36120 Advanced Machine Learning Application

Spring 2023

Assessment Task- 1

Submission Date: 8-Sept-2023

UTS Faculty of Transdisciplinary Innovation

Submitted By:

Saumya Bhutani

14360820

Table of Contents

CRISP-DM Methodology

1. Business understanding

The goal of the project – The primary aim is to assist stakeholders in the basketball ecosystem, including NBA teams, scouts, player agents, analysts, fans, and media, by providing a data-driven prediction of whether a college basketball player will be drafted into the NBA. This model is designed to forecast whether a college basketball player will be selected in the NBA draft, utilizing advanced machine learning classification techniques.

Value - Accurate predictions can have several positive outcomes, including efficient resource allocation for scouting, informed player evaluations, enhanced fan engagement, and strategic draft decisions. It also supports player agents in advising their clients effectively.

Leveraging machine learning techniques, helps NBA teams identify promising talent early, save scouting expenses, optimise player selection, and make data-driven decisions to strengthen their rosters.

Assessing the situation

The project relies on a range of player-related features such as player statistics, performance data, physical attributes, and historical trends. By analyzing college basketball player data from the past records, we aim to identify patterns and factors that influence a player's likelihood of being drafted into the NBA. in the basketball context, we suspect that player statistics, performance metrics, and physical attributes may play a crucial role in predicting whether a college basketball player will make it to the NBA.

For instance, we believe that a player's scoring efficiency (eFG and TS_per), their ability to contribute in various aspects of the game (AST_per, DRB_per, STL_per), and their physical attributes (height) might strongly influence their chances of getting drafted.

Determining the data science project goals

Employing classification modelling techniques to predict whether college basketball players will be drafted into the NBA. Evaluating the model's effectiveness by assessing AU ROC scores on the validation datasets.

Various tools and technologies to be used:

- Programming language - Python
- Libraries – numpy, pandas, matplotlib, seaborne, altair, scikit learn

2. Data understanding

Two datasets have been provided: one for training and the other for testing. The training dataset comprises 56,091 records and includes 64 features. Some categorical features are present, but a few features have incorrect data types. Most of the data consists of numeric values, with the majority being floats and the rest integers.

Below is the screenshot of the Training dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 56091 entries, 0 to 56090
Data columns (total 64 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   team                 56091 non-null  object
 1   conf                 56091 non-null  object
 2   GP                   56091 non-null  int64
 3   Min_per              56091 non-null  float64
 4   Ortg                 56091 non-null  float64
 5   usg                  56091 non-null  float64
 6   eFG                  56091 non-null  float64
 7   TS_per               56091 non-null  float64
 8   ORB_per              56091 non-null  float64
 9   DRB_per              56091 non-null  float64
 10  AST_per              56091 non-null  float64
 11  TO_per               56091 non-null  float64
 12  FTM                  56091 non-null  int64
 13  FTA                  56091 non-null  int64
 14  FT_per               56091 non-null  float64
 15  twoPM                56091 non-null  int64
 16  twoPA                56091 non-null  int64
 17  twoP_per             56091 non-null  float64
 18  TPM                  56091 non-null  int64
 19  TPA                  56091 non-null  int64
 20  TP_per               56091 non-null  float64
 21  blk_per              56091 non-null  float64
 22  stl_per              56091 non-null  float64
 23  ftr                  56091 non-null  float64
 24  yr                   55817 non-null  object
 25  ht                   56011 non-null  object
 26  num                  51422 non-null  object
 27  porpag               56091 non-null  float64
 28  adjoe                56091 non-null  float64
 29  pfr                  56091 non-null  float64
 30  year                 56091 non-null  int64
 31  type                 56091 non-null  object
 32  Rec_Rank             17036 non-null  float64
 33  ast_tov              51901 non-null  float64
 34  rimmade              50010 non-null  float64
 35  rimmade_rimmiss      50010 non-null  float64
 36  midmade              50010 non-null  float64
 37  midmade_midmiss      50010 non-null  float64
 38  rim_ratio            46627 non-null  float64
 39  mid_ratio            46403 non-null  float64
 40  dunksmade            50010 non-null  float64
 41  dunksmiss_dunksmade  50010 non-null  float64
 42  dunks_ratio          25298 non-null  float64
 43  pick                 1386 non-null   float64
 44  drtg                 56047 non-null  float64
 45  adrtg                56047 non-null  float64
 46  dporpag              56047 non-null  float64
 47  stops                56047 non-null  float64
 48  bpm                  56047 non-null  float64
 49  obpm                 56047 non-null  float64
 50  dbpm                 56047 non-null  float64
 51  gbpm                 56047 non-null  float64
 52  mp                   56053 non-null  float64
 53  ogbpm                56047 non-null  float64
 54  dgbpm                56047 non-null  float64
 55  oreb                 56053 non-null  float64
 56  dreb                 56053 non-null  float64
 57  treb                 56053 non-null  float64
 58  ast                  56053 non-null  float64
 59  stl                  56053 non-null  float64
 60  blk                  56053 non-null  float64
 61  pts                  56053 non-null  float64
 62  player_id            56091 non-null  object
 63  drafted              56091 non-null  float64
dtypes: float64(49), int64(8), object(7)
```

The datasets contains a lot of null values.

```
#heatmap highlighing null values in the dataframe
sns.heatmap(df_train.isnull(), yticklabels = False)
```

```
<Axes: >
```

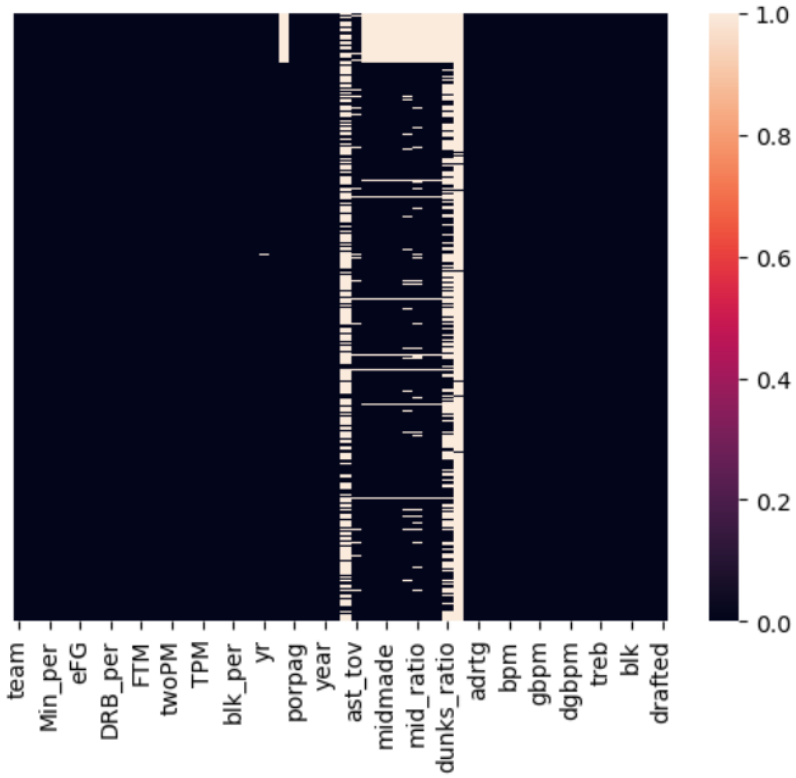

Figure 1: Heatmap shows null values in the training dataset.

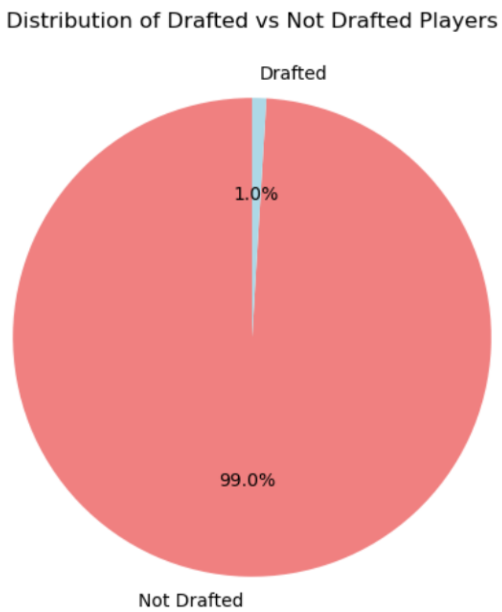

Figure 2: Pie chart shows class imbalance in the training dataset.

.

Experiment 1: XGBoost

Hypothesis:

The XGBoost model can effectively predict the players to drafted in NBA(target) by analysing various independent features such as player performance metrics, technicalities of player's game

Experiment 2: AdaBoost

Hypothesis:

The AdaBoost model can effectively predict the players to drafted in NBA(target) by analysing various independent features such as player performance metrics, technicalities of player's game etc.

Experiment 3: XGBoost with Hyperparameter Tuning (including L2 regularization)

Hypothesis:

This experiment aims to enhance the XGBoost model's predictive capabilities through hyperparameter tuning, including L2 regularization. The expectation is improved accuracy in forecasting NBA draft selections (target).

Experiment 4: Gradient Boost

Hypothesis:

The Gradient Boost model's effectiveness in predicting NBA draft picks (target) is expected due to its inherent strengths in ensemble learning and its ability to leverage player performance metrics and technical aspects.

Considering these hypotheses is worthwhile because it can bring data-driven insights to the NBA draft process, benefiting teams, players, scouts, and fans. It aligns with the industry's increasing reliance on analytics and addresses the complexities of selecting and predicting the success of future NBA players. The insights gained from this project can contribute to more informed decisions and a deeper understanding of the factors influencing the transition to professional basketball.

3. Data Preparation

For the 1st two Experiments, I performed the same data preparation strategy,

<u>Handling null values</u>

For the first three experiments, I followed a consistent data preparation strategy.

I encountered two features, 'pick' and 'Rec_Rank', which had a high percentage of missing values, with over 97% and 69% missing, respectively. Consequently, I decided to exclude these features from my analysis.

Additionally, I observed missing values in the following columns: 'ast_tov', 'rimmade', 'rimmade_rimmiss', 'mid_made', 'midmade_midmiss', 'dunksmade', 'dunksmiss_dunksmade', 'drtg', 'adrtg', 'dporpag', 'stops', 'bpm', 'obpm', 'dbpm', 'gbpm', 'mp', 'ogbpm', 'dgbpm', 'oreb', 'dreb', 'treb', 'ast', 'stl', 'blk', and 'pts'. To address this, I imputed the missing values in these columns with their respective mean values.

The missing values in columns containing ratios were imputed using the following formulas:

rim_ratio was calculated as rimmade / rimmade_rimmiss

mid_ratio was calculated as midmade / midmade_midmiss

dunks_ratio was calculated as dunksmade / dunksmiss_dunksmade

The remaining missing values in these ratio columns were replaced with their respective mean values.

The columns 'yr' and 'num' exhibit significant irregularities, rendering the data unreliable. Therefore, I have chosen to eliminate these columns.

In the 3rd experiment, I followed the below strategy:

I handled the 'ht' column (which is described in 4th experiment)

I used mean values to make the imputations.

Removed columns: 'team', 'conf', 'pick', 'Rec_Rank', 'num' and 'yr'.

In the 4th experiment, I followed the below strategy:

The missing values in columns containing ratios were imputed using the following formulas:

rim_ratio was calculated as rimmade / rimmade_rimmiss

mid_ratio was calculated as midmade / midmade_midmiss

dunks_ratio was calculated as dunksmade / dunksmiss_dunksmade

I conducted a skewness analysis for columns with missing values to determine their distribution characteristics. Based on the skewness levels observed, I applied the following imputation strategies:

- For columns with low skewness, I imputed missing values using the median.

- For columns with moderate skewness, I utilized the Yeo-Johnson transformation with the mean.
- For columns with high skewness, I applied the Yeo-Johnson transformation with the median for imputation.

Skewness	Features	Stategy
Low	Rec_Rank, rim_ratio, mid_ratio, dunks_ratio, pick, mp, ogbpm, dgbpm, oreb, dreb, treb, ast, stl, blk, pts, ht	Median
Moderate	Ast_tov, rimmade, rimmade_rimmiss, midmade, midmade_midmiss	Yeo-Johnson transformation with the mean
High	dunksmade, dunksmiss_dunksmad, drtg, adrtg, dporpag, stops, bpm, obpm, dbpm, gbpm	Yeo-Johnson transformation with the median

The decision to use the Yeo-Johnson transformation for imputations is based on its versatility, robustness, and adaptability to different skewness types. It offers greater flexibility by accommodating positive and negative skewness adjustments while working towards improved data normalization.

Removing features

I decided not to consider features such as team, and conf as they had many categories. The columns 'yr' and 'num' contain many irregularities. Since the data is unreliable. Thus I have decided to remove these columns.

Converting columns to suitable dtypes

I did not include the 'ht' column in the first two experiments in the modeling process. However, I addressed the 'ht' column in the third and fourth experiments. Initially, the data type was set as 'object,' the values were in date format, such as '2-June-2023.' I transformed them into integer data types representing feet and inches. For example, '2/06/2023' was converted to '2062023' and then '6022023' (M/d/Y format). Finally, I removed the year, resulting in '6.02,' which represents 6 feet 2 inches.

Data splitting

I divided the training data into two sets, training and validation, using an 80:20 ratio, since a separate test dataset was already provided. Therefore, I had a total of three different datasets, each with the following number of records:

Dataset	Records
Training	44872
Validation	11219
Testing	4970

Resampling of Training data

```
Class 0: 55555
Class 1: 536
Proportion: 103.65 : 1
```

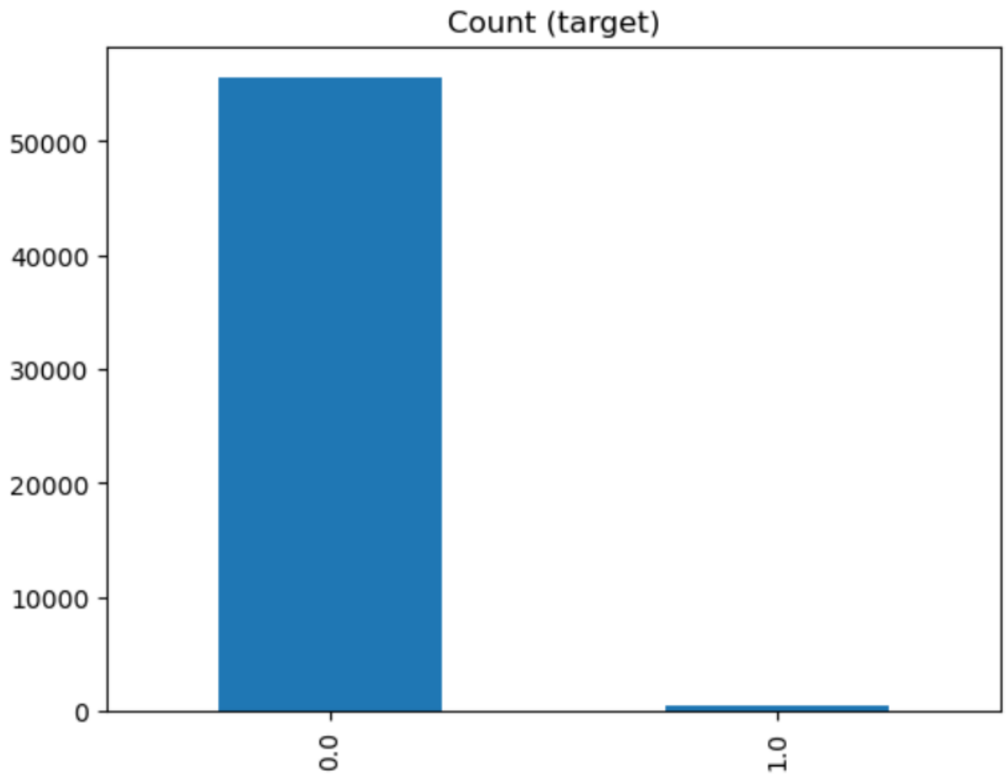

Figure 3: Barplot shows that the dataset is a highly imbalanced 'Target' feature in the Training set.

The target variable exhibits class imbalance, where Class 1 represents drafted players and Class 0 represents not drafted players. Resampling is crucial due to the significant class imbalance to prevent bias towards the majority class and improve prediction accuracy for the minority class.

For Resampling the Minority class, I have performed oversampling of minority class (1) using SMOTE(Synthetic Minority Over-sampling Technique) in the training dataset only.

Feature Engineering

For all the experiments, I performed the following feature engineering steps:

Dropped features: yr, num, 'team', 'conf'

Features Dropped	Reason
'team', 'conf'	Contains many categories
yr, num	Contains a lot of irregularities. The data is unreliable.

Feature Scaling

I applied feature scaling to all independent features in the training, validation, and testing sets. This step ensures that each feature contributes to the model fairly, enhances algorithm performance, and promotes effective model convergence. Feature scaling helps eliminate any biases introduced by varying feature scales, leading to more accurate predictions.

Data Distribution

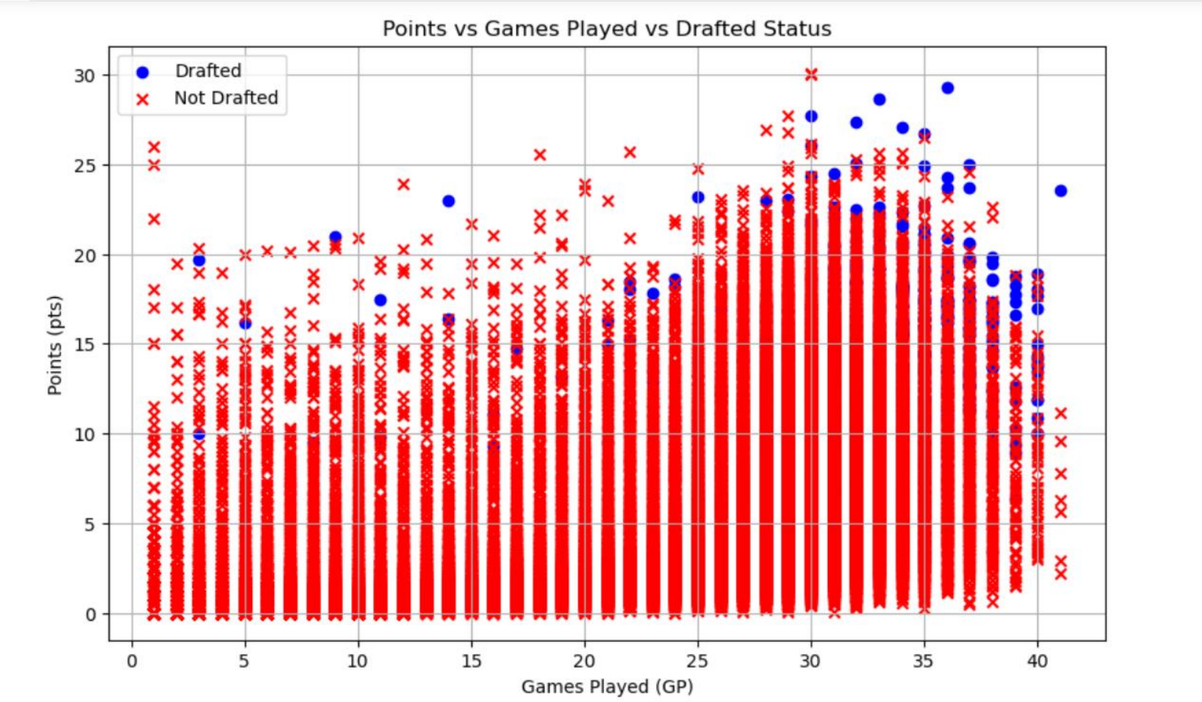

Figure 4: Scatterplot Showing the Distribution of Points vs. Games Played vs. Drafted Status of Players in the Training Set.

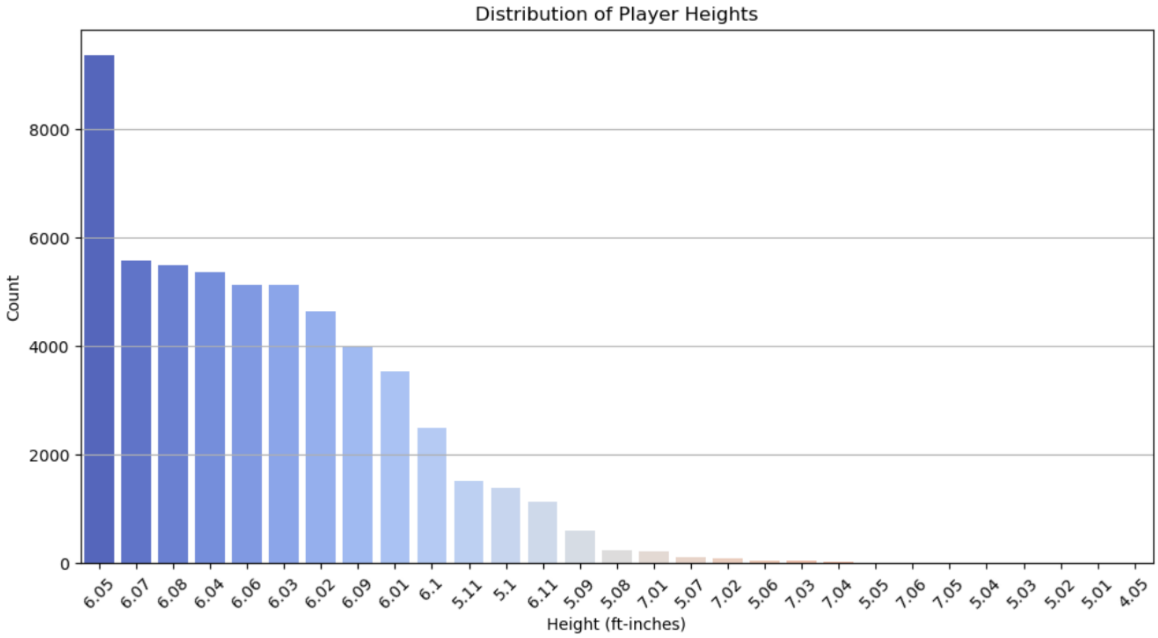

Figure 5: Barplot Showing the Distribution of height of Players in the Training Set.

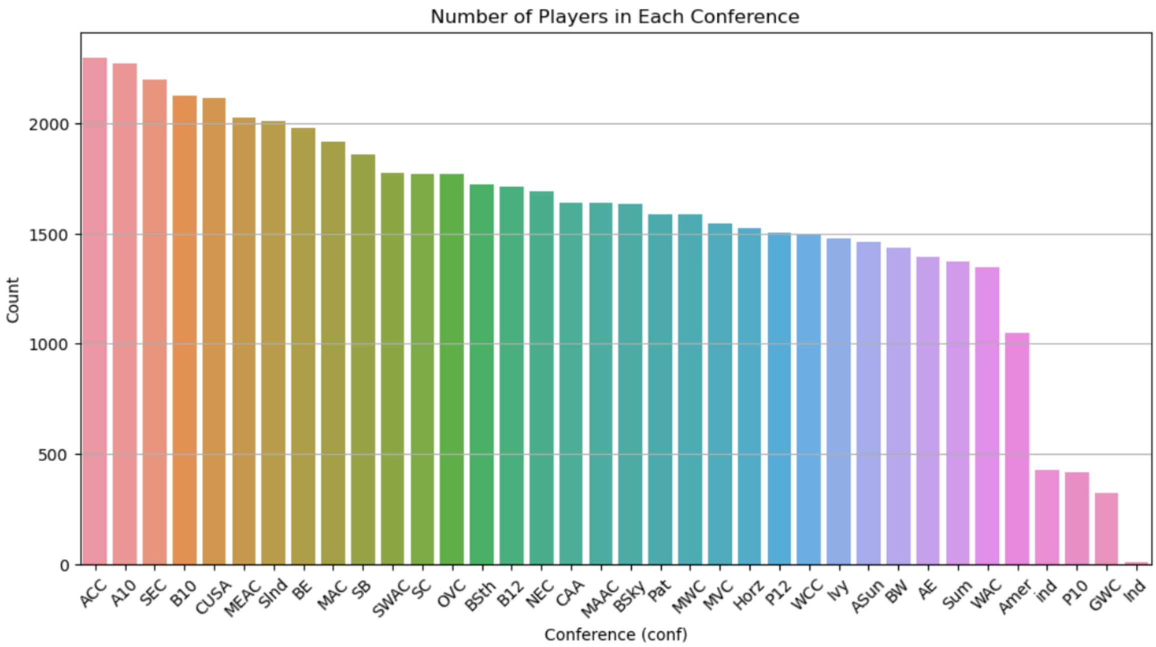

Figure 6: Barplot Displaying Player Counts Across Different Conferences in the Training Set.

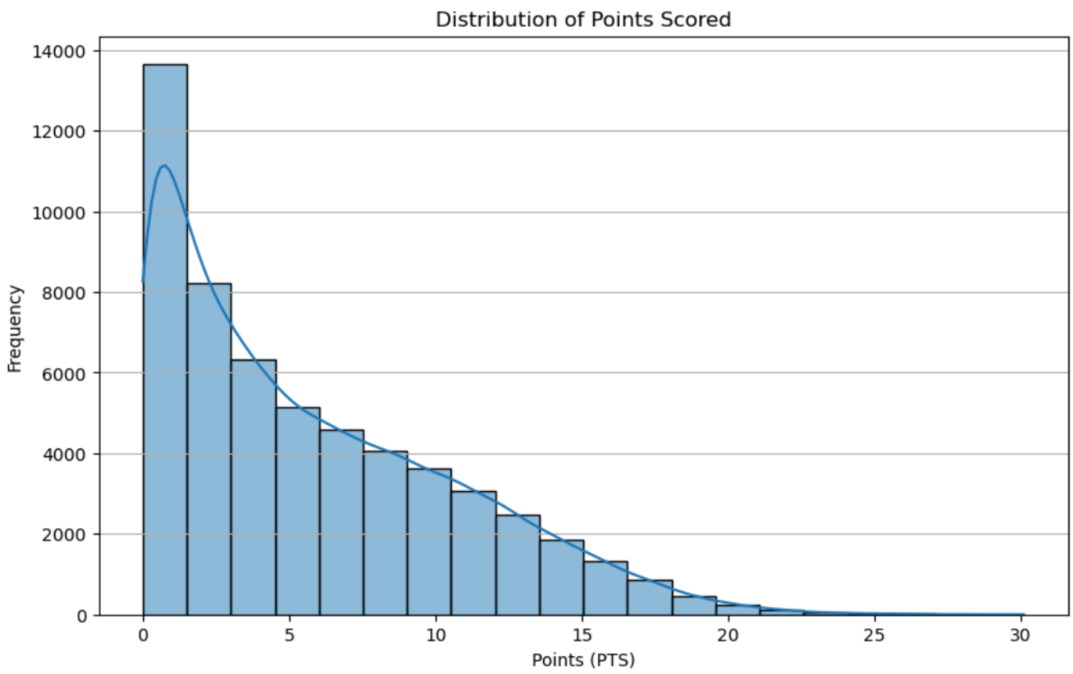

Figure 7: Distribution of points scored in the Training Set.

4. Modelling

In Experiments 1 and 2, I have selected the following features for analysis based on their relevance to the task. These features encompass player performance metrics, technical aspects of the player, and draft-related attributes, making them suitable for the analysis:

Selected X_features
'Min_per', 'usg', 'eFG', 'TS_per', 'ORB_per', 'DRB_per', 'AST_per', 'FT_per', 'twoP_per', 'TP_per', 'porpag', 'ast_tov', 'rim_ratio', 'mid_ratio', 'dunks_ratio', 'drtg', 'dporpag', 'stops', 'bpm', 'obpm', 'dbpm', 'gbpm', 'ogbpm', 'dgbpm', 'oreb', 'dreb', 'treb', 'ast', 'stl', 'blk', 'pts'

These features have been carefully chosen to capture various aspects of a player's performance and characteristics relevant to the analysis and prediction of their NBA draft prospects.

Experiment 1: Trained XGBoost classifier as per the hypothesis generated.

Experiment 2: Trained AdaBoost classifier as per the hypothesis generated.

Experiment 3: Trained the XGBoost classifier with automated hyperparameterization as per the hypothesis generated. (added features by using feature engineering techniques)

In this experiment, I used a correlation matrix to select features with a strong correlation to the target variable, 'drafted.' The selected features include player performance metrics, technical attributes, and draft-related characteristics. These features were chosen based on their correlation with the target variable in the analysis.

Selected X_features
'porpag', 'dunksmade', 'dunksmiss_dunksmade', 'dporpag', 'twoPM', 'FTM', 'FTA', 'twoPA', 'midmade', 'pts', 'midmade_midmiss', 'rimmade', 'stops', 'dreb', 'treb', 'rimmade_rimmiss', 'blk', 'mp', 'Min_per', 'stl', 'oreb', 'bpm', 'ast', 'gbpm', 'obpm', 'TPM', 'ogbpm', 'adjoe', 'TPA', 'usg', 'GP', 'dgbpm', 'dunks_ratio', 'dbpm', 'Ortg', 'FT_per', 'TS_per', 'rim_ratio', 'AST_per', 'eFG', 'twoP_per', 'TP_per', 'mid_ratio', 'ast_tov', 'DRB_per', 'blk_per', 'stl_per', 'ORB_per', 'ftr', 'ht', 'year', 'pfr', 'TO_per', 'drtg', 'adrtg'

Experiment 4: Trained the Gradient Boosting classifier as per the hypothesis generated.
In this experiment, I selected the top 56 features that showed a high correlation with the target variable and used a random forest model to perform feature importance analysis. The final set of features was carefully chosen by combining the results of both methods.

Selected X_features
'Min_per', 'Ortg', 'usg', 'eFG', 'TS_per', 'TO_per', 'AST_per', 'FTM', 'FTA', 'FT_per', 'twoPM', 'twoPA', 'twoP_per', 'TPA', 'TP_per', 'blk_per', 'stl_per', 'ftr', 'porpag', 'adjoe', 'pfr', 'Rec_Rank', 'ast_tov', 'rimmade', 'rimmade_rimmiss', 'midmade', 'midmade_midmiss', 'rim_ratio', 'mid_ratio', 'dunksmade', 'dunksmiss_dunksmade', 'dunks_ratio', 'drtg', 'adrtg', 'dporpag', 'stops', 'bpm', 'obpm', 'dbpm', 'gbpm', 'mp', 'ogbpm', 'dgbpm', 'oreb', 'dreb', 'treb', 'ast', 'stl', 'blk', 'pts', 'ht', 'pick'

By combining these methods, I validated the initial correlation findings, identified redundant features to improve model performance, and reduced computational overhead.
Below is the Feature Importance ranking generated by Random Forest:

Top Important Features (Ranked):

	Feature	Importance
0	dporpag	0.134251
1	porpag	0.109485
2	gbpm	0.084444
3	bpm	0.071579
4	Rec_Rank	0.067395
5	ogbpm	0.061088
6	adjoe	0.050939
7	twoPM	0.050271
8	stops	0.041205
9	obpm	0.038851
10	pts	0.031936
11	twoPA	0.025053
12	FTM	0.023215
13	FTA	0.021079
14	adrtg	0.015304
15	dunksmade	0.012898
16	mp	0.011743
17	dunksmiss_dunksmade	0.009514
18	treb	0.008300
19	midmade	0.007207
20	Ortg	0.007003
21	rimmade	0.006907
22	rimmade_rimmiss	0.006788
23	blk_per	0.006433
24	dgbpm	0.006382
25	drtg	0.005831
26	dreb	0.005697
27	dunks_ratio	0.005681
28	usg	0.004629
29	blk	0.003862
30	midmade_midmiss	0.003702
31	ht	0.003465
32	TO_per	0.003199
33	dbpm	0.003175
34	stl_per	0.002919
35	TP_per	0.002867
36	eFG	0.002851
37	mid_ratio	0.002809
38	twoP_per	0.002636
39	TPA	0.002619
40	oreb	0.002536
41	pfr	0.002495
42	ftr	0.002412
43	TS_per	0.002380
44	ast_tov	0.002278
45	stl	0.002241
46	ast	0.002206
47	rim_ratio	0.002161
48	year	0.002159
49	TPM	0.002149
50	AST_per	0.002116
51	FT_per	0.002110
52	GP	0.002009
53	ORB_per	0.001869
54	Min_per	0.001867
55	DRB_per	0.001799

Hyperparameters:

Each model underwent hyperparameter tuning using grid search to find the best settings. Grid search systematically explored hyperparameter ranges to optimize the AU ROC score, a key performance metric in each experiment.

- For Experiment 1: (XGBoost), colsample_bytree=0.8, learning_rate=0.1, max_depth=4, min_child_weight=1, n_estimators=200, subsample=0.8
- For Experiment 2 (AdaBoost), learning_rate = 0.5, n_estimators=200, random_state=43
- For Experiment 3 (XGBoost with regularization), colsample_bytree=0.8, learning_rate=0.01, max_depth=4, min_child_weight=1, n_estimators=400, subsample=0.8, reg_lambda = 0.5, random_state=42
- For Experiment 4 (Gradient Boost), learning_rate=0.1, n_estimators=100, random_state=42, max_depth=1

5. Evaluation

Performance Metric- AU ROC (Area Under the Receiver Operating Characteristic curve) is a reliable measure for binary classification tasks, particularly in scenarios where class imbalances or uneven costs of false positives and false negatives exist.

Classification Models		Validation set
S. No.	**Experiments**	**AU ROC score**
1	XGBClassifier(colsample_bytree=0.8, learning_rate=0.1, max_depth=4, min_child_weight=1, n_estimators=200, subsample=0.8, random_state=42)	0.8810
2	AdaBoostClassifier(learning_rate = 0.5, n_estimators=200, random_state=43)	0.9335
3	XGBClassifier(colsample_bytree=0.8, learning_rate=0.01, max_depth=4, min_child_weight=1, n_estimators=400, subsample=0.8, reg_lambda = 0.5, random_state=42)	0.9437
4	GradientBoostingClassifier(learning_rate=0.1, n_estimators=100, random_state=42, max_depth=1)	0.9676

Experiment 1

The XGBoost classifier is known for its robustness and good performance in many applications. The choice of hyperparameters seems reasonable, with a moderate learning rate, a modest tree depth (max_depth=4), and a substantial number of estimators (n_estimators=200).The AU ROC score of 0.8810 indicates decent performance on the test set. The model demonstrates good generalization. However, there is scope of improvement.

Experiment 2

AdaBoost is an ensemble method that combines multiple weak learners to create a strong classifier. The choice of hyperparameters includes a moderately high learning rate and a substantial number of estimators. The AU ROC score of 0.9335 suggests good performance on the test set. AdaBoost is considered as it is known for its ability to improve performance on a variety of classification tasks. Grid search has been utilized to find the best hyperparameters for the model.

Experiment 3

The AU ROC score of 0.9437 is slightly higher than Experiment 2, indicating better performance on the test set. While the learning rate is lower, the higher number of estimators allows the model to compensate for slower learning. This is done to ensure convergence. In this L2 regularization has been utilized to improve model stability and reduce the risk of overfitting. The regularization term, represented by the reg_lambda parameter set to 0.5, penalizes complex model structures, encouraging the model to favor simpler decision boundaries. This combination of a lower learning rate, a larger number of estimators, and L2 regularization contributes to the model's robustness and its ability to capture underlying patterns in the data without succumbing to noise or overcomplexity.

Experiment 4

GradientBoosting is a powerful ensemble method that builds decision trees sequentially to improve predictive performance. A moderate learning rate (0.1) and a relatively low tree depth which suggests a simpler model structure. The AU ROC score of 0.9676 is the highest among the experiments, indicating excellent performance on the test set. The choice of a shallow decision tree (max_depth=1) is interesting, as it results in a simpler model that still performs exceptionally well. This suggests that the model is capable of capturing complex relationships with a limited number of splits. The model generalises very effectively. The careful choice of independent features has contributed to the performance of the model.

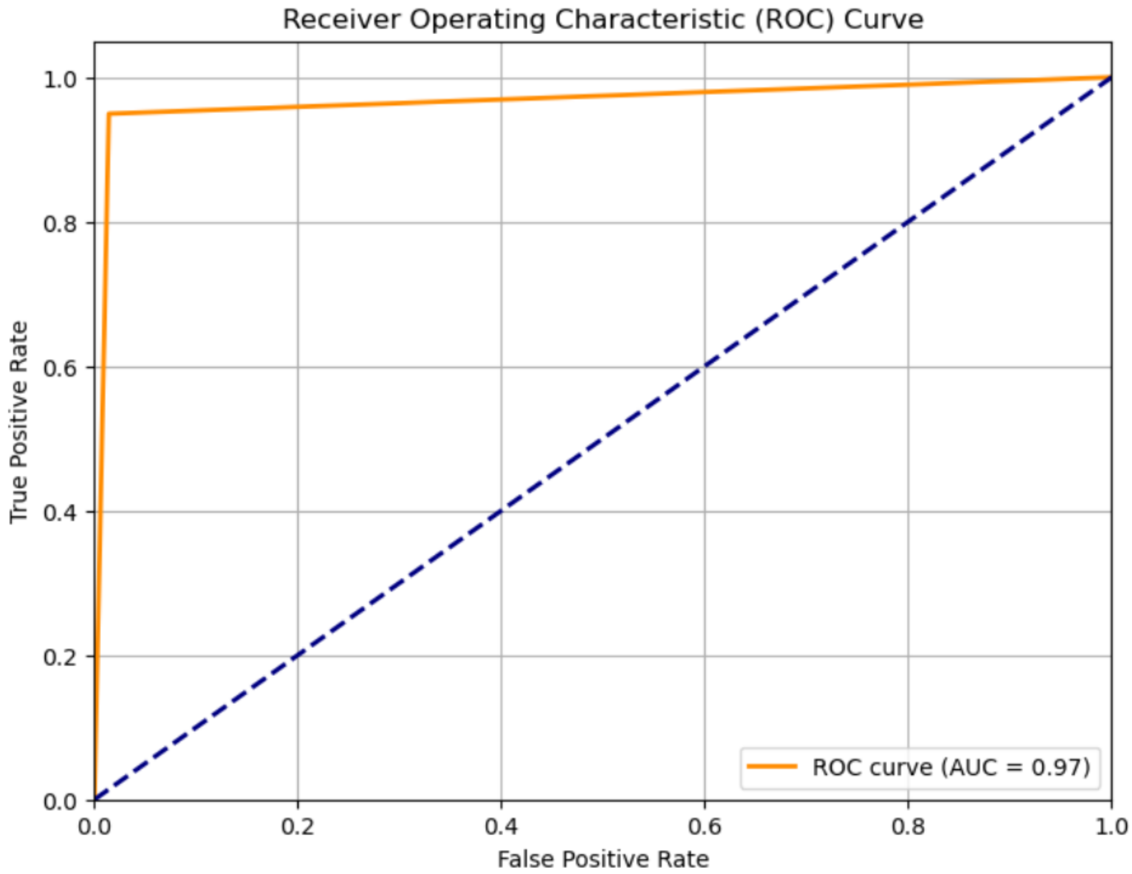

Figure 8: ROC curve for the best model (GradientBoost)

6. Deployment

In case, the model had a good performance on unseen data and is in alignment with the business requirements. The deployment process needs to be completed for stakeholders to access the results. In our case, we cannot proceed with deployment as the classification model does not perform well and is unreliable. We need to loop back to the previous steps.

Issues Faced

Feature Engineering Complexity: Crafting meaningful features from the available data posed a formidable challenge. Determining which player performance metrics and technical attributes were important for predicting NBA draft selections necessitated domain expertise.

Resource-Intensive Hyperparameter Tuning: Tuning hyperparameters for models like XGBoost and Gradient Boost demanded substantial computational resources. Identifying the optimal hyperparameter combination for superior model performance entailed extensive experimentation, sometimes leading to kernel crashes or unresponsive systems.

Model Interpretability Challenge: For complex ensemble models, comprehending the scores obtained were a challenge. Especially when the predicted probabilities are uploaded on Kaggle even if the model's performance was decent the score obtained was relatively low.

Generalization Concerns: Ensuring the trained models exhibited strong generalization capabilities to unseen data, especially for predicting future draft classes, remained an ongoing and critical consideration.

Demand for Computational Resources: Running extensive experiments and conducting hyperparameter tuning placed substantial demands on computational resources due to the dataset's size and complexity.

References

Hotz, N. (2022, February 22). What is the data science process? Data Science Process Alliance. https://www.datascience-pm.com/data-science-process/

Team, G. L. (2020, November 5). Why using CRISP-DM will make you a better Data Scientist? Great Learning Blog: Free Resources What Matters to Shape Your Career! https://www.mygreatlearning.com/blog/why-using-crisp-dm-will-make-you-a-better-data-scientist/

Appendices

Appendix A – Coding references

Note: Coding references has bee taken from lecture sources

NumPy. (2021). NumPy: The fundamental package for scientific computing with Python. Retrieved from https://numpy.org/(Accessed April 28, 2023)

pandas. (2022). pandas 1.3.3 documentation. Retrieved from https://pandas.pydata.org/docs/1.4.0/index.html(Accessed April 28, 2023)

scikit-learn. (2021). Scikit-learn: Machine learning in Python. Retrieved from https://scikit-learn.org/stable/index.html(Accessed April 28, 2023)

Lecture Tutorials

Figure (a)

Appendix B – Correlation Matrix

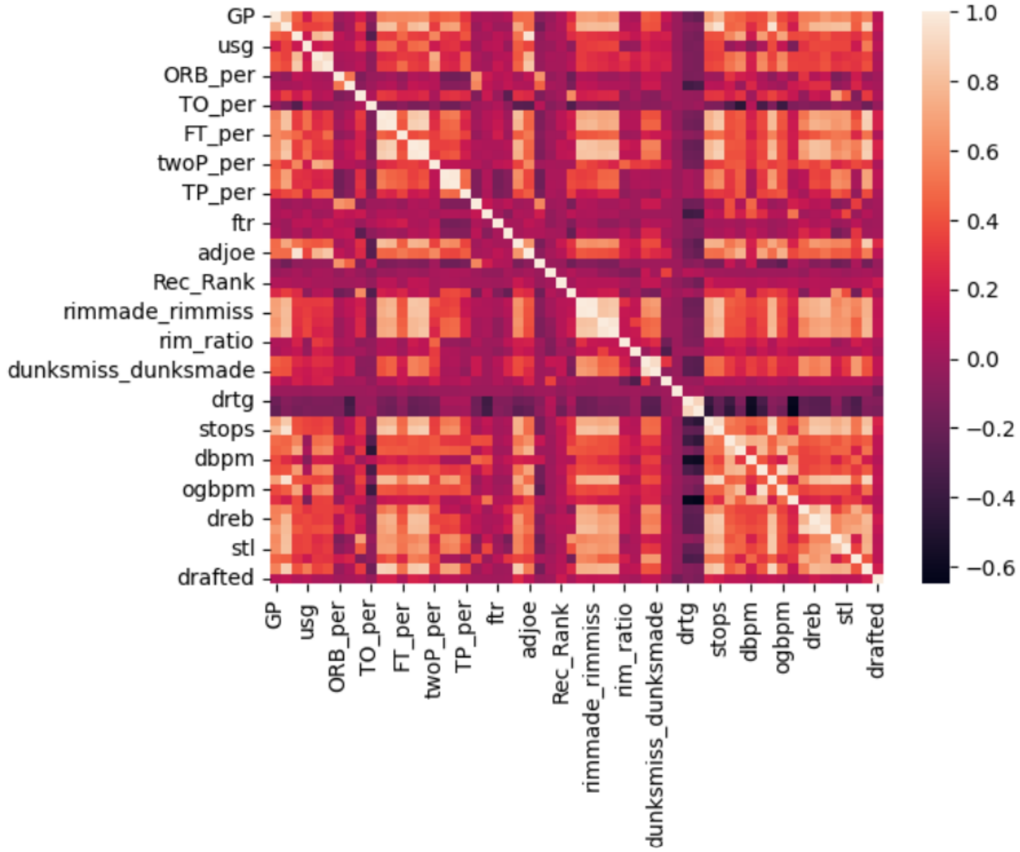

Figure (b)