# **EXPERIMENT REPORT**

|  |  |
| --- | --- |
| **Student Name** | Kiran Shanker Das |
| **Project Name** | AT1- Kaggle Competition |
| **Date** | 18th August 2023 |
| **Deliverables** | <best\_model.pynb>  < XGBoost>< Logistic Regression>  Github Link: <https://github.com/kirandas-dev/NBAComp> |

|  |  |
| --- | --- |
| 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?  **Goal and Utilization:** The project's core aim is to construct a predictive model that assesses the likelihood of a college basketball player being drafted by a professional team. By analysing player performance metrics, the model aims to identify the key factors contributing to draft eligibility. The model's predictions will be crucial for basketball teams and scouts during the draft selection process, guiding their decisions on player choices. This data-driven approach will help teams allocate resources more effectively, identify hidden talent, and refine their draft strategies to align with team requirements.  **Impact of Results:** Accurate predictions will offer substantial advantages. Teams can efficiently focus their scouting efforts on players with higher predicted draft probabilities, uncover hidden gems, and make informed draft decisions. However, incorrect predictions could lead to missed opportunities, misallocation of scouting resources, and poor draft choices. Thus, the project's success could redefine player draft processes, enhancing decision-making through accurate predictions, while inaccuracies could result in suboptimal outcomes affecting team performance and resource efficiency. |
| **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.  **Hypothesis:** Can considering a few more factors and attributes of college basketball players significantly influence their likelihood of being drafted by a professional team. We have already achieved a 95%+ accuracy rate from the previous model. Can we achieve a higher accuracy with feature engineering of missing values?  **Question**: What are the key factors and attributes of college basketball players that contribute to their likelihood of being drafted by a professional team?  Below are the considerations:  **Feature Selection:**  We selected features with normal distributions: 'Ortg', 'eFG', 'TS\_per', 'adjoe'.  Categorical features: 'yr', 'ht', 'num' were removed as they added no value.  We identified skewed features: 'GP', 'Min\_per', 'usg', and others.  **Feature Engineering and Imputation:**  KNN imputation is applied to normal distribution features using 5 neighbors.  Manual imputation with predefined values for specific features.  Mode imputation for the 'pick' feature.  Constant value imputation for 'Rec\_Rank'.  Manual imputation with a predefined median for 'rimmade\_rimmiss'.  Zero imputation for 'dunksmade', 'midmade', 'rimmade'.  Mode imputation for categorical features.  Median imputation for skewed features.  Central value imputation for percentage features. |
| **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.  **Expected Outcome:**  The experiment aims to create a predictive model for a college basketball player's draft likelihood. The expected outcome is a model with around around 98% accuracy in predicting draft success.  **Possible Scenarios:**  **High Accuracy:** Model accurately predicts draft outcomes (98%+ accuracy), providing valuable insights.  **Moderate Accuracy:** Model has 95-96% accuracy, still offering useful guidance.  **Bias:** Model shows accuracy but highlights bias, emphasizing fairness importance.  **Fit Issues:** Model overfits/underfits, requiring adjustments.  **Surprising Attributes:** New insights from unexpected attribute correlations.  **External Factors:** Model reveals external factors' importance.  **Feedback Loop:** Stakeholder feedback iteratively improves the model. |

|  |  |
| --- | --- |
| 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  **Data Filtering:** We loaded data from a CSV and filtered out columns with excessive missing values.  **Missing Value Imputation:** Filled numerical missing values with medians and categorical with modes.  **Feature Dropping:** Removed the 'year' column and other irrelevant columns.  **Feature Engineering:**  **KNN Imputation:** Features with normal distribution are imputed using KNN.  **Median Imputation:** Skewed features are imputed with medians.  **Mode Imputation:** Categorical features are imputed with modes.  **Outside Range Imputation:** 'Rec\_Rank' is set to a value outside the rank range.  **Zero Imputation:** Certain features are imputed with zeros.  **Percentage Imputation:** Percentage features are imputed with central values.  **Train-Validation-Test Split:** Split data into training, validation, and test sets, while keeping an eye on class distributions.  **Data Resampling:** Applied SMOTE for balancing and standard scaling for normalization.  **Saving Processed Data:** Saved processed datasets and resampling tools (Scaler, SMOTE) using joblib.  **Visualization:** Used correlation heatmap to identify highly correlated features.  **Highly Correlated Features:** Found pairs of features with strong correlations, crucial for addressing multicollinearity. |
| **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  **K-Nearest Neighbors (KNN) Imputation:** I am using the KNNImputer from scikit-learn to impute missing values in features that exhibit a normal distribution. KNN imputation estimates missing values based on the values of their k-nearest neighbors in the dataset. This method leverages the similarity between data points to impute missing values, which can preserve the underlying relationships in the data.  **Median Imputation:** For certain features like 'dunks\_ratio', 'dunksmiss\_dunksmade', 'rim\_ratio', and 'mid\_ratio', I am imputing missing values with the median of each respective feature. Median imputation is a robust method for imputing skewed or non-normally distributed data, as it's less sensitive to extreme values.  **Mode Imputation:** Imputing categorical features like 'yr', 'ht', and 'num' with the mode (most frequent value) is appropriate to maintain the categorical nature of the data.  **Imputing Outside the Range:** I have chosen to impute 'Rec\_Rank' with a value outside the range of existing ranks, such as -1. This approach signals that the data was missing and avoids biasing the model towards any specific rank.  **Zero Imputation:** Some features like 'dunksmade', 'midmade', and 'rimmade' are imputed with zeros, likely indicating that no dunks, mid-range shots, or rim shots were made.  **Imputing Percentage Features:** For percentage features like 'FT\_per', 'TP\_per', and 'blk\_per', you've imputed missing values with reasonable central values (medians), ensuring that the imputed data aligns with the typical distribution of these percentages. |
| **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  In this experiment, several XGBoost models were trained for classification. The goal was to predict a binary outcome. The dataset was preprocessed, and a set of features were selected based on their absolute coefficients obtained from a previously trained Logistic Regression model. The selected features were:  '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', 'porpag', 'adjoe', 'pfr', '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'  The chosen algorithm was XGBoost, which is an ensemble boosting algorithm known for its strong performance on structured data and its ability to capture complex relationships.  The hyperparameters were tuned using GridSearchCV with cross-validation. The following hyperparameters were tested:  **learning\_rate:** [0.01, 0.18, 0.2, 0.22, 0.25]  **max\_depth:** [2, 4, 5, 6]  **n\_estimators:** [250, 300, 310]  **reg\_lambda:** [0.1, 0.15, 0.2]  The rationale behind these hyperparameters is as follows:  **learning\_rate:** Determines the step size at each iteration. A lower value makes the optimization process more robust but slower.  **max\_depth:** Controls the depth of the individual trees. A higher value can capture more complex relationships but may lead to overfitting.  **n\_estimators:** The number of boosting rounds (trees). More trees can improve performance but may increase computation time.  **reg\_lambda:** L2 regularization term on weights. It helps prevent overfitting.  Models that were not trained and the reasoning behind it:  **Random Forest:** Not trained because XGBoost generally outperforms Random Forest on structured data due to its better handling of gradient boosting.  **SVM:** Not trained because XGBoost often performs better on structured data without the need for extensive tuning. |

|  |  |
| --- | --- |
| 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.  **Performance Scores:**  Best XGBoost Model (with selected features and hyperparameters):  Validation Accuracy: 98.37%  Validation F1 Score: 98.38%  Test Accuracy: 98.92%  Test F1 Score: 98.92%  Validation AUROC: 99.77%  Test AUROC: 99.85%  **Potential Causes:**  **Inadequate Feature Coverage:**  Some features might not adequately capture the patterns in underperforming cases, leading to suboptimal predictions.  **Complex Interactions:**  The model might struggle with capturing complex interactions between features that are more prevalent in underperforming cases.  **Model Sensitivity to Hyperparameters:**  While the chosen hyperparameters work well overall, there might be certain cases where fine-tuning hyperparameters specifically for underperforming cases could lead to improvements.  **Conclusion:** Both models show promise, but refinement can enhance performance. We are investigating causes, monitoring and testing for real-world deployment. |
| **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 experiments revealed that both the best\_params (best\_params **=** {'learning\_rate': 0.0145, 'max\_depth': 2, 'n\_estimators': 400, 'reg\_lambda': 0.5}  ) used on XGBoost models performed well in predicting basketball player success. The logistic regression achieved around 99.3% accuracy on both validation and test sets, indicating its reliability.  Potential Impacts:  **Recruitment Decisions**: Misclassifying successful players could lead to missing out on talent.  **Team Composition**: Incorrect predictions might affect team performance.  **Resource Allocation**: Wrongly allocating resources to players could hinder growth.  **Fan Engagement**: Misclassified players might impact fan excitement and sponsorships.  **Team Reputation**: Incorrect predictions could harm team reputation and fan loyalty.  **Opponent Strategy**: Mistaken player assessments could affect opponent strategies.  **Financial Consequences**: Poor player choices may result in financial losses.  Continuously refining models, addressing limitations, and validating predictions is crucial to make informed decisions and prevent negative business outcomes. |
| **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.  Issues Faced During Experiments:  **Data Quality and Preprocessing**:  **Missing Values**: Solved by imputing with median/mode. Future: Explore advanced imputation techniques.  **Feature Selection**: Solved by domain knowledge. Future: Experiment with automated feature selection methods.  **Class Imbalance**:  **Imbalanced Classes**: Solved using SMOTE. Future: Explore other resampling techniques or ensemble methods.  **Model Selection and Performance**:  **Model Choice**: Explored logistic regression and XGBoost so far. Solved by evaluating multiple models. Future: Experiment with different models like Ada-Boost.  **Overfitting**: Observed in XGBoost on test set. Solved by hyperparameter tuning and feature selection. Future: Regularization techniques, more data.  **Hyperparameter Tuning**:  **Manual Tuning**: Solved by using GridSearchCV. Future: Implement more feature engineering to compliment GridSearch process for finding the best hyperparameters  **Feature Importance**: Solved by analyzing coefficients and XGBoost's feature importance. Future: Explore SHAP values, LIME, or other interpretability methods.  **Scalability and Generalization**:  **Data Scale**: Solved by applying standard scaling. Future: Test model scalability with larger datasets.  **Model Deployment**:  **Saving Model**: Solved using joblib. Future: Explore deployment platforms, containerization, and monitoring.  **Evaluation Metrics**:  **Metric Choice**: Solved by using accuracy, AUROC, etc. Future: Consider domain-specific metrics or ensemble models.  Issues to Address in Future Experiments:  **External Validation**: Validate models on more recent or external data to assess generalization.  **Feature Engineering**: Explore creating new features, interactions, or transformation methods.  **Hyperparameter Optimization**: Implement advanced optimization techniques like Bayesian optimization.  **Explainability**: Utilize advanced explainability tools for better model interpretation. |

|  |  |
| --- | --- |
| 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.  **Outcome of the Experiment:**  Missing values were imputed using KNNImputer for normal distribution features, and specific values for other features.  Feature selection was performed by dropping irrelevant columns.  Class imbalance was addressed using SMOTE after scaling.  **New Insights Gained:**  Effective feature selection and engineering reduced dimensionality and noise.  Class imbalance in the target variable was identified and addressed.  **Further Experimentation Rationale:**  Proceed with training and evaluating classification models.  Hyperparameter tuning and exploring ensemble methods can improve results.  Evaluate models using appropriate metrics for imbalanced data.  Consider domain expertise and potential external validation.  **Dead End Consideration:**  Without evaluating model performance, it's premature to label the approach a "dead end". Although, a score 99.3 is pretty decent in real world use case. It would only be prudent to test this model on more unseen data to see how good it is in generalizing.  Focus on model training, evaluation, and tuning to validate preprocessing steps and feature selection. |
| **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.  **Potential Next Steps and Experiments:**  **Hyperparameter Tuning:** Fine-tune model parameters to enhance performance.  **Advanced Algorithms:** Experiment with more complex algorithms like Random Forest for better predictions.  **External Data Integration:** Combine player rankings and college stats to enhance predictive power.  **Ranking:**  Hyperparameter Tuning  Feature Engineering  External Data Integration  **Deployment Steps (if Achieved Business Goals):**  **Validation:** Test models on new season's data.  **Setup and Integration:** Prepare production environment and integrate selected model.  **Data Pipeline:** Create real-time data feed for predictions.  **Monitoring and Updates:** Implement performance monitoring and continuous model updates.  **Domain Expertise:**  Continued experimentation benefits from domain expertise, aligning models with basketball player evaluation methods and trends, leading to more accurate predictions and actionable insights. |