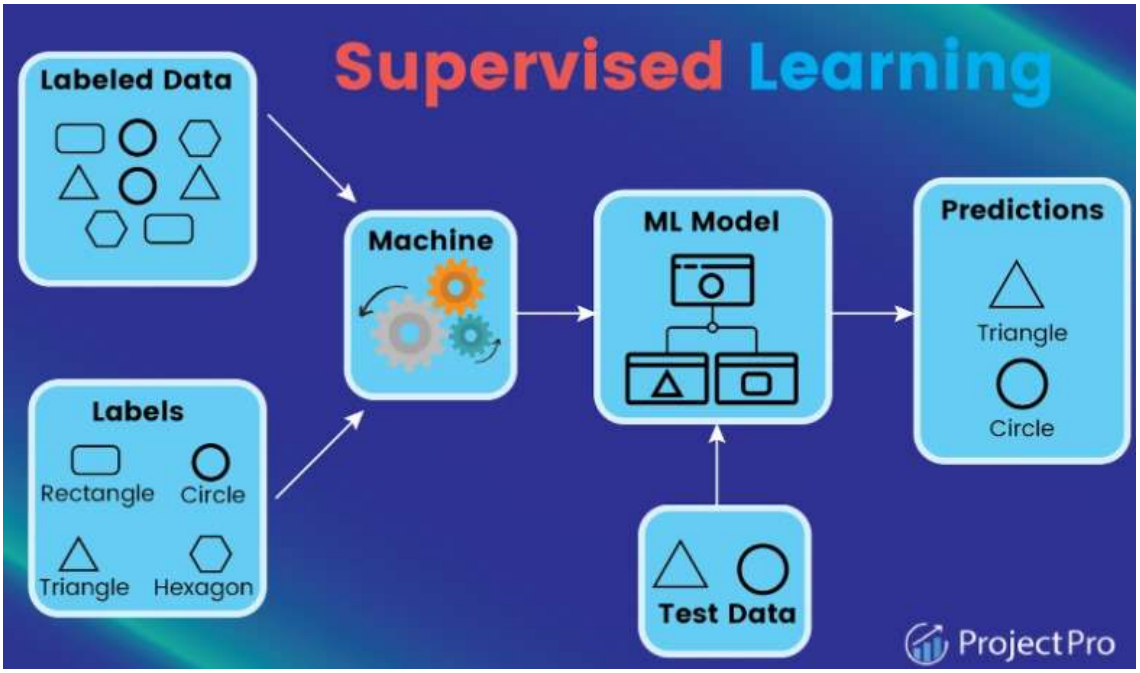
Name: Mahjabeen Mohiuddin
Subject: Advance Machine Learning and Algorithms
Student id :24610507
Course: Data Science and Innovation

The NBA Draft Prediction



Business Understanding:

The purpose of the project was to apply the supervised machine learning binary classifier predictive models using CRISP_DM(Cross Industry Standard Process for the Data Mining) approach on the NBA draft annual event dataset to help the business organizers to select the best players in the teams from American as well as international professional leagues to join their rosters.

The organizers of the NBA Draft event need an optimal solution that helps them in understanding the previous performances of players to plan their players selection strategies which in turn helps them to attract the spectators and this leads to grow exponentially in their business and avoids loss on various factors which could be seen in the machine learning models.

Therefore, this project will analyse and explore the data to pin out the set of variables that will help the business in understanding the patterns of players previous years performance to gain maximum profit and to avoid the exponential loses by foreseeing whether to take the players in the teams or not with the help of predicted values and AU-ROC Score.

The part of the project is to make models and select the best model based on AU-ROC Score closer to 1. We would know which of our models has AU-ROC Score closer to 1 by participating in the Kaggle competition on player selection for NBA.

DATA UNDERSTANDING

The data used for modelling in this project has 56091 observations and 64 features in the train dataset and 4970 observations and 63 features in the test dataset. The focus of each observation is on players previous performances and the available variables include the attributes if combined altogether gives the outcome whether the player is eligible to be selected in the NBA Draft teams or not.

```
Index(['team', 'conf', 'GP', 'Min_per', 'Ortg', 'usg', 'eFG', 'TS_per',
       'ORB_per', 'DRB_per', 'AST_per', 'TO_per', 'FTM', 'FTA', 'FT_per',
       'twoPM', 'twoPA', 'twoP_per', 'TPM', 'TPA', 'TP_per', 'blk_per',
       'stl_per', 'ftr', 'yr', 'ht', 'num', 'porpag', 'adjoe', 'pfr', 'year',
       'type', 'Rec_Rank', 'ast_tov', 'rimmade', 'rimmade_rimmiss', 'midmade',
       'midmade_midmiss', 'rim_ratio', 'mid_ratio', 'dunksmade',
       'dunksmiss_dunksmade', 'dunks_ratio', 'pick', 'drtg', 'adrtg',
       'dporpag', 'stops', 'bpm', 'obpm', 'dbpm', 'gbpm', 'mp', 'ogbpm',
       'dgbpm', 'oreb', 'dreb', 'treb', 'ast', 'stl', 'blk', 'pts',
       'player_id', 'drafted'],
      dtype='object')
```

	team	conf	GP	Min_per	Ortg	usg	eFG	TS_per	ORB_per	DRB_per	...	dgbpm	oreb	dreb	treb	ast	stl	blk	pts	player_id	drafted
0	South Alabama	SB	26	29.5	97.3	16.6	42.5	44.43	1.6	4.6	...	-1.941150	0.1923	0.6154	0.8077	1.1923	0.3462	0.0385	3.8846	7be2aead-da4e-4d13-a74b-4c1e692e2368	0.0
1	Utah St.	WAC	34	60.9	108.3	14.9	52.4	54.48	3.8	6.3	...	-0.247934	0.6765	1.2647	1.9412	1.8235	0.4118	0.2353	5.9412	61de55d9-1582-4ea4-b593-44f6aa6524a6	0.0
2	South Florida	BE	27	72.0	96.2	21.8	45.7	47.98	2.1	8.0	...	-0.883163	0.6296	2.3333	2.9630	1.9630	0.4815	0.0000	12.1852	efdc4cfc-9dd0-4bf8-acef-7273e4d5b655	0.0
3	Pepperdine	WCC	30	44.5	97.7	16.0	53.6	53.69	4.1	9.4	...	-0.393459	0.7000	1.4333	2.1333	1.1000	0.5667	0.1333	4.9333	14f05660-bb3c-4868-b3dd-09bcdb64279d	0.0
4	Pacific	BW	33	56.2	96.5	22.0	52.8	54.31	8.3	18.6	...	-0.668318	1.4242	3.3030	4.7273	0.8485	0.4545	0.3333	7.5758	a58db52f-fbba-4e7b-83d0-371efcfed039	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
56086	Niagara	MAAC	1	0.1	0.0	48.9	0.0	0.00	0.0	0.0	...	-17.439600	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	9eded9ee-0eb4-49f4-914a-f58924797bdf	0.0
56087	Northwestern St.	Slnd	1	0.2	206.9	35.9	100.0	102.56	0.0	0.0	...	9.392350	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	4.0000	5b539feb-1736-44ed-ba62-82bce86b12d6	0.0
56088	Texas Southern	SWAC	1	0.6	48.5	28.9	0.0	52.63	0.0	15.9	...	-3.240610	0.0000	1.0000	1.0000	0.0000	0.0000	0.0000	2.0000	e95d08eb-1273-4094-9173-c47368222c0d	0.0
56089	Vanderbilt	SEC	1	0.1	300.0	20.0	150.0	150.00	0.0	0.0	...	16.362500	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	3.0000	2f315a7b-2e82-44a2-8597-1779102ace09	0.0
56090	Chicago St.	WAC	19	21.9	55.0	16.4	24.5	30.60	0.6	7.2	...	-4.935520	0.0455	0.9091	0.9545	0.5909	0.4091	0.0000	2.4091	65571c18-b2b4-4ba6-84db-7357a76e6f59	0.0

56091 rows × 64 columns

Figure: Dataset brief

Out of sixty-four columns, thirteen columns found to be irrelevant and a few of them are identifiers in both training and testing datasets. These irrelevant columns will be handled in the data cleaning section of the report. The performance of the players is determined by the drafted column of the dataset which is a dependant variable. This dependant variable has the dependencies on remaining fifty variables of the dataset.

To conclude, the dataset has the relevant information that support the supervised machine learning to make a predictive model.

Data Cleaning and Preparation:

The dataset contains 64 features, and some of the features are irrelevant and a few among them are identifiers which may impact the model performance, so they are dropped. The selected features from train and test datasets that are useful for prediction are in numeric data type and their abbreviations are:

```
Index(['GP', 'Min_per', 'Ortg', 'usg', 'eFG', 'TS_per', 'ORB_per', 'DRB_per',
       'AST_per', 'TO_per', 'FTM', 'FTA', 'FT_per', 'twoPM', 'twoPA',
       'twoP_per', 'TPM', 'TPA', 'TP_per', 'blk_per', 'ftr', 'pfr', 'Rec_Rank',
       'ast_tov', 'rimmade', 'rimmade_rimmiss', 'midmade', 'midmade_midmiss',
       'rim_ratio', 'mid_ratio', 'dunksmade', 'dunksmiss_dunksmade',
       'dunks_ratio', 'pick', 'drtg', 'stops', 'bpm', 'obpm', 'dbpm', 'gbpm',
       'mp', 'ogbpm', 'dgbpm', 'oreb', 'dreb', 'treb', 'ast', 'stl', 'blk',
       'pts', 'drafted'],
      dtype='object')
```

Figure: Selected features

The features that are dropped are shown below.

```
'team','conf', 'stl_per','yr','ht','num', 'porpag','adjoe', 'year','type','adrtg','dporpag', 'player_id'
```

Figure : Dropped Features.

Dealing the nan values:

- Initially there were 185037 nan values in the selected features and these nan values are replaced using the custom functions such as median_null, mean_null and replace_null_with_Zero
- The nan values in the features ‘Rec_Rank’, ‘ast_tov’, ‘rimmade’, ‘rimmade_rimmiss’ and ‘midnade_midmiss’ are replaced by median value of each respective columns using custom function “**median_null**”. The remaining nan values are 128717.
- The nan values in the columns such as 'Rec_Rank', 'ast_tov', 'rimmade', 'rimmade_rimmiss', 'midmade_midmiss' are replaced by the mean value of each column by using custom function “**mean_null**” and remaining nan values in the dataframe are 5824.
- The remaining nan values are replaced by 0 using the custom function “**replace_null_with_Zero**”.
- Duplicated columns are analysed using the function “duplicated()” and found there is 16 duplicated rows in the training dataset and they were removed using drop.duplicated() function.

StandardScaler:

- StandardScaler is a function of sciket learn, it had help the data to scale and center the numeric variables. It has features such as Improves Model Convergence, Prevents Features Dominance, improves Model Performance, terpretability, Easier Comparison.

- Scaling the data to resize the distribution of values to make the mean of the observed values 0 and the standard deviation 1.

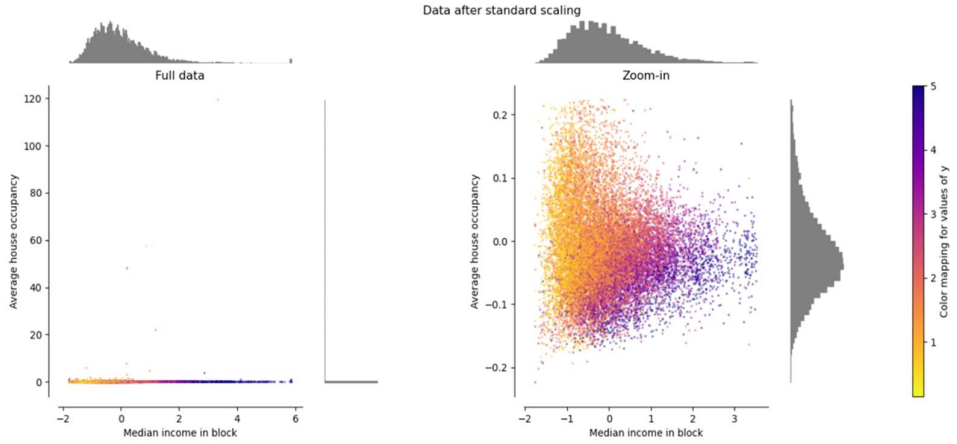


Figure: Standard Scaler

SMOTE Sampling:

- There were 56091 observations in the dataset, the processing time was too long, and there were many limitations of RAM, so the data was sampled with over sampling technique called as SMOTE sampler which is a class of imblearn package.
- Sampled the training dataframe using the SMOTE sampler with hyperparameters such as "sampling_strategy='auto', random_state=42, k_neighbors=5, n_jobs=None.
- The SMote sampler helps in balancing the unbalanced classes by combining the features of the target cases with the features of their neighbours.

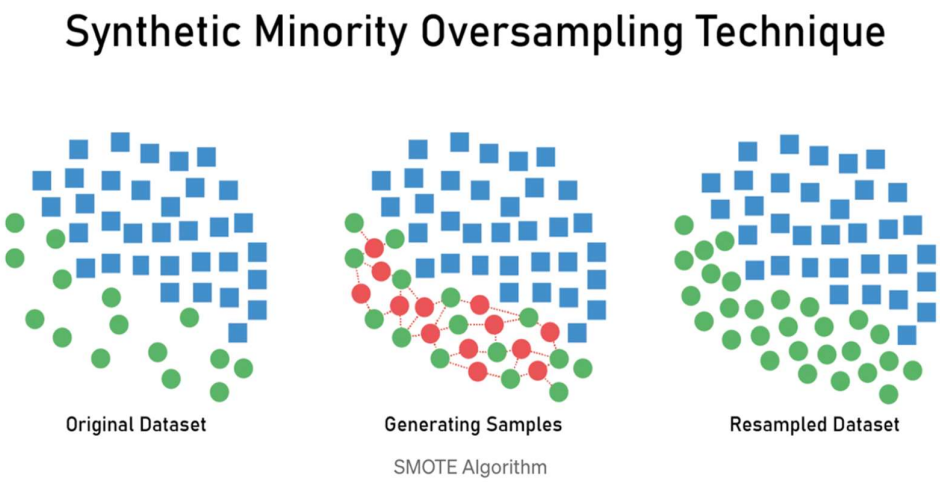


Figure: Smote Sampling Technique

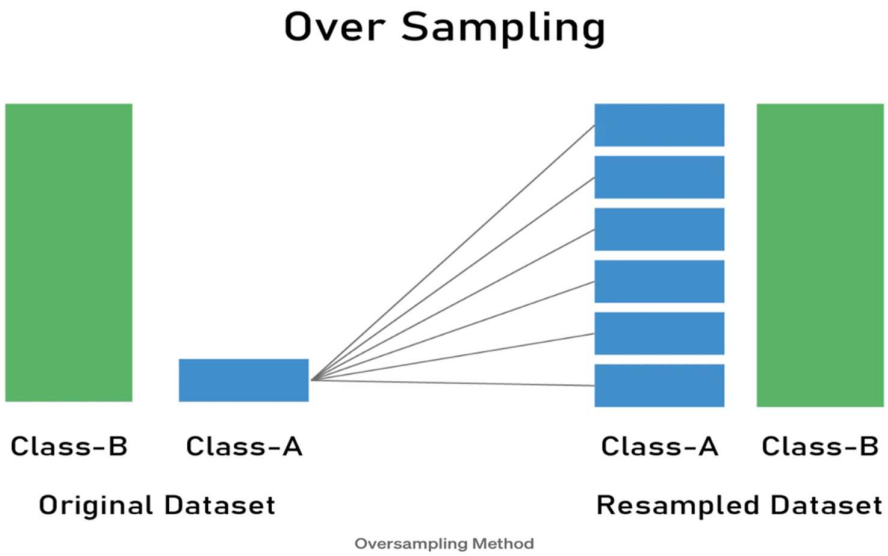


Figure: Sampling Process

SMOTE (Synthetic Minority Over-sampling Technique) is a technique for imbalance datasets, where the number of imbalance data points belonging to one class (the minority class) is significantly lower than the data points belonging to another class (the majority class). It is designed to achieve more balanced dataset by addressing the problem where it over samples the majority class to achieve more balanced dataset.

Splitting features and target :

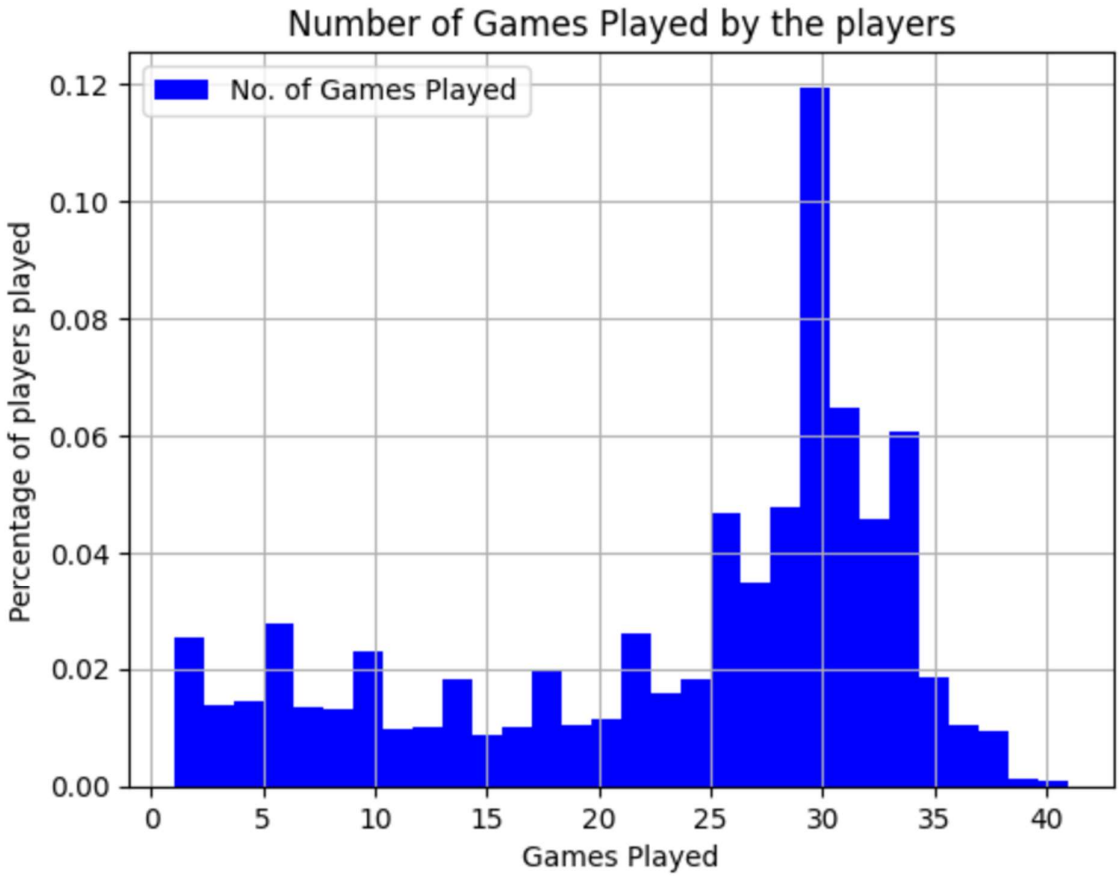
The column drafted is a target variable in the dataset and it is assigned to y, where as the remaining variables are considered as features and assigned to X using a custom function drop_target().

Split data into train and validation sets:

The train dataframe is split into train set and validation set in ratio of 8:2 with the help of scikit learn function called as train_test_split. And the test dataset has only limited number of observations and it is used to test the impact of prediction to get AU-ROC result from Kaggle.

Using a validation set to evaluate your model on unseen data to increases the generalising capability of the model.

Exploratory Data Analysis:



Figures: % of players Vs games played

From the above graph, it is noted that 12% of players played around 29-31 games.

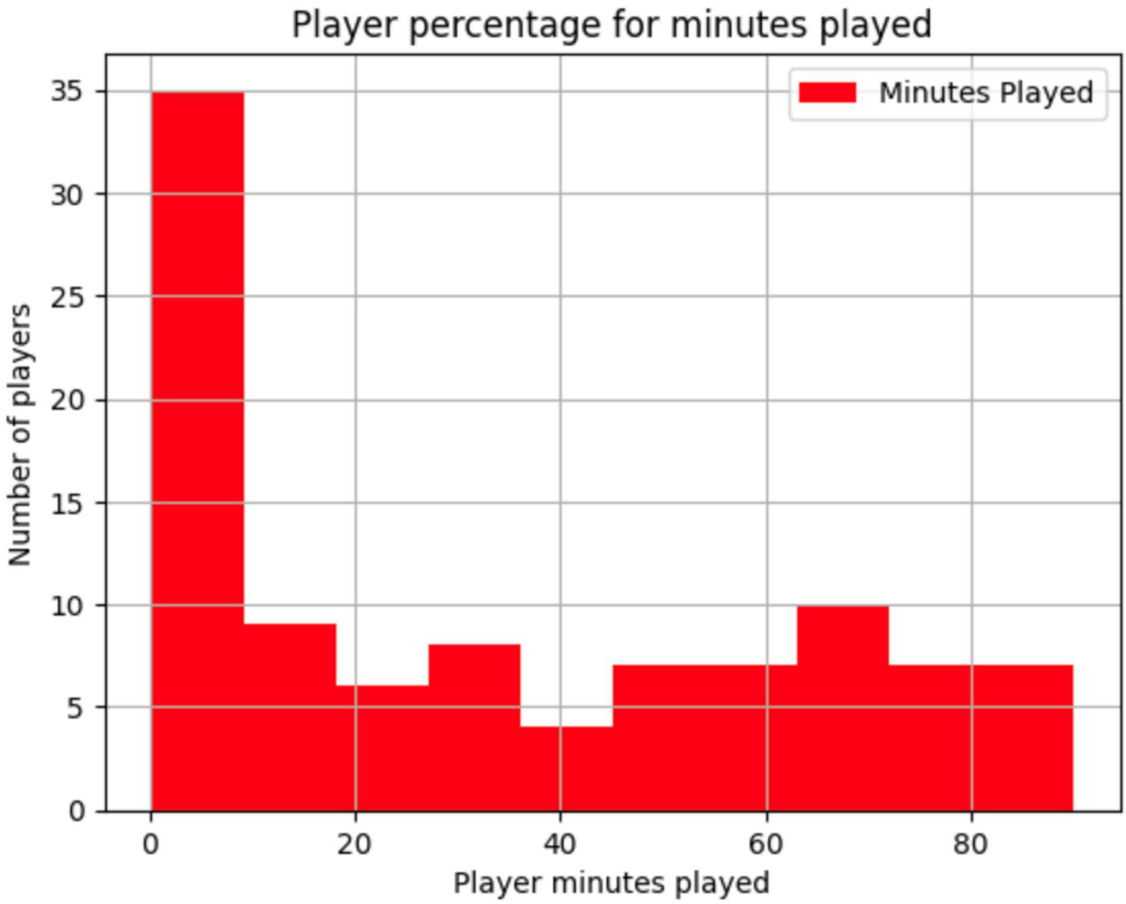


Figure: Number of players Vs minutes played by the players

The above graph was produced using 100 samples from the training dataset. The analysis from the above graph is that, 35 players could only play between 0 and 10 minutes whereas 10 players could last on the field from 63 to 70 minutes.

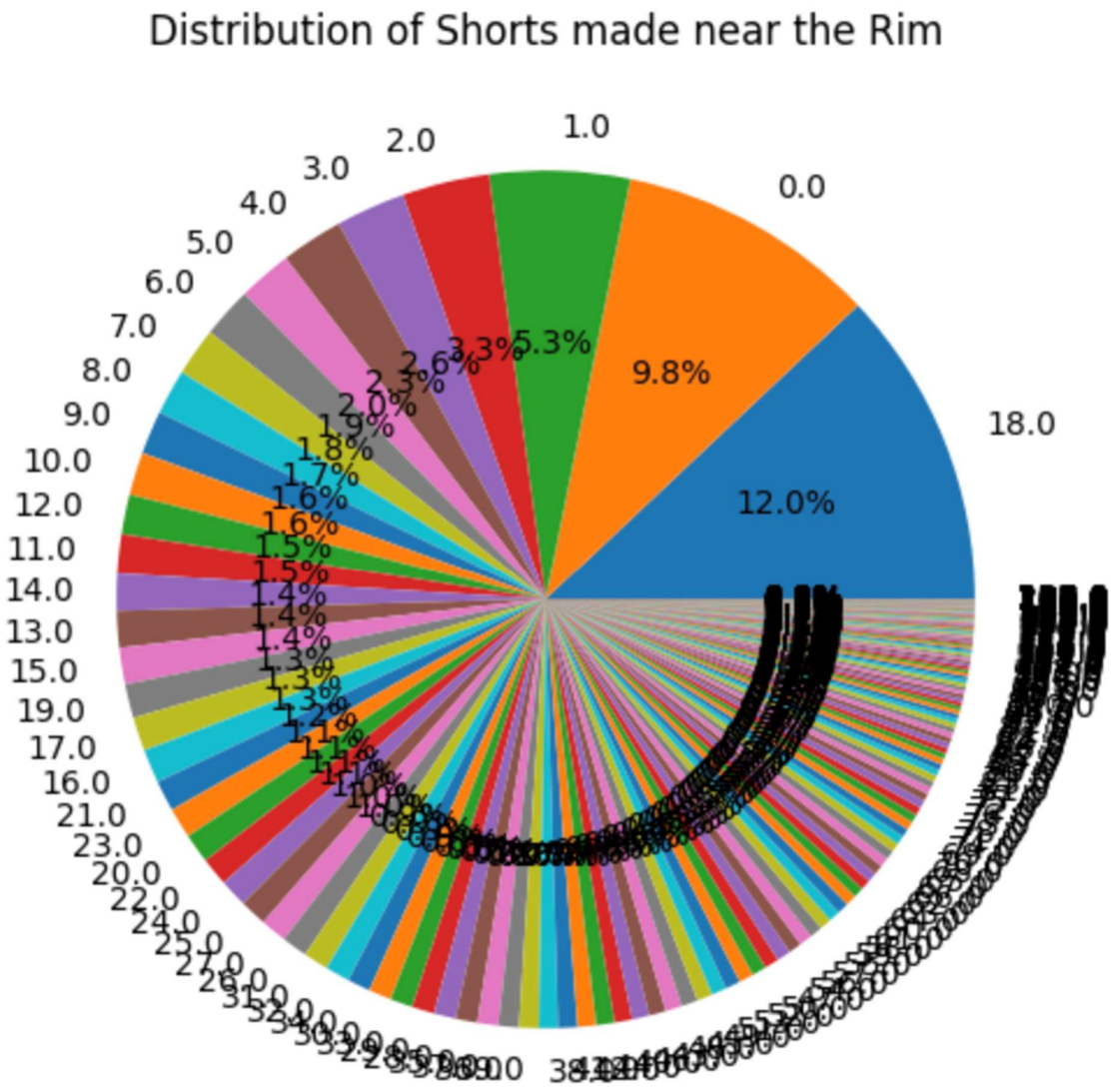


Figure: Distribution of shorts made near the Rim.

From the above pie-chart, it is evident that only 12% of players were able to play the shorts near the rim.

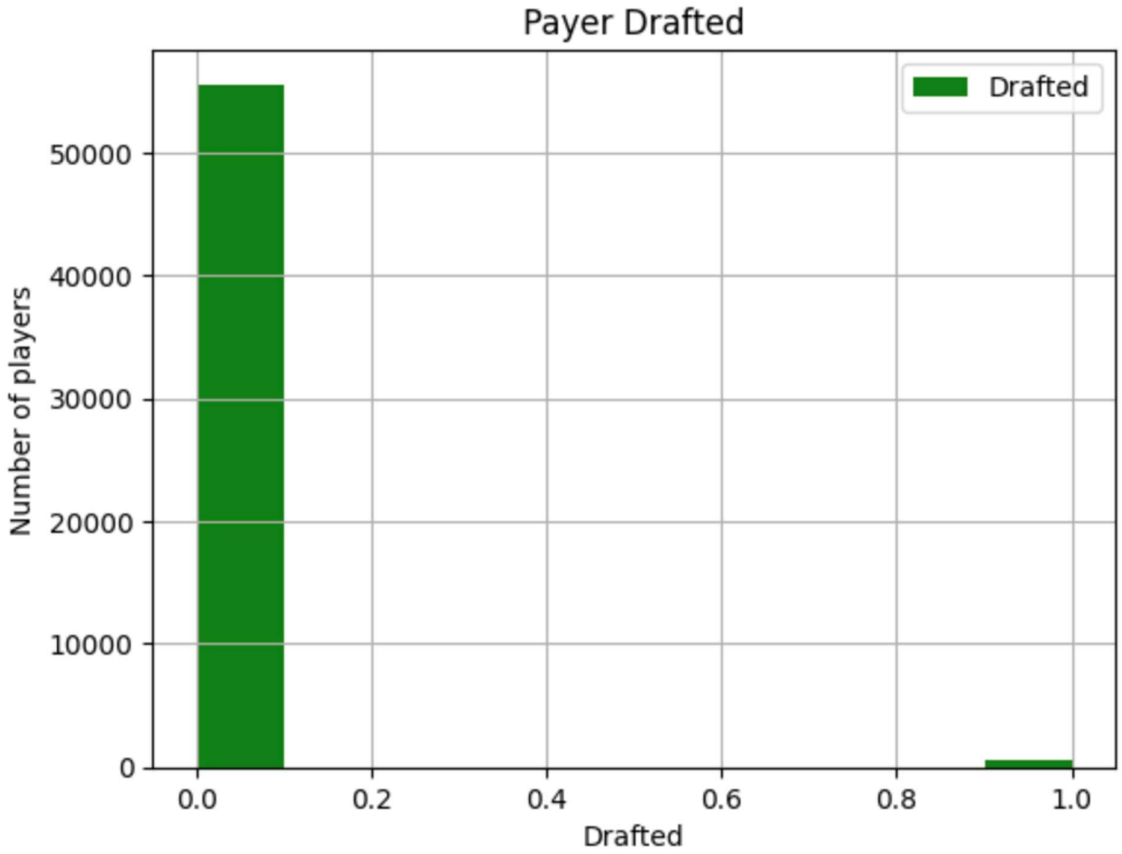


Figure: Prayer Drafted

The dataset shows the distribution of players among the two binary classes that out of 56091 players, around 1000 players were selected previously.

MODELLING

The major difference between supervised machine learning and unsupervised machine learning is that the supervised machine learning has the target variable also called as labels whereas unsupervised machine learning has only featured and there will be no feedback from the machine whether the predicted values are correct or wrong or far.

The Supervised machine learning is split into two categories known as Regressor and Classifier where in Regression the label will be in the form of continuous values and in Classifier, the label is in the form of classes.

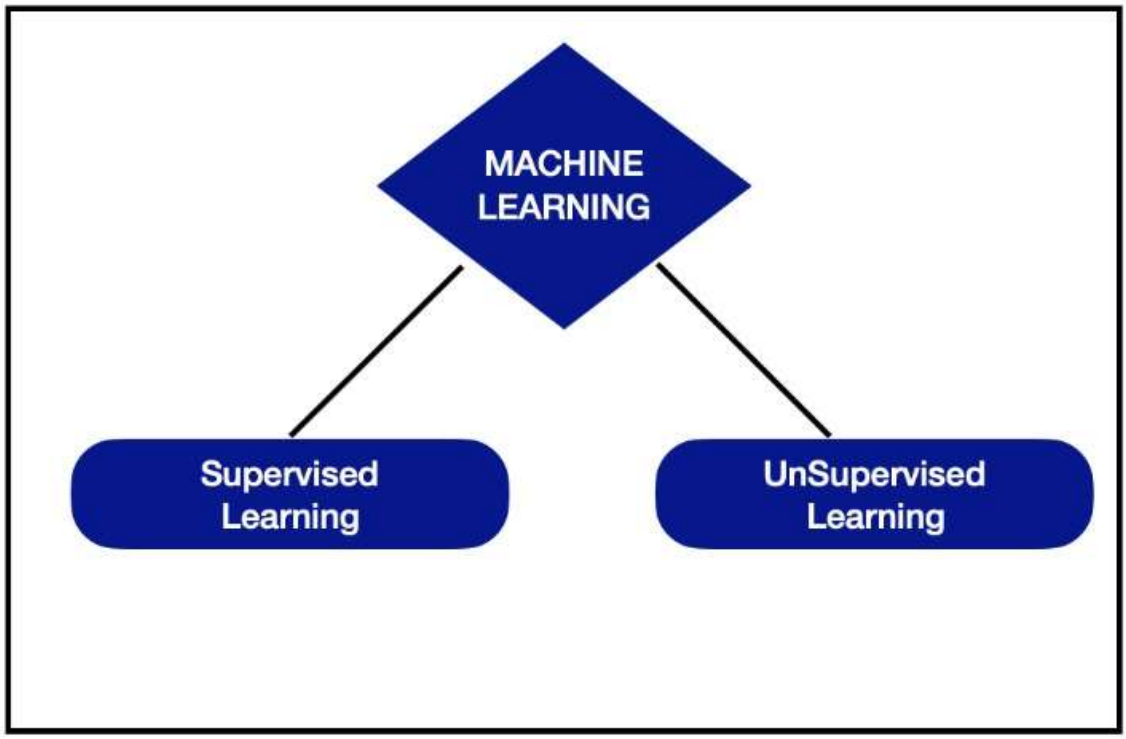


Figure: Supervised Machine Learning Types

Using Supervised machine learning binary classifier to make the predictive models for NBA event draft and Kaggle competition by implementing custom package classes and functions to clean, fit and predict the models.

Binary Classification Models:

The testing data frame is left after cleaning the data in it, to access it later to assign its player id column with the obtained training dataframe probability scores.

Baseline Model - NullBinaryClassifier Class:

The base line model is dumb model, and it is made from cleaned data of the training dataset by using the target variable. It is computed by applying the mode function on the target variable. This model is used to make comparison from the trained model which helps in determining whether the trained model beats the baseline model or not.

- **Custom Class: NullBinaryClassifier:** Using the custom class "NullBinaryClassifier", analysing the baseline model.
- **Custom Function: print_classifier_scores:**
  This custom"print_classifier_scores" function is used to print the accuracy score and f1 score of both the train set and the validation set.
- Base line score can be analysed using the accuracy score matrix. The result of the baseline model should be low compared to the training set. If the training set has a score closer to 1, whereas the baseline model has a score less than the training set, then this means the model beats the baseline model; otherwise, it is stated that the model cannot beat the baseline model.
- The baseline model score is 0.500

Model 1 - Polynomial Feature with Logistic Regression:

Polynomial Feature: The polynomial feature is imported to perform the polynomial regression on the data to make the ROC curve that covers most of the data points using square of the equation as a hyper-parameter.

Logistic Regression Model: Logistic regression is used to fit the binary classification data into the model. The hyperparameters used to train a model are:

hyperparameters: penalty= ‘elasticnet’ ,solver= 'saga',
class_weight='balanced', C= 1.0, fit_intercept=True
1. The Elastic-Net mixing parameter, with 0 <= l1_ratio <= 1
2.Setting l1_ratio=0 is equivalent to using penalty='l2'
3.Solver = Saga , it helps the algorithm to optimize the problem and it is helpful for the fast execution of large data.
4.class_weight = “balanced”.The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data.
5. C= 1. Inverse of regularization strength; it must be a positive float.
6. fit_intercept : Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.

Accuracy score: The accuracy score of the train, validation sets are around 0.970 and 0.969. This model underfits the data. This model bet the base line model with its accuracy scores. There is slight underfitting of data.

Roc_Score: The AU-Roc score obtained from the Kaggle by uploading the prediction of the model is 0.8624

This score are above the threshold level but a bit far from 1, therefore it is not considered as good score.

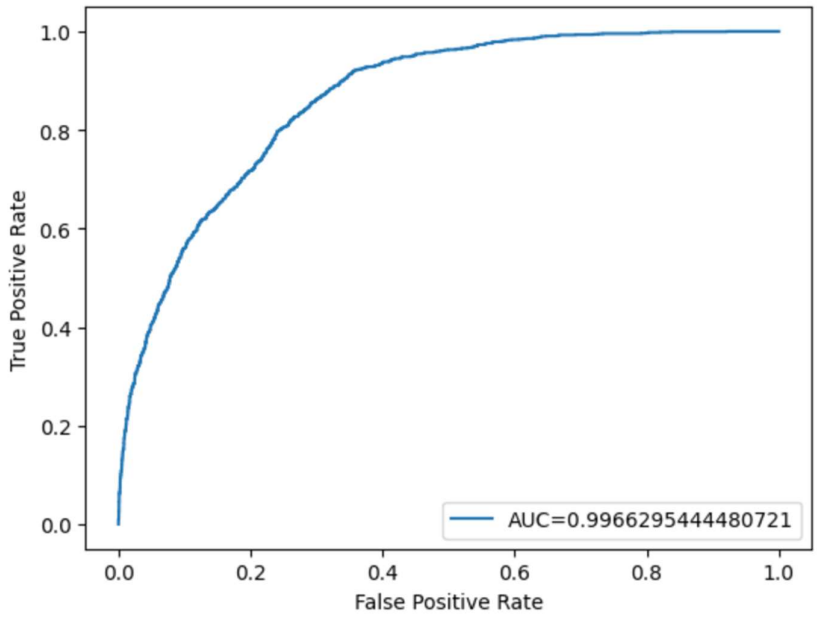


Figure: Roc curve

The resultant Roc curve of the model is 0.9966295 which is very close to 1.

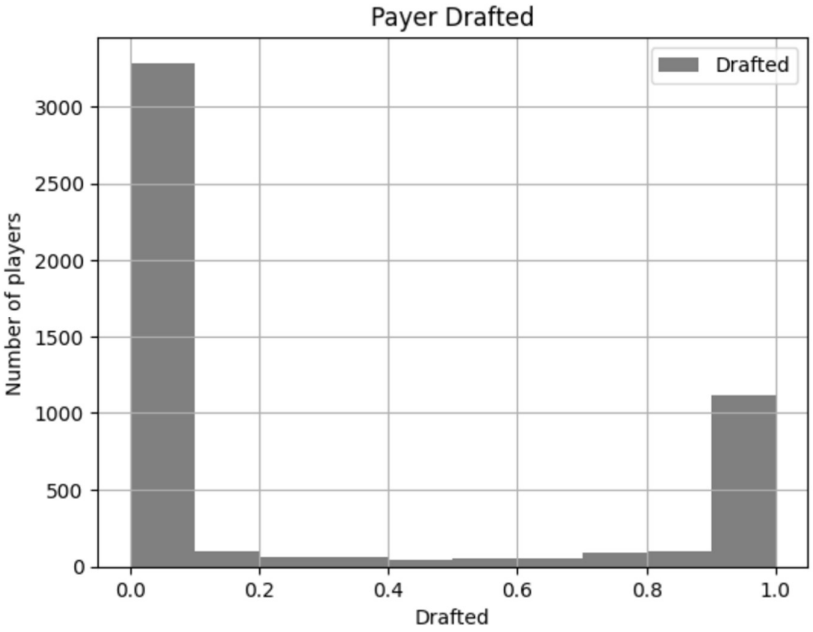


Figure: Drafted from Model 1

The figure shows the prediction of Roc curve for the test sets that in a sample of four-thousand players, around eleven-thousand players were drafted in the teams of NBA.

Model 2 – GradientBoostingClassifier:

- The gradient booster reduces errors by fitting the data into a new predictor in each iteration instead of fitting the data onto predecessors. It fits the new predictor to the residual error made by the previous predictor.
    - **The hyperparameters used in this model are:**
    - **loss = log_loss:** It refers to the binomial and multinomial deviance. This log loss helps produce better probabilistic outputs.
    - **learning_rate=0.1:** It shrinks the contribution of each tree. The learning rate ranges from 0.0 to infinity.
    - **n_estimators=100:** n estimators define the number of trees in the forest and boosting stages to perform. The large number of estimators results in better model performance and reduces the risk of overfitting.
    - **min_samples_split = 2:** This defines the minimum number of sample splits in a node. It helps control the overfitting of data. Selecting a high number of splits may lead to underfitting of the data.
    - **min_samples_leaf=1:** It defines the minimum number of samples at a leave node.
    - **min_weight_fraction_leaf=0.0:** It defines the total number of observations in fractions instead of integers.
    - **max_depth=3**: It defines the maximum depth of a tree as 3. The deeper the tree gets, the more splits it takes and the more information it captures.
    - **min_impurity_decrease=0.0:** Reducs the impurities when splitting the nodes
    - **random_state = 42**: To get the same results each time.
- The accuracy score of train and validation sets are : 0.995 and 0.9941
- The AU-Roc Score from Kaggle for predictive model is 0.9995

- The probability scores obtained showed the probability of a player's performance and chances of players being selected in teams.

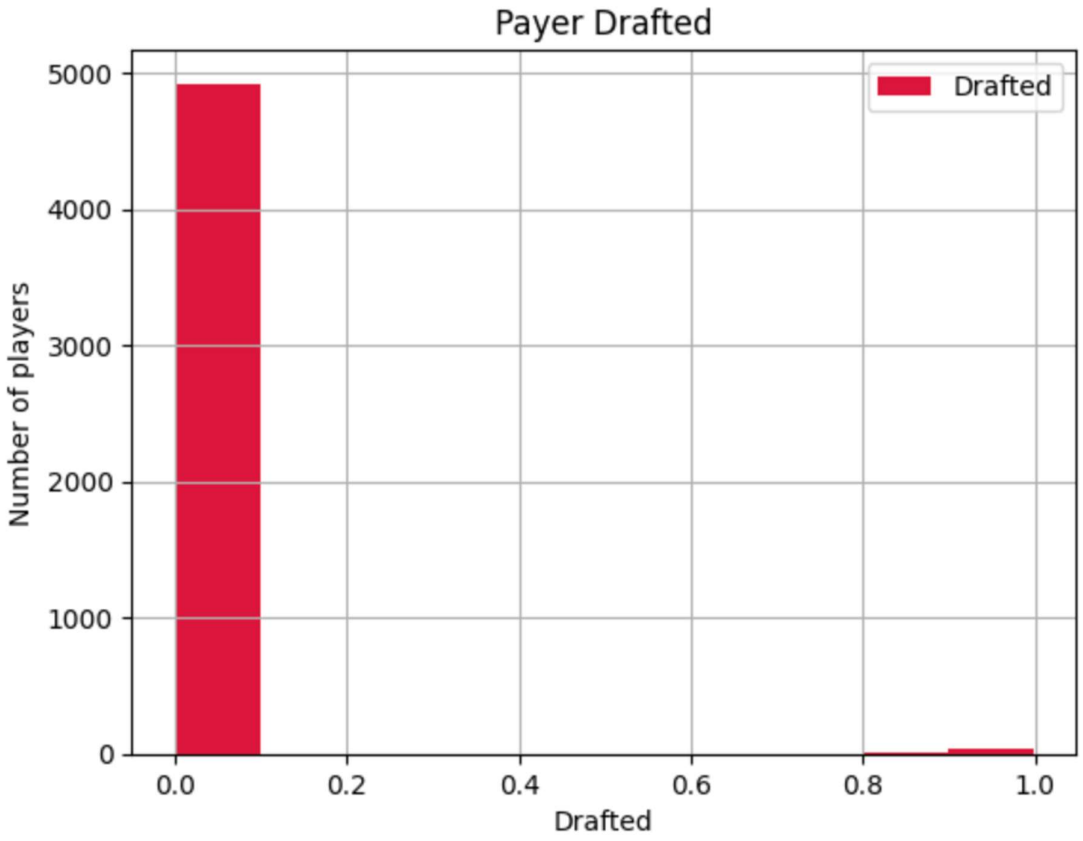


Figure: Drafted result of Model 2

output_NBA_Draft_GBC_Model_1_week2.csv
Complete · 14d ago
0.9995

The above figure shows that around 50 players were drafted from the sample of 5000 players and the Roc score obtained from Kaggle is 0.9995 which is the best score as it is closer to 1.

Model 3 - Decision Tree Classifier

- It is a tree-like structure classifier, where the internal nodes represent the features of a dataset, the branches represent the decision rules, and each leaf node represents the outcome.

- The decision node and leaf node are used to make the decision and have multiple branches, while the leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions are made based on the features of the dataset.

Important features of the decision tree are:

- **Root Node:** The root node is where the decision tree starts, and from the root node, the data is split into two or more homogeneous sets.
- **Splitting:** It is the process of dividing a node into multiple sub-nodes
- **Decision node:** This node makes the decision regarding an input feature. It does branching off of internal nodes connects them to leaf nodes or other internal nodes.
- **Leaf node:** when a sub-node does not further split into additional sub-nodes, it represents possible outcomes.
- **Pruning:** It is the process of removing unwanted branches from a tree.
- **Branch:** A subsection of the decision tree consisting of multiple nodes.

The hyperparameter used is:

- **Random state 42:** It helps in getting the same train and test sets across different executions.

Custom Function- assess_classifier_set: The custom function "assess_classifier_set" helps in performing the prediction on the training and validation sets and prints the accuracy and f1 score of the training set and validation set.

- The accuracy score of train and validation sets are: 1.0 and 0.994.
- The f1 score of the train and validation sets are: 1.0 and 0.994.
- The AU-Roc Score is 0.79523.
- Therefore, validation set is underfitting the data.

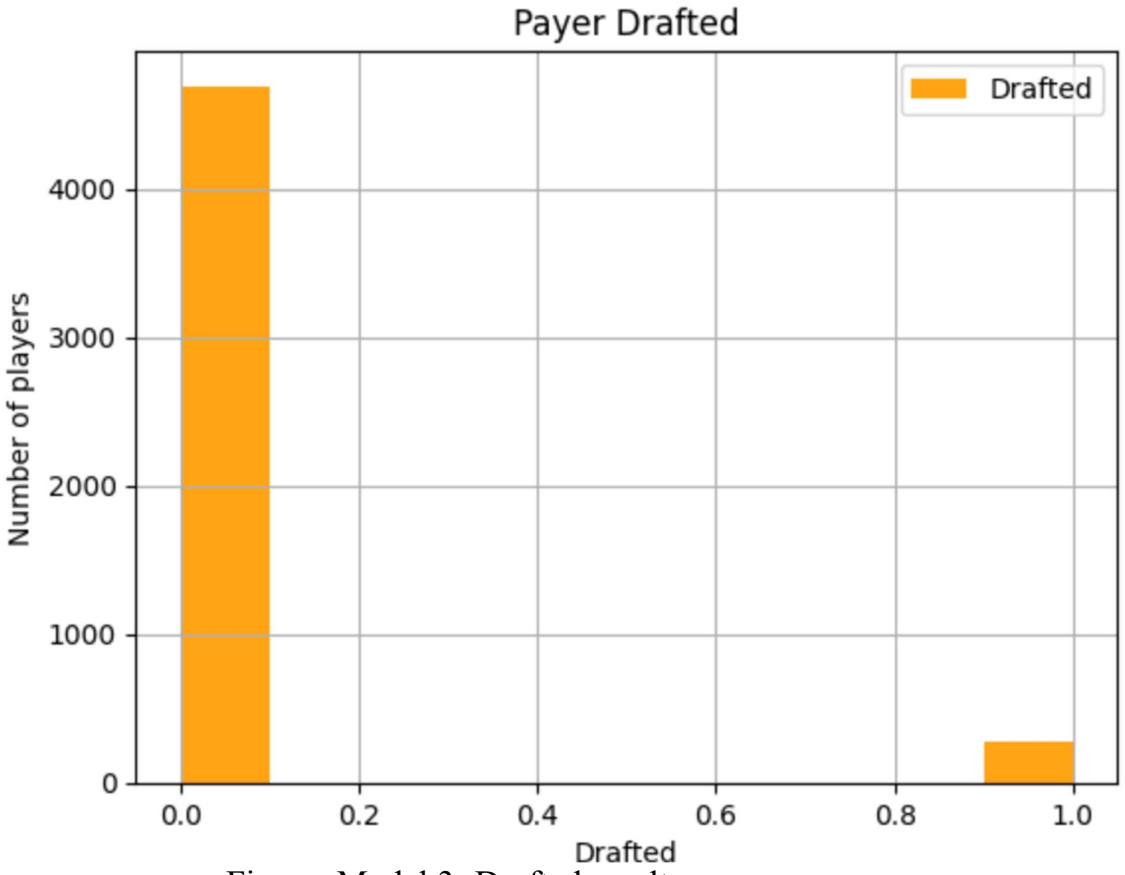
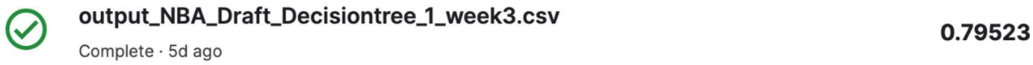


Figure: Model 3 -Drafted result



The probability result obtained from the model 3 shows in the graph that around 200 players were drafted from the sample of 5000 in the teams of NBA draft. The Au-Roc score obtained from Kaggle is 0.79523 which is not a good score and the model does not give the accurate prediction.

Model 4 – Multi Layer Perception (MLP) Classifier:

The Multi-Layer Perception (MLP) is the most common type of neural network. It is a function that maps the input to the output. It has single input, output layer and between them there can be any number of hidden layers.
The same set of neurons are present in both input layer and features, whereas hidden layer can have more than one neuron.

Every neuron is a linear function on which activation function is applied to solve complicated problems. For classification, cross-entropy is the loss function.
The output from each of the layer is given as an input to all the neurons in the subsequent layers.

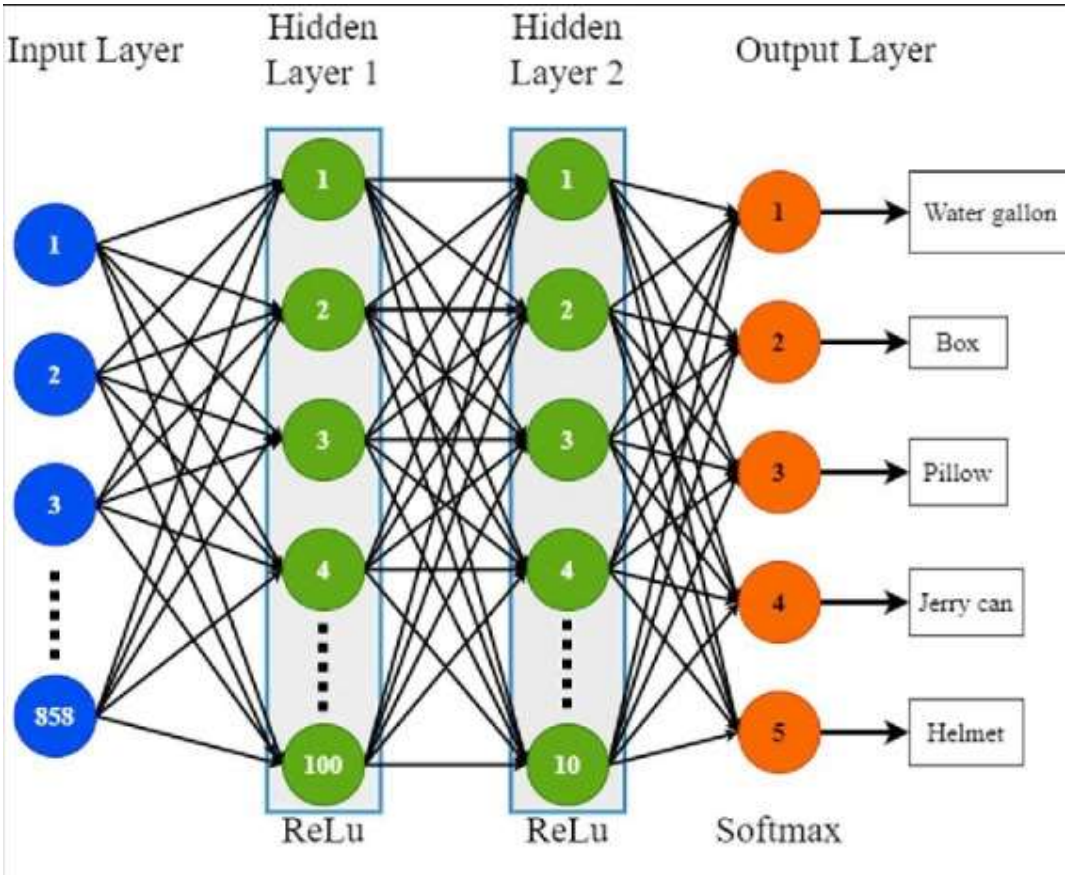


Figure : Multi layer perception classifier.

The hyperparameters used are:

Max_iter =300: It is used to give maximum iterations and assigning it to 300 iterations.
solver='lbfgs': The solver is used for weight optimization, and “ibfgs” is an optimizer in the family of quasi-Newton methods.
alpha=1e-5: It is a strength of L2 regularization term. This L2 regularization is dividing the sample size when adding to the loss.
hidden_layer_sizes=(5, 2): It represents the number of neurons in the hidden layer.
random_state=1: Assigning 1 to get same splitting sets when each time the code is run.

Multi Layer Perception Classifier:

The custom function "fit_assess_classifier()" is used to fit the data, predict the data, testing and printing accuracy and f1 score.

MODEL :

Accuracy Score:

Training Set Score: 0.891

Validation set score: 0.895

F1 score :

Training Set Score: 0.890

Validation set score: 0.894

- **The Roc score obtained from Kaggle is 0.9498.**
- The performance matrix clearly indicates that the baseline score is low compared to the training set, and therefore, we can say that the model beats the baseline model score.
- Comparing the training and validation scores, it can be concluded that the validation sets are slightly underfitting the data slightly.
- The Roc Score is **0.9498**, which means the model has accurately predicted the probability of players getting selected in the NBA Draft based on their previous performances.

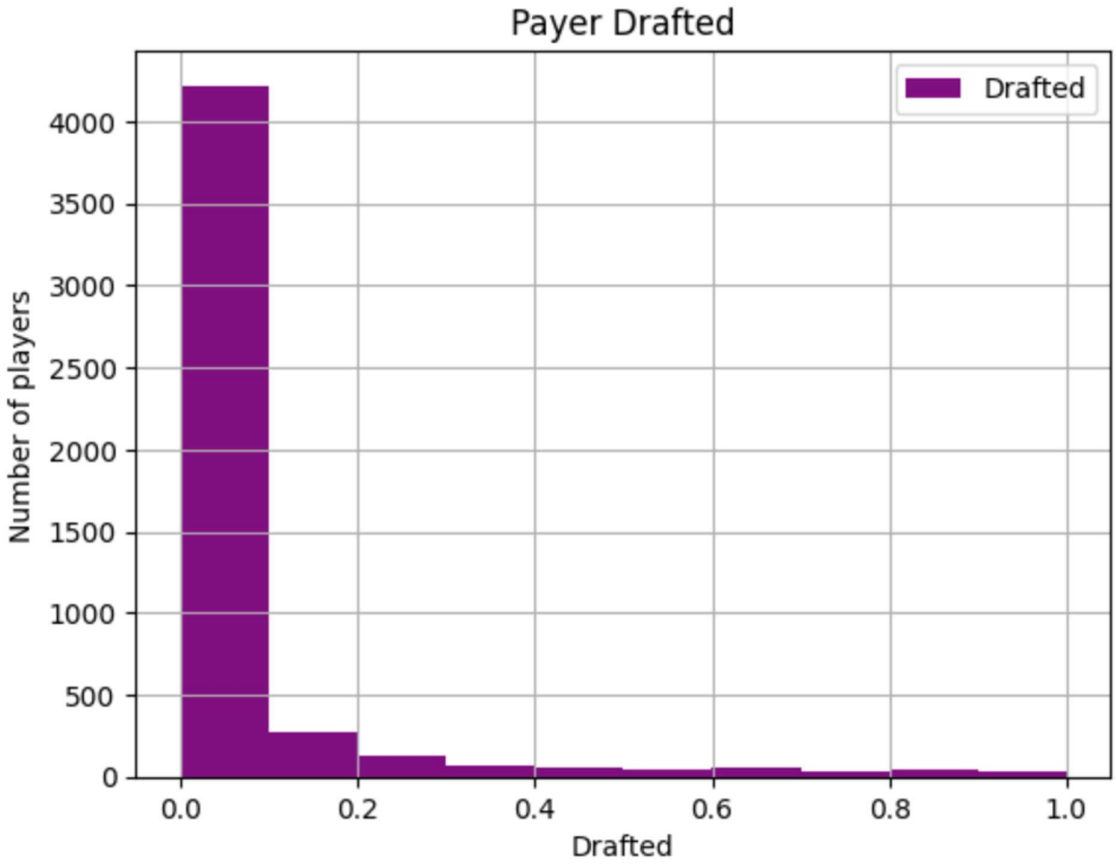


Figure: Model 4 Drafted result

DEPLOYMENT

This report includes high-level summary for the supervised binary classifier models to identify which model performs well to provide the closest prediction that can be used to help the stake holders to select the best capable players based on their previous performances. At this point of time, out of all the models there are two models that has the AU-ROC score more than 0.9% i.e., Model 2 and Model 4. Between those two models, Model 2 has the AU-ROC score 0.9995 which is almost 1 and the accuracy score of train and validation sets as 0.995 and 0.9941. Therefore, the recommended model for the deployment is Model 2.

Appendices:
List of Custom functions made:

Class : NullBinaryClassifier
Functions:

- NullBinaryClassifier class
- Drop_nan_values.
- Replace_null_with_Zero
- Median_null
- Mean_null
- Drop_target
- Random_split_sets
- Save_sets
- Load_sets
- Fit_assess_classifer
- Assess_classifier_set
- Print_classifier_scores

NBA-Draft Abbreviations full form

feature	name	description
1	team	Name of team
2	conf	Name of conference
3	GP	Games played
4	Min_per	Player's percentage of available team minutes played
5	ORtg	ORtg - Offensive Rating
6	usg	Usg% - Usage Percentage
7	eFG	eFG% - Effective Field Goal Percentage
8	TS_per	TS% - True Shooting Percentage
9	ORB_per	ORB% - Offensive Rebound Percentage
10	DRB_per	DRB% - Defensive Rebound Percentage
11	AST_per	AST% - Assist Percentage
12	TO_per	TOV% - Turnover Percentage
13	FTM	Free Throws
14	FTA	Free Throw Attempts
15	FT_per	Free Throw Percentage; the formula is FTM / FTA.
16	twoPM	2P - 2-Point Field Goals
17	twoPA	2PA - 2-Point Field Goal Attempts
18	twoP_per	2P% - 2-Point Field Goal Percentage; the formula is 2P / 2PA.
19	TPM	3P - 3-Point Field Goals
20	TPA	3PA - 3-Point Field Goal Attempts
21	TP_per	3P% - 3-Point Field Goal Percentage
22	blk_per	BLK% - Block Percentage
23	stl_per	STL% - Steal Percentage
27	ftr	Frequency throw rato
28	yr	Studentâ€™s year of study
29	ht	Height of student

30	num	Player's number
31	porpag	Points Over Replacement Per Adjusted Game
32	adjoe	AdjO â€“ Adjusted offensive efficiency
33	SSS	Percent of Team's Personal Fouls
34	year	Season's year
35	type	Type of metrics displayed
36	Rec_Rank	Recruiting rank i.e. what the player was ranked as a recruit coming out of high school
37	ast_tov	Ratio Assists against Turnovers
38	rimmade	Shots made at or near the rim
39	rimmade_	Sum of Shots made at or near the rim and Shots missed
40	midmade	Two point shots that were not made at or near the rim
41	midmade_	Sum of Two point shots that were not made at or near the rim and Shots missed
42	rim_ratio	Ratio between Shots made at or near the rim against Shots missed
43	mid_ratio	Ratio between Two point shots that were not made at or near the rim and Shots missed
44	dunksmad	Dunks made
45	dunksmiss	Sum of Dunks made and Dunks missed
46	dunks_rati	Ratio between Dunks made and Dunks missed
47	pick	Order of NBA draft
48	drtg	DRtg - Defensive Rating
49	adrtg	Adjusted DRtg
50	dporpag	Asdjusted porpag
54	stops	Stops - Stops
55	bpm	BPM - Estimate the playerâ€™s contribution in points above league average per 100 possessions played
56	obpm	Offensive BPM
57	dbpm	Defensive BPM
58	gbpm	BPM 2.0
59	mp	MP - Minutes Played (available since the 1951-52 season)
60	ogbpm	Offensive BPM 2.0
61	dgbpm	Defensive BPM 2.0
62	oreb	ORB - Offensive Rebounds (available since the 1973-74 season in the NBA)
63	dreb	DRB - Defensive Rebounds (available since the 1973-74 season in the NBA)
64	treb	TRB - Total Rebounds (available since the 1950-51 season)
65	ast	AST - Assists
66	stl	STL - Steals (available since the 1973-74 season in the NBA)
67	blk	BLK - Blocks (available since the 1973-74 season in the NBA)
68	pts	PTS - Points
69	player_id	Unique identifier of player
70	drafted	Target - Was the player drafted at the end of the season

REFERENCES:

- *Sklearn.tree.decisiontreeclassifier*. scikit. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
- *1.17. neural network models (supervised)*. scikit. (n.d.-a). https://scikit-learn.org/stable/modules/neural_networks_supervised.html
- *Sklearn.preprocessing.PolynomialFeatures*. scikit. (n.d.-b). https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html
- *Sklearn.linear_model.logisticregression*. scikit. (n.d.-b). https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
- (N.d.). https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html.
- (N.d.). https://heartbeat.comet.ml/resampling-to-properly-handle-imbalanced-datasets-in-machine-learning-64d82c16ceaa.

GitHubLink:

https://github.com/MAHJABEENMOHIUDDIN/Adv_ML_Project_Community_Prediction_Competition/tree/master