***DATA PREPARATION:***

Dataset given contains all data of attempts made by Ronaldo to score a goal

Found more than one column for

**[**remaining\_min,

remaining\_sec,

power\_of\_shot ,

Knockout\_match,

Distance\_of\_shot**]**

By using df[‘column’].describe() analysed which one is better column and which one is dummy

And also by looking at a few data points.

For example take Knockout\_match the value should show whether it is Knockout or not but

dummy\_column is showing other than 0 or 1 some random float number

So retained better one and dropped the other

Similarly for time and distance columns

Found some missing values in each column like around 5 % missing data in each column

area\_of\_shot object

shot\_basics object

range\_of\_shot object

For above categorical variables except replace missing values by mode

After replacing i compared % of goals scored before and after filling of missing values

Such that i avoid biased column, I used crosstab and matplot bar charts.

For Knockout\_match column replaced missing values with mode value

game\_season object

home/away object

For above columns I observed a trend i.e they occurred as set. So i used **for** loop to replace missing data by value right above it.

Thus make sure that proper data is filled in missing cells

For numeric

remaining\_min float64

power\_of\_shot float64

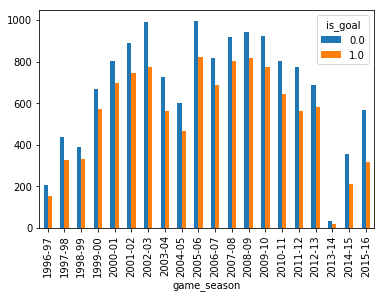
distance\_of\_shot float64

remaining\_sec float64

I used median replace its value before i checked **. describe**  to make sure that mean and median are close

***EDA***:

When replacing missing data i visualized data using matplot bar chart most importantly for categorical data



For example if we consider above plot which is for game\_season vs goals

I observed over seasons his performance/number of goals attempted first increased

Later reduced by the end i.e b/w 2014-2016 when we compare number of total goals attempted vs goals scored has lot of difference than remaining seasons.

I also observed that happened right after 2013-2014 where his attempts in the season fell drastically to very low value that is odd unless he played very few games or missed games due to injury . this injury may be reason to is later dip in his performance.

Also we can see it is important input column to build our model

I dropped unnecessary columns such as

match\_event\_id

location\_x

location\_y

match\_id

Team\_id

Which obviously has no impact on outcome i.e goal scored or not

To support the above step - say your marks does not depend on your phone number or roll number

When it comes to team name its same for all data points no need to consider

team\_name

Latitude longitude columns are ignored because we already have **home/ away** column

remaining\_min float64

power\_of\_shot float64

knockout\_match float64

game\_season object

remaining\_sec float64

distance\_of\_shot float64

area\_of\_shot object

shot\_basics object

range\_of\_shot object

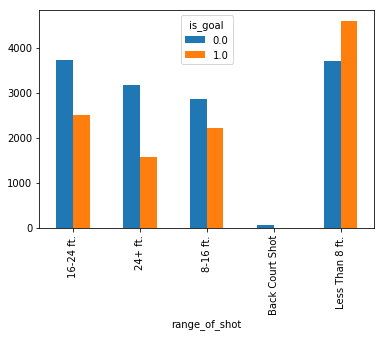
home/away object

type\_of\_shot object

type\_of\_combined\_shot object

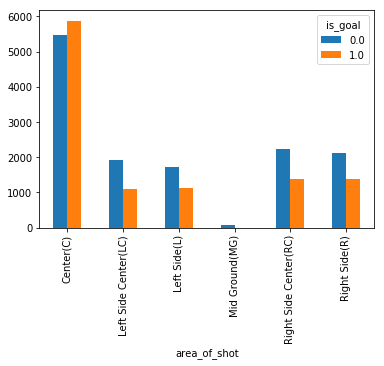
Above columns remaining I found them as potential independent variable/ features

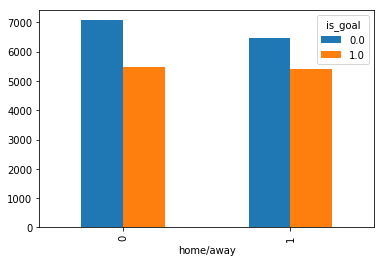
Few insights from plotted graphs

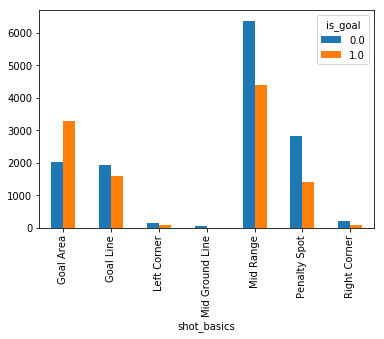


In the above graph we can observe **less than 8 ft** range has more scored goals than missed and also it occurred more than any other that makes it mode. When back court shot almost failed all times and also occurred very less compared to others. At first I thought it was an error in data but took chances and considered.

Below are a few of the graphs I visualized to know trends and other insights







I further simplified by combining mins and sec to mins column for example

Min-4.0

Sec-30.0

Min-4.5(total)

Into min not into secs because when you convert into seconds the value looks bigger when compared to other column values which may send wrong indication to model and model may be deviated. So i found it is better to keep in mins

When it comes to **type\_of\_shot** and **type\_of\_combined\_shot** noticed that for each data point only one of the columns is filled so I merged both but before that i changed small **s** of shot into **S** for **type\_of\_combined\_shot** such that when merged those act as different categories

Home or away as i didn't understand the meaning of data filled so I initially took each as a different category and then after building model realised its bringing down accuracy so considered all cells which has **vs** as one category and **@** as other this helped my model to predict better than before

I found better prediction when power of shot are taken as categorical instead of numerical

( You can find this in log\_reg\_2.py file)

***Model building:***

For building classification model basic method is Logistic regression

Then I tried SVC then gridsearch to hyper parameter tuning

To give model i have to change object type into model acceptable type so i used one hot encoder for all categorical

I split i**s\_goal** with **nan** as *final\_test* data and remaining as *final\_train*

I further splitted *final\_train* to create training and testing set

So that i can compare model accuracy on both training and testing data and omit overfitting

After I build Logistic regression on all *final\_train* then predicted probability and created dataframe using this probability values and shot id

As asked in sample submission

***Conclusion:***

Along with Logistic regression I have also built SVM and grid search model which got its best accuracy where hyperparameters fixed at {C=10, gamma=0.01, kernel='rbf'} after parameter tuning it took time to get these still its score is less. So i prefered logistic regression.

When i used neural networks got very less accuracy (<50%), (project files neural\_netwok.py)

Random Forest gives little better accuracy if we do parameter tuning.

For now my best accuracy at around 63% occurred on Logistic regression model.

I found that some times even though accuracy is low ***Mae*** is lesser compared to higher accuracy models.

I surmise that data cleaning is reason for less accuracy.

It could be better if I have certain game knowledge about football.

*#\*\*imp* ***.py*** *files for different models are uploaded in different file you can find in project files zip file\*\*#*