Project Report: Predictive Modeling on NBA Players

1. Attribute Selection and Feature Engineering

When preparing the data for training, I examined the dataset for irrelevant attributes. We identified the 'Name' column as a non-numeric, text attribute with no predictive value for our models. Therefore, we removed it to avoid introducing anomalies or noise into the models. I did not add any new attributes were added since the dataset provided sufficient features for training. However, we ensured that missing values in the dataset were handled using mean imputation to maintain data integrity and completeness.

2. Feature Normalization

I applied Feature scaling using StandardScaler for distance-based models like K-Nearest Neighbors (KNN) and Artificial Neural Networks (ANN). Using Normalization significantly improved the performance of the KNN model, because it relies on distance metrics that are sensitive to feature scales. For ANN models, normalization contributed to faster convergence and improved performance stability. For models like Random Forest and Logistic Regression, scaling had little to no effect since they are not distance-based.

3. Effect of Regularization in Logistic Regression

I experimented with different penalty parameters ('none', 'l1', 'l2', and 'elasticnet') in logistic regression to observe their impact on performance. The F1 score on testing data varied like:

- **None**: The model tended to overfit, leading to lower performance.
- **L1 (Lasso)**: Encouraged sparsity in the model, reducing overfitting but sometimes at the cost of slightly reduced accuracy.
- **L2 (Ridge)**: It Provided a strong balance, reducing overfitting while maintaining performance.
- **ElasticNet**: This Offered flexibility and performed competitively. In my experiments, **Logistic Regression with L2 regularization** delivered the best F1 score on testing, showing that controlling complexity through ridge regression worked best with our dataset.

4. Hyperparameter Tuning with GridSearch

I used GridSearchCV to tune hyperparameters for models, which led to the following best parameters:

- **KNN**: n_neighbors=5 gave the best trade-off between bias and variance.

- **Random Forest**: The best results were achieved with n_estimators=200, balancing model complexity and performance.
- **Logistic Regression**: The penalty='l2' setting was optimal, along with the solver liblinear.
- **ANN (MLPClassifier)**: The best performance came from using hidden layers of size (64, 32), learning_rate_init=0.001, solver='adam', and early_stopping=True.

5. Best Performing Model

Model	**10-Fold CV Mean F1 Score**	**Test F1 Score**
K-Nearest Neighbors (KNN)	0.7147	0.7598
Random Forest	0.7457	0.7765
Logistic Regression (L2)	0.8056	0.7911
Artificial Neural Network (ANN)	0.7616	0.7899

Among all the models I tested, **Logistic Regression with L2 regularization** provided the best F1 score on the test set in my experiments. It demonstrated a good balance between bias and variance, excellent generalization ability, and performed consistently well across different train-test splits.

Although Random Forest and ANN models performed strongly, Logistic Regression stood out for its interpretability, simplicity, and reliable performance with the tuned penalty parameters.

Conclusion: Based on the results, the Logistic Regression model with L2 penalty and tuned solver settings emerged as the best performer. Careful preprocessing, scaling for distance-based models, and hyperparameter tuning played crucial roles in achieving these optimal results.