Udacity - Machine Learning Nano Degree – Capstone Project

Calvin Wright - 8th June 2019

FIFA 2019 Player Data

UDACITY

Contents

Project Overview

Intro

Every year sports teams around the world are acknowledging more and more the importance and advantages there are to using data and statistics to prepare for the future. Huge investments are being made in data analysis research and development in an attempt to get ahead of the competition. The game doesn't end on the pitch or even in training anymore, but instead on a screen where complex analysis, modelling and predictions are made and reported back to staff who make critical decisions for the clubs. Companies such as Stats Sports and Catapult are winning new clientle all the time as any top ranking sports team that doesn't use data in someway is falling behind the curve. As well as that, it opens the door for poorer teams who don't have unlimited funding as now they can use that money more wisely in an attempt to level the playing field.

This project is based on that approach. Attempting to evaluate players based on their attributes. In this project I set out to determine two separate models that could predict a fair Value and Wage for a football player based on a set of attributes and characteristics. I used a dataset scrapped from the FIFA 2019 game which contained background information on the player such as Club, Age, Nationality and also their attributes such as Speed, Shot Accuracy, Defence etc. After reading Moneyball I have had a great interest in how sports analytics works and how it is pushing out the old way of scouting where people we often going off instinct and gut feeling. Players would be ignored if they didn't fit the image a scout had in their head before going in search. Models have no bias. For example it may be thought that all centre backs should be tall and a good defender will be ignored if they aren't above the average height. A machine learning model can both test this speculative assumption and also have some other counteractive attribute that was until then unknow like if in general defenders are better if they're tall but if they're speed is in the 80 percentile height is no longer as important so small and fast players will not be overlooked.

Overall this project would be related to a football team building a model that places fair prices on players for the transfer market and another model to decide a fair wage for the player. If the player's market value is greater than the predicted amount then they are therefore not worth buying. If a players market value is less than what the model predicts then the player is undervalued and should be considered. The same then goes for the agreed upon wage if the club would like to purchase the player.

Data

The dataset I used was of FIFA 2019 players and was downloaded from Kaggle, link in notes. This dataset contained 18,000 players(rows) and 52 attributes (columns). Originally I

examined the dataset via excel to see if it would be suitable for this project. After that it was imported to my python notebook and examined there.

```
data=pd.read_csv("FIFA.csv")
print("Number of rows : {}".format(len(data)))
data.head()
```

```
Number of rows : 18159
```

	ID	Name	Age	Nationality	Overall	Potential	Club	Value	Wage	Special	...	Penalties	Composure	Marking	StandingTackle	S
0	20801	Cristiano Ronaldo	33	Portugal	94	94	Juventus	77000000.0	405000.0	2228	...	85	95	28	31	
1	158023	L. Messi	31	Argentina	94	94	FC Barcelona	110500000.0	565000.0	2202	...	75	96	33	28	
2	190871	Neymar Jr	26	Brazil	92	93	Paris Saint-Germain	118500000.0	290000.0	2143	...	81	94	27	24	
3	155862	Sergio Ramos	32	Spain	91	91	Real Madrid	51000000.0	380000.0	2201	...	75	82	87	92	
4	177003	L. Modrić	32	Croatia	91	91	Real Madrid	67000000.0	420000.0	2280	...	82	84	60	76	

You probably recognise the names of what FIFA determines to be the greatest players in the world but may have known the vast amounts of money they are bringing home each week!

Many people think that wages have increased too much in recent years but it is a "fair" market after all. Teams technically should only be able to spend the same amount that they bring in on revenue. This project was not focussed on any trends that have taken place in recent years but just on fitting the current monetary state of the game.

From first glance you can see that Value and Wage do not perfectly correlate with each other. Some players can be valued less than others but still earn a higher wage than them. ID and Name were dropped from the data as they provide no real insight or information a model can use.

Then I took a quick examination of the fields I wanted to predict, Wage and Value.

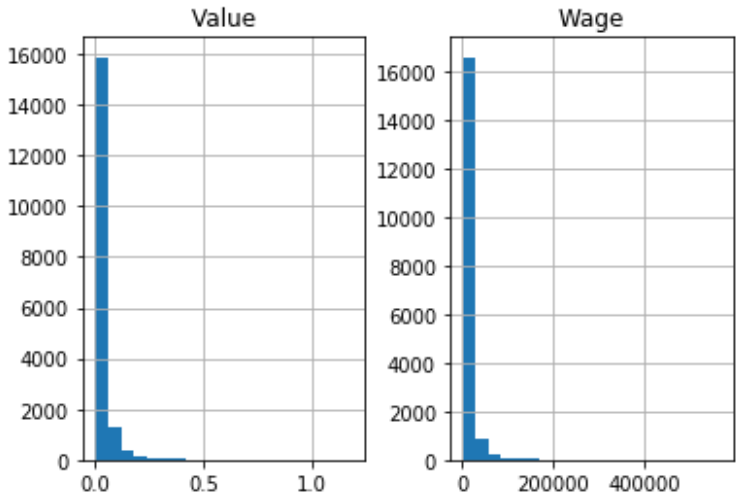
```
Wages Summary
count     17907.0
mean       9888.0
std       22149.0
min        1000.0
25%        1000.0
50%        3000.0
75%        9000.0
max      565000.0
Name: Wage, dtype: float64
```

```
Value Summary
count        17907.0
mean       2450133.0
std        5633207.0
min          10000.0
25%         325000.0
50%         700000.0
75%        2100000.0
max      118500000.0
Name: Value, dtype: float64
```

Straight away I noticed how uneven the data was with some players, such as those in the last chart are earning both valued and earning vast amounts more than the average player. This was something I had invisioned before starting the project and thought it would be a good idea to at some point removing outlier like Ronaldo and Messi from the data. And since the data has a finite starting point but no technical finite end point it is very skewed. Most players being built up before the average as it is also effected by the outliers, and then tailing off to the right where the premium players lye.

```
data[["Wage","Value"]].hist(bins=20)
```

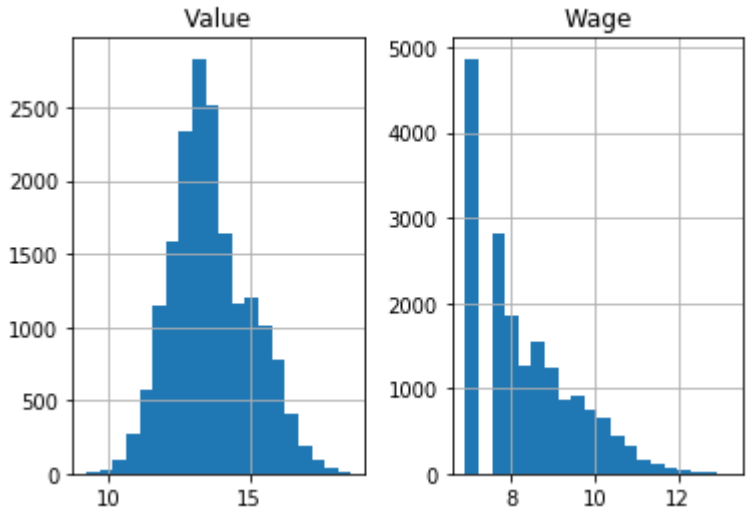
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000000002DABD048>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000001F4D5710>]],
      dtype=object)
```



Methodology

Data Cleaning and Manipulation

Due the skewed nature of the predictor variables seen in the last section I started off by normalizing these variables with a log transformation. The resulting histogram below shows a much more centred set of data.

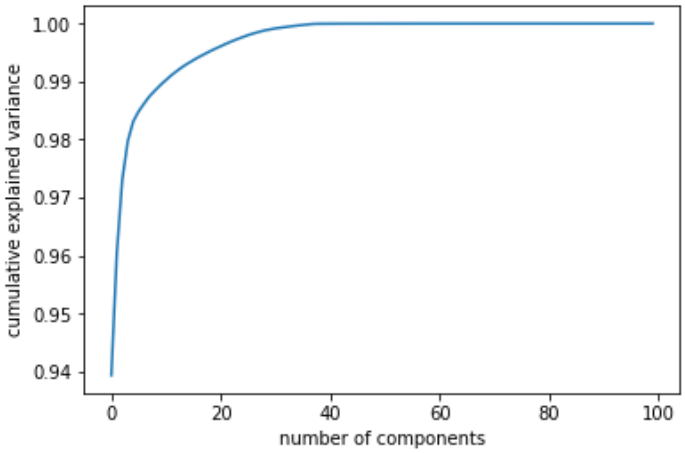


After this I split the data into training and testing data. This was done twice for Value and Wage, so there were two training sets and two testing sets. We had a training set of 14325 samples and a testing set of 3582 samples. Training and testing sets are created in order to fit a model to our training data and then test it's performance against another separate dataset. If the model only does well on the training set but not the testing set there is evidence of overfitting and we must simplify the model.

After one-hot encoding the dataset there were 896 features, many of which I had a suspicion would not be in anyway insightful to the prediction of either Value or Wage. For example the team and nationality may only be important for certain values. For example if a player is English could be a significant variable for multiple reasons. First, Football is very popular in England and having an English star on your team will make the fans their interested in your matches if they previously were not. This could increase the amount of money the team receive per match, increase jersey sales, tickets etc. Or, if the club is English themselves, they need a certain amount of home-grown players in the squad so nationality is important to them. The same may not be said for player from Angola for example where football isn't as popular and the national team are not well known. The same argument can be made for the club they play at. For this reason, I performed Principle Component Analysis (PCA) on the training set to find the top 100 features that explain the results. This allowed us to ignore meaningless data which would have vastly decreased the time needed to fit our model to the full set.

By applying PCA I reduced the dimensionality of the data form 896 input variables to 100 with practically no loss of information. At this point I considered a further reduction of the data as 95% of the information is held in the first 3 PCA components. However I left if for the time being and continued to fit a model.

```
Original number of features 896
Taking the top 100 features via PCA
[0.93942043 0.02107688 0.01260711]
The top 100 features can explain 0.999986701411 of the variation in the Value data.
```



First, I allowed the 100 features to remain in the model parameters, fit the model an ran a grid search on it to comprise the optimal model. However over these two steps I saw how slow it was to fit and train an SVM(2+ hours). On seeing this result I reduced the data down to 2 dimensions and still captured over 95% of the variability. To me this was an obvious choice as I am giving up very little variability but should have increased speed greatly.

For my model I wanted to use a SVM for regression (SVR). The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem.

The SVM

- Kernal: RBF
- Class Weight: Balanced
- C: ?
- Gamma: ?

I chose a radial basis function kernel model and then used the gridsearCV to find the optimal values for C and Gamma. The C parameter trades off correct classification of training examples against maximization of the decision function's margin. For larger values of C, a smaller margin will be accepted if the decision function is better at classifying all training points correctly. A lower C will encourage a larger margin, therefore a simpler decision function, at the cost of training accuracy. In other words``C`` behaves as a regularization parameter in the SVM. The behavior of the model is very sensitive to the gamma parameter. If gamma is too large, the radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with C will be able to prevent overfitting. When gamma is very small, the model is too constrained and cannot capture the

complexity or “shape” of the data. The region of influence of any selected support vector would include the whole training set.

GridSearchCV

I tried a variety of values for C and Gamma as seen in the above grid. Then each of them is used to fit a model. The best combination is our resulting model.

C	1,000	5,000	10,000	50,000	100,000
Gamma	0.0001	0.0005	0.005	0.01	0.1

the grid search achieved the following optimal models:

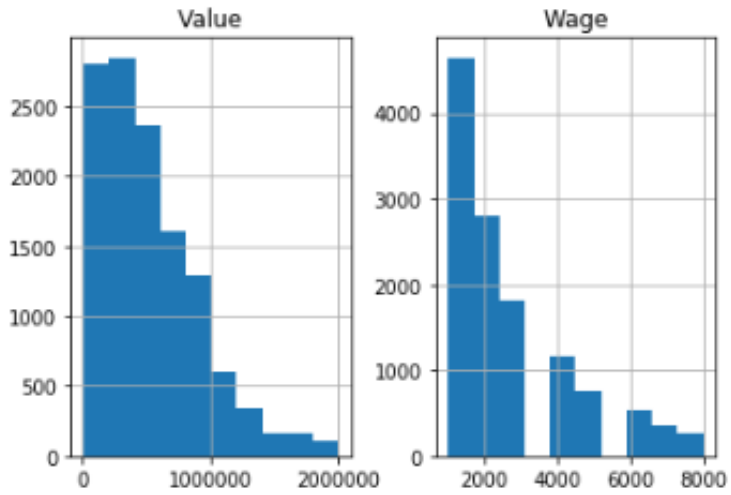
```
Best Wage estimator found by grid search:
SVC(C=100000.0, cache_size=200, class_weight='balanced', coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)


Best Value estimator found by grid search:
SVC(C=5000.0, cache_size=200, class_weight='balanced', coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

Evaluation

Once I had the optimal parameters for the model and fit it to the triainng set it was time to evaluate it with the testing set and see if it could be considered appropriate. An R-Squared test was deemed the most appropriate method of evaluation. R-squared (R2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. So a high R-Squared (near 1) means that all the variance in the prediction data (Wage/Value) can be explained by our player features.
I performed the R-squared evaluation on both models with their respective testing sets and they both performed quite well, getting above

Improvements

In an attempt to improve results I decided to remove the outliers from the dataset. I am also going to switch to using a decision tree based regression model to show variation in the final project. So bringing in the data again under a new name I filtered out samples that had a Value or Wage higher than the 75% percentile, which was £2,100,000 and £9,000 respectively. This left me with a dataset containing 12,304 samples. Although the data was still skewed in both cases it was much more centralized.



I then used a decision tree regressor to predict Value and Wages this time round. I utilised the gridSearchCV again to get the optimal max depth of the model. Here the Wage model had a max_depth of 4 and the Value model had a max_depth of 10. After fitting the models separately I re-evaluated them and saw better results. Now I had R-squared values of .91 for Wage and 1 for Value. Meaning both models were fit very well. In fact the Value model's score suggests that the model can explain 100% of the variation in the data.

Real Life Application

I work for a football club who has asked me to examine a player the head office wishes to purchase in the next transfer window. The club mentions that he is an excellent midfield player and that other clubs are interested which also indicates he is a good player. They give me his stats profile.

	Age	Overall	Potential	Value	Wage	Special	International Reputation	Weak Foot	Skill Moves	Height	...	Position_RB	Position_RCB	Position_RCM
245	30	82	82	16500000	46000	2061	1	4	2	175	...	0	0	1

It would cost the club £16,500,000 to purchase this player and then his wages would be £46,000.

Is he worth this much?

Running his profile through the model I got the following results:

```
if predictcheckValue>np.array(data_check["Value"]) and predictcheckWage>np.array(data_check["Wage"]):
    print("This player is under rated. We should buy")
else:
    print("This player is over rated. We should not buy")
```

```
For this player we predicted a Value of 64016129.0, and his actual Value was [102000000]
For this player we predicted a Wage of 461250.0, and his actual Wage was [355000]
This player is over rated. We should not buy
```

So, my advice would be not to buy this player. If instead we look at another player chosen from the same position at random we get the following results:

```
For this player we predicted a Value of 1814005.0, and his actual Value was [1600000]
For this player we predicted a Wage of 7980.0, and his actual Wage was [6000]
This player is under rated. We should buy
```

	Age	Overall	Potential	Value	Wage	Special	International Reputation	Weak Foot	Skill Moves	Height	...	Position_RB	Position_RCB	Position_RCM	F
5070	30	70	70	1600000	6000	1907	1	3	3	154	...	0	0	1	

Reflection

In summation I would like to say that I really enjoyed working with this dataset. There was a lot of cool information there. As I said, football is something close to me and I would love to work in the field of sports science using big data and machine learning.

I was surprised by the reduction in dimensionality using PCA I was able to perform and still have a highly successful model.

I saw great success in predicting both parameters with their unique models. I know this is something nearly all big football clubs do look into, but in greater detail and parameters that apply to a single club, their style of football, their budget and the players who they already have.

I would have like to have personalized the project in some way like this, i.e making it related to a specific club and thinking of the attributes they may be more focused on, or using their stats from last year and seeing where they needed improvements weather it be in defence or attack and having a bias for those players in the model.

References

1. FIFA Player Model - https://www.kaggle.com/aishwarya1992/fifa-data-analysis-player-value-prediction
2. MoneyBall Model
    a. https://towardsdatascience.com/linear-regression-moneyball-part-1-b93b3b9f5b53
    b. https://towardsdatascience.com/linear-regression-moneyball-part-2-175a9dc72e89
3. Udacity Examples
    a. https://github.com/udacity/machine-learning/blob/master/projects/capstone/report-example-1.pdf
    b. https://github.com/udacity/machine-learning/blob/master/projects/capstone/report-example-2.pdf
    c. https://github.com/udacity/machine-learning/blob/master/projects/capstone/report-example-3.pdf
4. Data - https://www.kaggle.com/karangadiya/fifa19