# **EXPERIMENT REPORT**

|  |  |
| --- | --- |
| **Student Name** | Shivani Nandkishor Nipane\_24622969 |
| **Project Name** | **Assessment1- Binary Classification Project** |
| **Date** | 28/06/2023 |
| **Deliverables** | <Nipane\_ShivaniNandkishor-24622969-week1\_dataexploration.ipynb>  <Nipane\_ShivaniNandkishor-24622969-week1\_xgboost.ipynb>  <xgboost> |

|  |  |
| --- | --- |
| 1. **EXPERIMENT BACKGROUND** | |
| Provide information about the problem/project such as the scope, the overall objective, expectations. Lay down the goal of this experiment and what are the insights, answers you want to gain or level of performance you are expecting to reach. | |
| **1.a. Business Objective** | Explain clearly what is the goal of this project for the business. How will the results be used? What will be the impact of accurate or incorrect results?  The main goal of this project is to develop a predictive model that uses a rookie player's statistics to predict if they will last at least 5 years in the NBA league. This will be used by stakeholders such as team managers, talent scouts, analysts, or potentially even sports betting companies to inform their decisions and strategies.  Here's a detailed breakdown of how these results might be used and the impact they could have:  **Team Management and Talent Scouts:** These stakeholders are always looking for promising talent that will have longevity in their careers. If a rookie player is predicted to have a career of at least 5 years in the NBA, they may be considered a more stable and worthwhile investment for the team. This could influence decisions on who to draft, trade, or invest in for training and development.  **Sports Analysts and Commentators:** Predictions about player longevity could be used to inform analyses, articles, and broadcasts. This could generate interesting discussions and content for fans, and analysts who consistently make accurate predictions could improve their reputation.  **Sports Betting Companies or Bettors:** Accurate predictions about player longevity could be used to set betting odds or inform betting decisions.  As for the impact of accurate or incorrect results:  **Accurate Results:** Accurate predictions can provide valuable insights and give a competitive edge to the stakeholders. It can help in making informed decisions that could lead to better team performance, more accurate analyses, and better-informed betting.  **Incorrect Results:** Incorrect predictions can have significant consequences. A team may invest in a player who doesn't perform well or has a shorter career than expected, leading to wasted resources and potential missed opportunities. For sports analysts, incorrect predictions can damage their credibility. For bettors or sports betting companies, it could lead to financial losses.  The predictive model's performance should be continuously monitored and updated to incorporate the most recent data and ensure its reliability and accuracy. Regular maintenance and updating of the model is necessary to ensure it remains effective and accurate over time. |
| **1.b. Hypothesis** | Present the hypothesis you want to test, the question you want to answer or the insight you are seeking. Explain the reasons why you think it is worthwhile considering it,  The primary hypothesis for this project is: "A player's statistics during their rookie season can predict whether they will last at least 5 years in the NBA." Essentially, we want to answer the question: "Can we predict a player's career longevity based on their rookie season performance?"  This hypothesis is worthwhile for several reasons:  1. **Strategic Decisions:** If proven true, the hypothesis can significantly aid strategic decision-making for various stakeholders in the NBA, such as team management and talent scouts. Knowing whether a player is likely to have a long career could influence decisions about recruitment, development, and investment.  2. **Resource Allocation:** Teams could allocate their resources more efficiently by focusing on players likely to have longer careers.  3. **Fan Engagement:** Predicting player longevity can also help engage fans, as it gives them another dimension to consider when following their favorite teams and players. |
| **1.c. Experiment Objective** | Detail what will be the expected outcome of the experiment. If possible, estimate the goal you are expecting. List the possible scenarios resulting from this experiment.  The expected outcome of this experiment is a trained machine learning model, specifically using XGBoost, that can predict whether a rookie player will have a career lasting at least 5 years in the NBA, based on their performance statistics from their first season.  The measure of success for the model will be its performance on the AUROC (Area Under the Receiver Operating Characteristics) score. While it's hard to provide a specific estimate without knowing the data and problem complexity, a common target in binary classification problems is an AUROC above 0.7 for the test set, which could indicate a reasonably good model. However, the higher the AUROC, the better the model is at distinguishing between players who last at least 5 years and those who don't.  Here are the possible scenarios resulting from this experiment:  1. High AUROC Score (> 0.7) : This is the best-case scenario, where the model's predictions are accurate and reliable. This would allow stakeholders to use this model confidently to inform their decisions. If the score is particularly high, say over 0.85, the model is performing excellently.  2. Moderate AUROC Score (0.5 - 0.7) : In this scenario, the model has some predictive power but is not very reliable. The model may need further tuning, or additional, more informative features may be needed to improve the model's performance.  3. Low AUROC Score (< 0.5) : This is the worst-case scenario, where the model's predictions are not better than random guessing. In this case, a different approach may be needed, such as using a different algorithm, adding more features, or redefining the problem. |

|  |  |
| --- | --- |
| 1. **EXPERIMENT DETAILS** | |
| Elaborate on the approach taken for this experiment. List the different steps/techniques used and explain the rationale for choosing them. | |
| **2.a. Data Preparation** | Describe the steps taken for preparing the data (if any). Explain the rationale why you had to perform these steps. List also the steps you decided to not execute and the reasoning behind it. Highlight any step that may potentially be important for future experiments  The data preprocessing steps taken:  1. Splitting the dataset : The provided training set was split into a separate training set and a validation set. This was done to create a separate dataset (validation set) that the model has never seen during training. It's used to evaluate the model's performance and to check for overfitting. As the test set provided didn't contain target labels, it couldn't be used to evaluate the model's performance.  2. Handling Negative Values : Certain columns were found to contain negative values, which doesn't make sense in the context of the dataset (e.g. you can't have negative games played or a negative percentage for free throws). The values in these columns were replaced with their absolute values, under the assumption that the negatives were data entry errors.  3. Skewness : The skewness of all features was calculated. This is an important step as many machine learning algorithms assume that the data follows a Gaussian distribution. High skewness can lead to poor model performance.  4. Creating new features : New features were created that represented the average per game for each statistic (e.g., minutes per game, field goals made per game, etc.). This was done to capture the average performance of the player in each game, as the total values could be influenced by the number of games played.  5. Handling infinite values : Any infinite values resulting from the creation of the new features (possibly due to division by zero when games played was zero) were replaced with NaN.  6. Correlation Matrix : A correlation matrix was generated for the training set to understand the relationships between the different features. This is a crucial step to identify highly correlated features which might need handling to avoid multicollinearity.  Here's a potential preprocessing step that wasn't performed but might be important for future experiments:  - Handling missing values : Although it was stated that the dataset didn't contain missing values, it's always a good practice to check for missing values and handle them appropriately if any are found. Missing values can lead to erroneous results from machine learning models.  In addition, it's worthwhile mentioning that the code presents opportunities for improvement:  - Data Transformation : Some features were found to have high skewness. While the skewness was calculated, no transformation (like logarithm or square root transformation) was applied to reduce the skewness.  - Feature Scaling : No feature scaling (like standardization or normalization) was performed. This step is necessary for many machine learning algorithms that are sensitive to the scale of the features.  - Feature Selection : Although a correlation matrix was created, no explicit feature selection or dimensionality reduction techniques were applied. While it's fine for a first pass model, for more refined models, feature selection might be important to improve performance. |
| **2.b. Feature Engineering** | Describe the steps taken for generating features (if any). Explain the rationale why you had to perform these steps. List also the feature you decided to remove and the reasoning behind it. Highlight any feature that may potentially be important for future experiments  Given the presented code and information, here's the feature generation step taken:  Creating Per Game Features : Features were created to represent per game statistics for various attributes (e.g., minutes per game, field goals made per game, etc.). This step was undertaken to normalize player performance based on the number of games played. For example, a player who has played fewer games might have fewer total points simply because they have had fewer opportunities to score, not necessarily because they are a worse player. Therefore, creating per game features helps to better evaluate player performance on an average game basis.  However, there are a few potential feature generation opportunities that were not pursued in the code, but could be considered in future experiments:  1. Interaction Features : The creation of interaction features (i.e., features that are derived from two or more existing features) could potentially uncover new meaningful insights and improve model performance. For example, an interaction feature combining assists and points could capture how scoring ability interacts with teamwork.  2. Polynomial Features : Polynomial features could be created to capture more complex relationships between features. This is especially relevant for linear models, which cannot capture non-linear relationships between features.  3. Domain Specific Features : Considering domain specific knowledge to create new features could also be valuable. For example, in basketball, a popular statistic is the Player Efficiency Rating (PER), which is a measure of a player's per-minute productivity. It's a complex formula that takes into account various aspects of a player's performance.  Lastly, while a correlation matrix was generated, no explicit features were removed. In the future, this could be a useful step in feature selection, especially for features that are highly correlated with each other, to avoid the issue of multicollinearity. |
| **2.c. Modelling** | Describe the model(s) trained for this experiment and why you choose them. List the hyperparameter tuned and the values tested and also the rationale why you choose them. List also the models you decided to not train and the reasoning behind it. Highlight any model or hyperparameter that may potentially be important for future experiments  The chosen model for this experiment was XGBoost, a powerful and versatile machine learning algorithm known for its high performance and speed. XGBoost is based on the Gradient Boosting framework, which creates an ensemble of decision trees that predict the target variable. These trees are constructed in a sequential manner where each subsequent tree learns from the mistakes of the previous one, reducing the overall error.  The following hyperparameters were tuned using GridSearchCV:  - `n\_estimators`: [100, 200, 300]. This hyperparameter specifies the number of gradient boosted trees to use (i.e., the size of the ensemble). A larger number typically improves performance but also increases training time and the risk of overfitting.  - `learning\_rate`: [0.01, 0.05, 0.1]. This is the step size used in each update of the model weights. Smaller values make the model more robust to overfitting, but at the cost of longer training time.  - `max\_depth`: [3, 5, 7, 11, 13]. This parameter controls the maximum depth of each tree in the ensemble. Deeper trees can capture more complex patterns, but also increase the risk of overfitting.  This selection of hyperparameters is quite standard for an XGBoost model, aiming to balance model complexity and risk of overfitting.  The models not trained in this experiment could include other ensemble methods like Random Forests, or simpler models such as Logistic Regression. The rationale behind not choosing these models could be the high performance of XGBoost in many machine learning tasks, including classification tasks like this one. However, in future experiments, it could be valuable to compare the performance of XGBoost with other models to ensure it is the best choice for this particular task.  After hyperparameter tuning, there was a noticeable improvement in the Area Under the Receiver Operating Characteristic curve (AUROC) metric, which increased from 0.6601 to 0.6985. This metric is a common evaluation measure for binary classification problems, providing an aggregate measure of model performance across all possible classification thresholds. An AUROC of 0.5 suggests no discrimination (i.e., ability to distinguish the classes), 1 indicates perfect discrimination, and values in between reflect different levels of discrimination. Thus, the improvement here is quite significant, demonstrating the importance of hyperparameter tuning in model performance. |

|  |  |
| --- | --- |
| 1. **EXPERIMENT RESULTS** | |
| Analyse in detail the results achieved from this experiment from a technical and business perspective. Not only report performance metrics results but also any interpretation on model features, incorrect results, risks identified. | |
| **3.a. Technical Performance** | Score of the relevant performance metric(s). Provide analysis on the main underperforming cases/observations and potential root causes.  As mentioned earlier, the Area Under the Receiver Operating Characteristic (AUROC) was used as the performance metric in this project, which is a popular choice for binary classification problems. The final score for AUROC after hyperparameter tuning was 0.6985.  While this is not a bad score (it is better than random guessing, which would score 0.5), there is certainly room for improvement. An AUROC of 1 indicates perfect classification, so there are evidently some cases that the model is not classifying correctly.  To better understand which cases are being misclassified, we can look at the confusion matrix. This will allow us to see how many cases were true positives, false positives, true negatives, and false negatives. Unfortunately, without the actual confusion matrix or additional data, we can't provide a detailed analysis of the misclassified cases.  However, some potential root causes for misclassification could include:  1. Quality and relevance of the features : The predictive power of a model is highly dependent on the quality and relevance of the input features. If important features are missing, or if many features are noisy or irrelevant, the model's performance can suffer.  2. Imbalanced dataset : If the dataset is heavily imbalanced (i.e., there are many more examples of one class than the other), the model might learn to simply predict the majority class most of the time. This would result in a high accuracy but poor performance when it comes to correctly classifying the minority class. In this case, it was mentioned that the labels are imbalanced with over 83% of players playing at least 5 years in the NBA.  3. Model complexity : The complexity of the model (i.e., its capacity to learn complex patterns) can also impact its performance. If the model is too simple, it might not be able to learn the underlying patterns in the data (underfitting). On the other hand, if it is too complex, it might start to learn the noise in the data instead of the actual patterns (overfitting).  Improving on these points by adding more relevant features, dealing with the class imbalance, and fine-tuning the model complexity might help in improving the model's performance. |
| **3.b. Business Impact** | Interpret the results of the experiments related to the business objective set earlier. Estimate the impacts of the incorrect results for the business (some results may have more impact compared to others)  The objective of this project was to predict whether a basketball player would play at least 5 years in the NBA, with the ultimate goal of informing recruitment and retention strategies. The current XGBoost model yields an AUROC score of 0.6985, indicating it has a reasonable level of discriminative power between players who will and won't last 5 years.  However, given the high-stakes nature of NBA recruitment—where decisions can involve millions of dollars—it's important to acknowledge that every incorrect prediction carries potential impacts.  There are two types of incorrect results that can arise:  1. False Positives (Type I Error) : These occur when the model incorrectly predicts that a player will last 5 years in the NBA. The business implications of this error could be significant, especially if the organization invests heavily in the development of a player based on this prediction. This could lead to financial losses, missed opportunities to recruit other players, and even damage to the team's performance and reputation.  2. False Negatives (Type II Error) : These occur when the model incorrectly predicts that a player will not last 5 years in the NBA. This could potentially lead to the organization missing out on talented players who might turn out to be valuable assets.  Improving the model's performance would reduce these errors, thereby potentially saving the organization from significant financial losses and missed opportunities. It could also assist in making better strategic decisions in terms of player development and team building.  On the other hand, it's important to remember that predictions made by the model should not be the sole determinant in player recruitment and retention. They should be used as part of a holistic decision-making process, in conjunction with other crucial factors like the player's health, performance under pressure, ability to work as part of a team, and more.  In the future, adding these factors into the model as features, if the data is available, could help improve the model's performance and make its predictions even more useful for the business. |
| **3.c. Encountered Issues** | List all the issues you faced during the experiments (solved and unsolved). Present solutions or workarounds for overcoming them. Highlight also the issues that may have to be dealt with in future experiments.  During the execution of the experiments, the following issues were encountered:  1. Data Imbalance : The training data was significantly imbalanced with a majority of players lasting 5 years in the NBA. This could potentially bias the model towards predicting the majority class. The imbalance issue was not directly addressed in this experiment. However, future iterations could implement techniques such as SMOTE (Synthetic Minority Over-sampling Technique) or ADASYN (Adaptive Synthetic Sampling) to create synthetic samples and balance the data.  2. Missing `TARGET\_5Yrs` column in the test set : This column was not available in the test set which made it difficult to evaluate the model. The workaround was to split the training set into a training and validation set. However, this reduced the amount of data available for training the model. In future experiments, if a labelled test set becomes available, it would be beneficial for evaluating the model.  3. Presence of Negative Values in the Data : Some columns contained negative values which didn't make sense in the context (e.g., number of games played or block percentage). Some of these were assumed to be typos and replaced with absolute values, while others were kept as they formed a normal distribution, suggesting they may not be errors. Future experiments would benefit from a deeper understanding or clarification of these negative values.  4. Hyperparameter Tuning : The selection of hyperparameters was done through a grid search, which is computationally expensive. An alternative approach for future experiments could be to use a randomized search or Bayesian optimization for hyperparameter tuning, which could potentially yield similar results with less computational effort.  5. Feature Engineering : New features were created such as 'per game' statistics, which added value to the model. However, feature engineering is an iterative process and further experimentation with different features could lead to improvements in the model's performance.  6. Model Selection : Only XGBoost was used in this experiment due to its reputation for high performance on structured data. However, future experiments should also consider other models like Logistic Regression, LightGBM or even Neural Networks. Ensemble methods combining several models could also be a strategy to improve performance. |

|  |  |
| --- | --- |
| 1. **FUTURE EXPERIMENT** | |
| Reflect on the experiment and highlight the key information/insights you gained from it that are valuable for the overall project objectives from a technical and business perspective. | |
| **4.a. Key Learning** | Reflect on the outcome of the experiment and list the new insights you gained from it. Provide rationale for pursuing more experimentation with the current approach or call out if you think it is a dead end.  The experiment provided some valuable insights into predicting NBA career longevity based on rookie year statistics. Here are some key takeaways:  1. Data Imbalance : The significant data imbalance indicates that most players do last for 5 years or more in the NBA. This context is critical as it influences the strategy for modeling, evaluation metrics choice, and how results are interpreted.  2. Feature Importance : The feature engineering step of creating per-game statistics suggests that per-game performance might be more indicative of a player's potential career longevity than cumulative statistics. This intuition aligns with domain knowledge, as a player's per-game statistics are usually a better reflection of their talent and ability.  3. Model Performance : The performance of the XGBoost model was decent, with an AUROC of 0.6985 after hyperparameter tuning. It suggests that the XGBoost model can be a good candidate for this type of problem. However, there's room for improvement.  Based on these insights, it's worthwhile to continue exploring this problem with the current approach. The following avenues could be considered for further experimentation:  1. Address Data Imbalance : Techniques such as SMOTE or ADASYN could be employed to balance the dataset and potentially improve model performance.  2. Explore Other Models : While XGBoost performed reasonably well, other models like LightGBM, CatBoost, or even neural networks could be explored. Ensemble models could also be a potential way to boost performance.  3. Further Feature Engineering : More features could be engineered, and a thorough feature selection process could be employed to isolate the most important features.  4. Evaluation Metric : In light of the imbalanced data, metrics that are more robust to imbalance, such as F1 score, Precision, Recall, or the Area Under the Precision-Recall Curve (AUPRC) could be used.  5. More Advanced Hyperparameter Tuning : Methods such as randomized search or Bayesian optimization could be used to improve the efficiency of the hyperparameter tuning process.  Overall, the experiment serves as a good starting point, and the gained insights inform us about potential future steps to refine the model and improve its performance. |
| **4.b. Suggestions / Recommendations** | Given the results achieved and the overall objective of the project, list the potential next steps and experiments. For each of them assess the expected uplift or gains and rank them accordingly. If the experiment achieved the required outcome for the business, recommend the steps to deploy this solution into production.  Given the results and the overall objective of the project, potential next steps and experiments can be as follows:  1. Address Data Imbalance (Expected uplift: High) : The first priority should be to address the data imbalance in the target variable. Techniques such as SMOTE or ADASYN could be used to artificially balance the dataset and potentially improve model performance.  2. Further Feature Engineering (Expected uplift: Medium-High) : Additional features could be engineered based on domain knowledge, and a rigorous feature selection process could be employed to isolate the most predictive features.  3. Explore Other Models (Expected uplift: Medium) : Although XGBoost provided decent results, experimenting with other models like LightGBM, CatBoost, or even neural networks could potentially lead to better results. Ensemble techniques could also be explored.  4. More Advanced Hyperparameter Tuning (Expected uplift: Medium) : Methods such as randomized search or Bayesian optimization could be used to improve the efficiency of the hyperparameter tuning process and potentially enhance the model's performance.  5. Evaluation Metric (Expected uplift: Low-Medium) : It could be beneficial to explore other metrics that are more robust to imbalanced data, such as the F1 score, Precision, Recall, or the Area Under the Precision-Recall Curve (AUPRC).  If the experiment has achieved the required business outcome, the next steps should be to:  1. Validation : Validate the model with a new set of data to ensure that it can generalize well to unseen data.  2. Model Interpretability : Interpret the model using techniques like SHAP (SHapley Additive exPlanations) to understand the impact of each feature on the prediction.  3. Deployment : The model could be wrapped into a web service (using tools like Flask or FastAPI) and deployed to a server or a cloud platform like AWS, Google Cloud, or Azure. This would allow other systems or services in the business to use it.  4. Monitoring : It would be important to monitor the model's performance over time to ensure it maintains its accuracy as new data comes in. If the model's performance decreases significantly, it may need to be retrained or tweaked.  5. Documentation : Ensure that the project, including the model, its parameters, performance, and any assumptions made, is well-documented. This will be essential for future maintenance and improvements. |