# Player Strategy Classification Using Logistics Regression and Neural Network

Saif Mathur
Amity University Mumbai
**1** PUBLICATION **0** CITATIONS
SEE PROFILE

# Player Strategy Classification Using Logistics Regression and Neural Network

Saif Mathur, Dr. Manoj Devare
*Amity University Mumbai, India*
*saifmathur41@gmail.com*

## ABSTRACT

*This article addresses the classification of the player strategy based on the feature vector available from the Fifa gaming website. The logistic regression with the sigmoidal function is used for finding the performance of the model, and once the performance is found at the over-fitted, the ANN model is taken into the consideration for evaluating the better performance of the model. The keras library is used for the model preparation.*

Keywords: Logistic regression, ANN, Overfitting, Underfitting, Variance, and Bias.

## INTRODUCTION

The data set used here is from the Fifa gaming website. The objective of this work is to classify the strategy of game plan by the individual player i.e. either the attacking as 1 or the defense as 0. The input is tuple of different attributes values. As the standard procedures firstly all the missing values are replaced with the median values. The Neural Network model is implemented used Kera where the input layer Neuron 29 attributes. There are also hidden and outer layer. The problem is classification problem where the linear model could be more bias and will not find the clear boundary the better choice was initially the Logistic regression.

There are total 17981 feature vector, which is sufficient number. As in case of the classification problems the maximum number of records for the one class are 5000. There are 75 attributes of each tuple, describing the player.

Xingbao Gao ; Li-Zhi Liao .(2010 ) mentions about the linear and quadratic programming problems in real time by introducing some new vectors. Tim Haifley (2002) mentions about the use of the linear regression. The discussion on the dimensionality reduction and Principal Components analysis is found for the specific applications. (Xingfu Zhang ; Xiangmin Ren (2011).

## METHODOLOGY

The numpy and pandas libraries of the Python are used to read the values from the dataset available in the .CSV values. The python code is described here.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

dataset = pd.read_csv('CompleteDataset.csv')
dataset.head()

columns_req = ['Acceleration', 'Aggression', 'Agility', 'Balance', 'Ball
control',
       'Composure', 'Crossing', 'Curve', 'Dribbling', 'Finishing',
       'Free kick accuracy', 'Heading accuracy', 'Interceptions',
       'Jumping', 'Long passing', 'Long shots', 'Marking', 'Penalties',
       'Positioning', 'Reactions', 'Short passing', 'Shot power',
       'Sliding tackle', 'Sprint speed', 'Stamina', 'Standing tackle',
       'Strength', 'Vision', 'Volleys', 'Preferred Positions']


columns_rearranged = ['Aggression','Crossing', 'Curve', 'Dribbling',
'Finishing',
       'Free kick accuracy', 'Heading accuracy', 'Long shots','Penalties',
'Shot power', 'Volleys',
       'Short passing', 'Long passing',
       'Interceptions', 'Marking', 'Sliding tackle', 'Standing tackle',
       'Strength', 'Vision', 'Acceleration', 'Agility',
       'Reactions', 'Stamina', 'Balance', 'Ball
control','Composure','Jumping',
       'Sprint speed', 'Positioning','Preferred Positions']

new_dataset = dataset[columns_rearranged]
new_dataset.head()

new_dataset['Preferred Positions'] = new_dataset['Preferred
Positions'].str.strip()
new_dataset = new_dataset[new_dataset['Preferred Positions'] != 'GK']
new_dataset.head()
new_dataset.isnull().values.any()
p = new_dataset['Preferred Positions'].str.split().apply(lambda x:
x[0]).unique()
df_new = new_dataset.copy()
df_new.drop(df_new.index, inplace=True)

for i in p:
    df_temp = new_dataset[new_dataset['Preferred Positions'].str.contains(i)]
    df_temp['Preferred Positions'] = i
    df_new = df_new.append(df_temp, ignore_index=True)

df_new.iloc[::500, :]

cols = [col for col in new_dataset.columns if col not in ['Preferred
Positions']]

for i in cols:
    df_new[i] = df_new[i].apply(lambda x: eval(x) if isinstance(x,str) else
x)

df_new.iloc[::500, :]

mapping = {'ST': 1, 'RW': 1, 'LW': 1, 'RM': 1, 'CM': 1, 'LM': 1, 'CAM': 1,
'CF': 1,
           'CDM': 0, 'CB': 0, 'LB': 0, 'RB': 0, 'RWB': 0, 'LWB': 0}

df_new = df_new.replace({'Preferred Positions':mapping})

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
df_new.iloc[:,0:29] = sc.fit_transform(df_new.iloc[:,0:29])
```

```
#df_new is cleaned at this part

X = df_new.iloc[:,:-1].values
y = df_new.iloc[:,-1].values

from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size =
0.2,random_state = 0)

from sklearn.dummy import DummyClassifier
dc = DummyClassifier(strategy = 'most_frequent')
dc.fit(X_train,y_train)

'''
from sklearn.linear_model import LogisticRegression
classifier_log = LogisticRegression().fit(X_train,y_train)
y_pred = classifier_log.predict(X_test)
acc_log = classifier_log.score(X_test,y_test)

import keras
from keras.layers import Dense
from keras.models import Sequential
classifier = Sequential()
classifier.add(Dense(output_dim = 15, init = 'uniform',activation =
'relu',input_dim = 29))
classifier.add(Dense(output_dim = 15, init = 'uniform',activation = 'relu'))
classifier.add(Dense(output_dim = 1, init = 'uniform',activation =
'sigmoid'))

classifier.compile(optimizer = 'adam',loss = 'binary_crossentropy',metrics =
['accuracy'])

classifier.fit(X_train,y_train,batch_size = 10,epochs = 100)
y_pred_ann = classifier.predict(X_test)
y_pred_ann = y_pred_ann > 0.5

from sklearn.metrics import confusion_matrix
cm_for_ann = confusion_matrix(y_test,y_pred_ann)

acc_ann = (4654/5451)*100
```

The code for the performance measurement and graph plotting is as

```
history = classifier.fit(X_train,y_train,batch_size = 10,epochs = 100)
#confusion matrix for model prediction
from sklearn.metrics import confusion_matrix
cm_for_ann = confusion_matrix(y_test,y_pred_ann)

#actual accuracy
acc_ann = (4654/5451)*100

#visualization
from ann_visualizer.visualize import ann_viz;
```

```
ann_viz(classifier, title="plot for project")

#summary for classifier
from keras.utils import plot_model
plot_model(classifier, to_file='model.png')

import matplotlib.pyplot as plt

# Plot training accuracy values
plt.plot(history.history['acc'])
#plt.plot(history.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training loss values
plt.plot(history.history['loss'])
#plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```

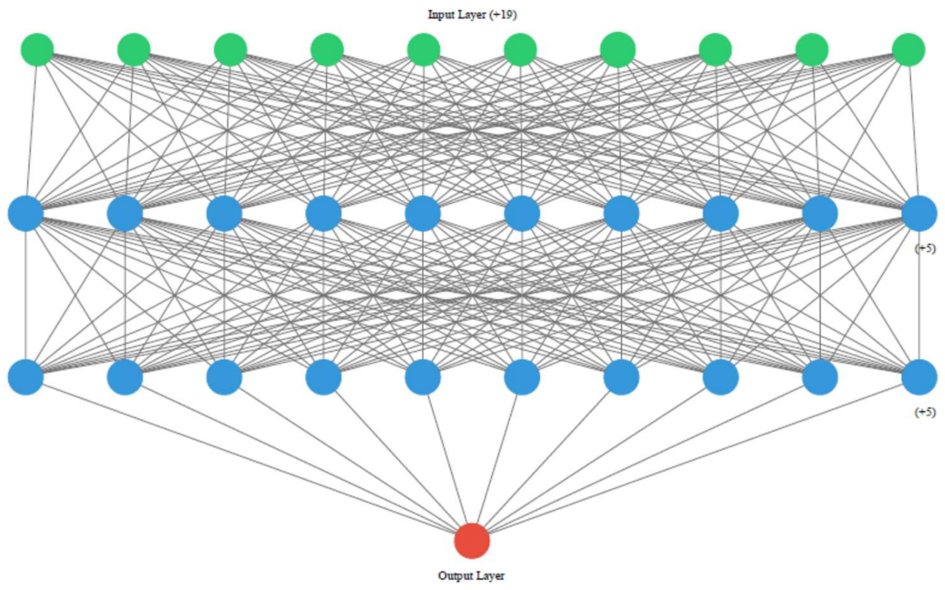

*Fig. 1: Graphical representation of the NN model*

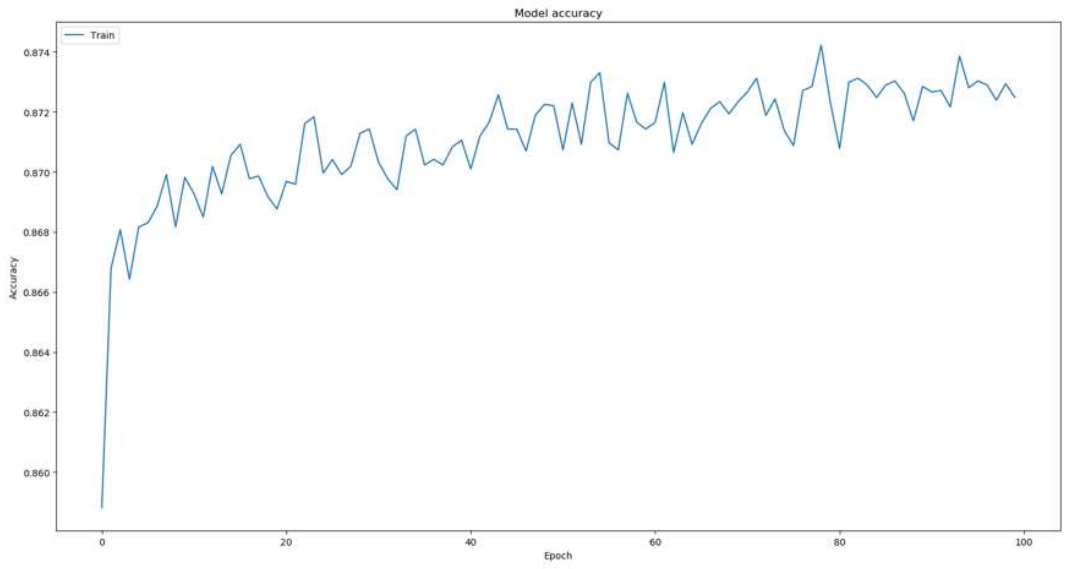

*Fig. 2: Model Accuracy versus the Epoch in NN.*

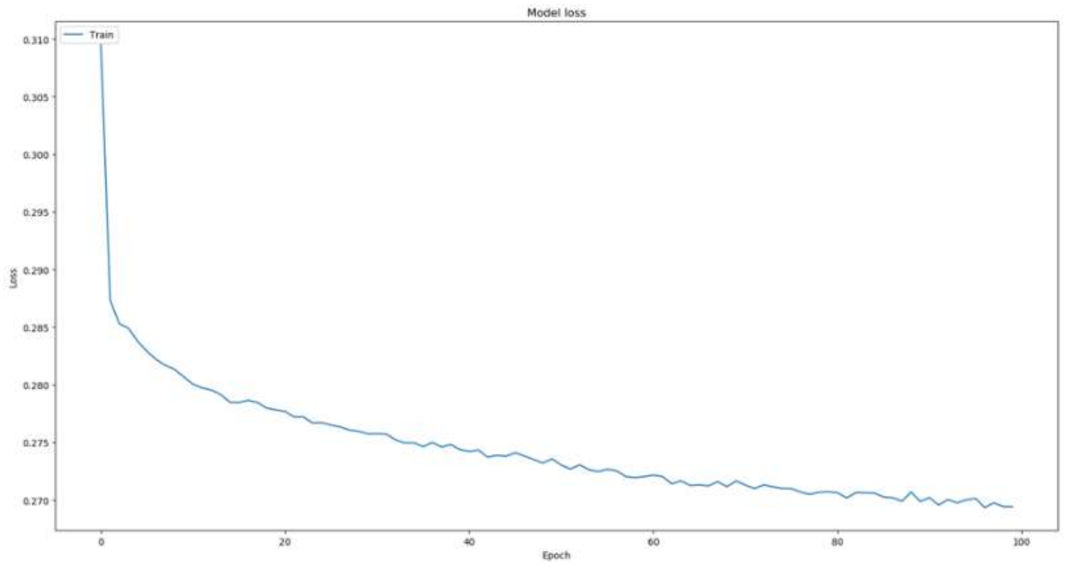

*Fig. 3: Model Loss versus the Epoch in NN.*

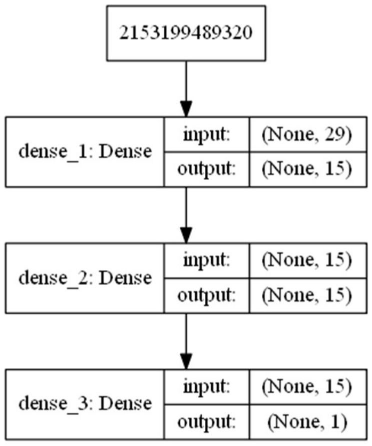

*Fig. 4: NN Design*

## RESULTS

As shown in the Table 1The logistic regression model shows the accuracy of 80%, and in case of Neural Network it is 87%.

*Table 1: Model and accuracy*

| **Model** | **Accuracy** |
|---|---|
| Logistic Regression | 80% |
| Neural Network | 87% |

It has been observed that, the model need to be chosen with proper care, perhaps the next model decision tree which will be provide better results.

## CONCLUSION

The balanced data which is having the sufficient number of tuples in the classes for the categorical problems are important. The dimensionality reduction has not been taken into the consideration as first of all researcher trying to find out the results without it. The principal component analysis (PCA) can be useful by calculating the Eigen Vector and corresponding Eigen values. As the ML suffers from the problem of the under fitting and overfitting, the choice of the dataset, its cleaning and proper regularization plays important role in finding the proper model.

## REFERENCES

Tim Haifley (2002). Linear Logistic Regression: An Introduction, IEEE IRW Final Report 184-187.

Xingbao Gao ; Li-Zhi Liao .(2010 ).A New One-Layer Neural Network for Linear and Quadratic Programming, IEEE Transactions on Neural Networks, 918 – 929.

Xingfu Zhang ; Xiangmin Ren (2011). Two Dimensional Principal Component Analysis based Independent Component Analysisfor face recognition 2011 International Conference on Multimedia Technology, 934 – 936.