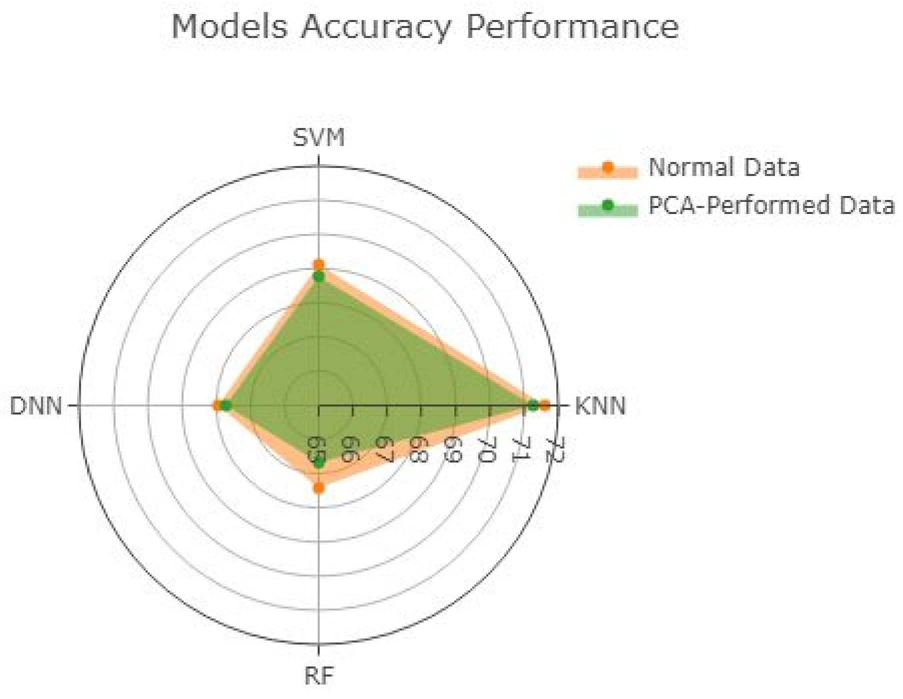
```
[1] "PCA KNN Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction Blue Red
      Blue   18  24
      Red   337 780

               Accuracy : 0.6885
```

A.1)

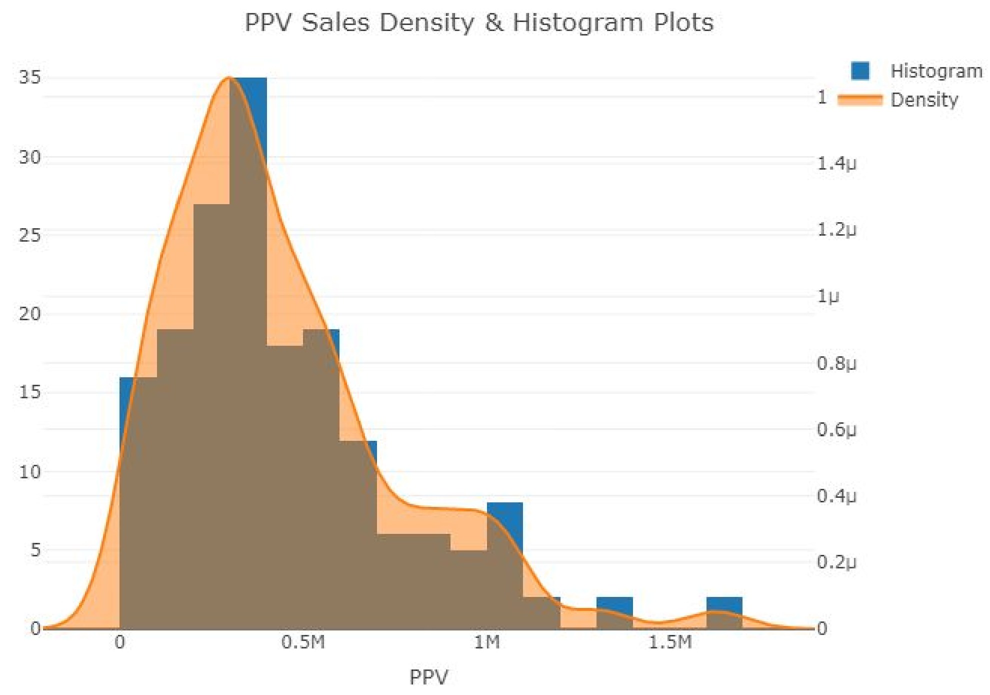


```
[1] "Normal KNN Consufion Matrix:"
Confusion Matrix and Statistics

          Reference
Prediction Blue Red
      Blue   12  16
      Red   343 788

               Accuracy : 0.6903
```

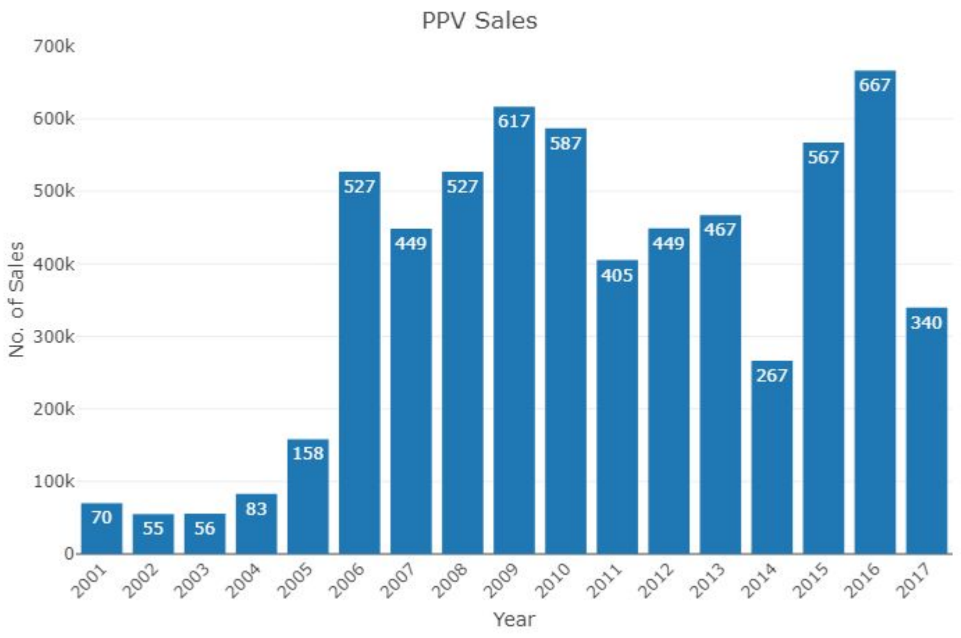
A.2)



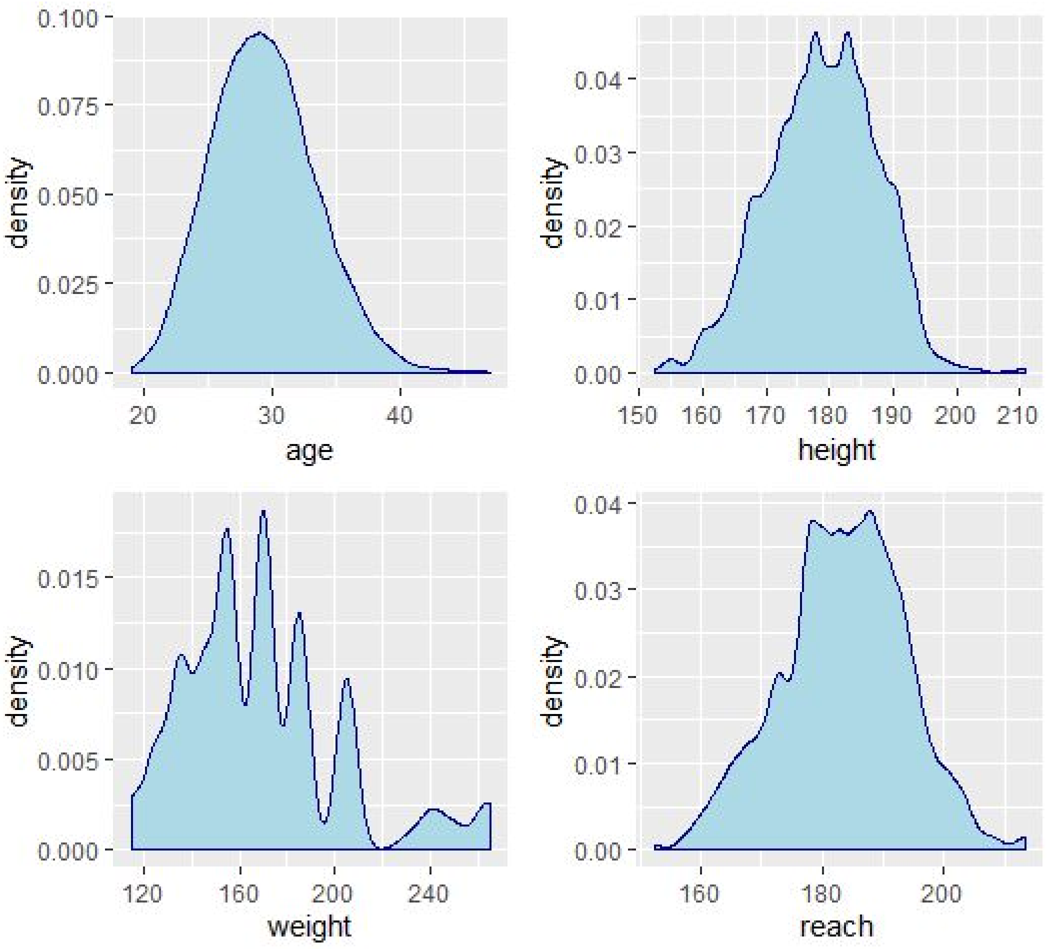
A.3) Models accuracy%.



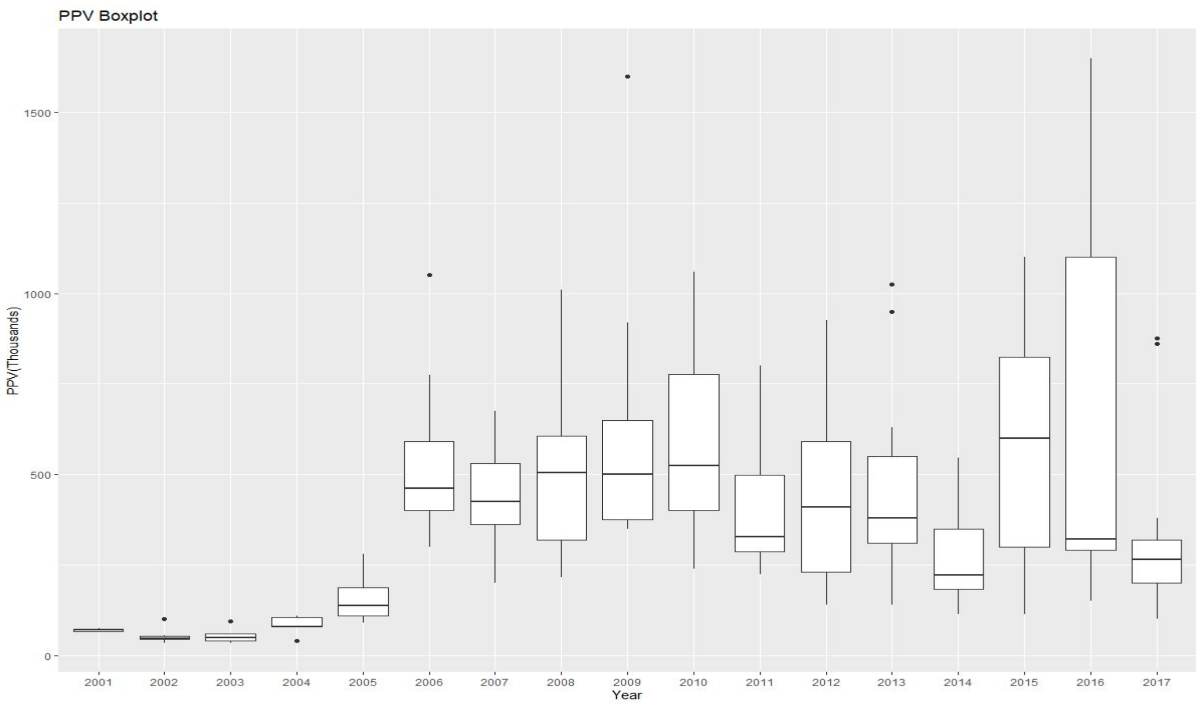
A.4) PPV sales density and histogram plots



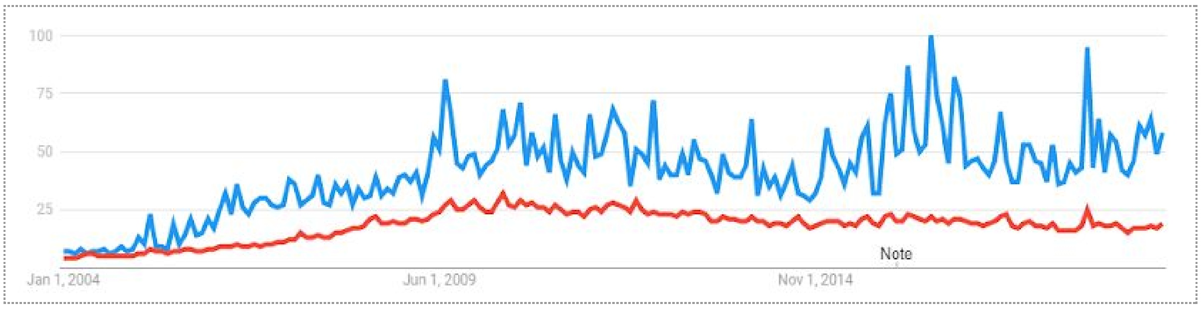
A.5) PPV sales 2001-2017



A.6) four density plots for fighters



A.7) PPV boxplot



A.8) Google trend comparison between UFC (blue) and MMA (red). (Google Trends, 2019)