COMPARATIVE MODEL ANALYSIS FOR DIABETES PREDICTION

DA 204O: **DATA SCIENCE IN PRACTICE: COURSE PROJECT REPORT**

DATE: 5TH DECEMBER 2024

BY,

Kapil Soni, IISc, kapilsoni@iisc.ac.in

Mahendra Mahajan, IISc, mmahendra@iisc.ac.in

Malu Jayachandran, IISc, maluj@iisc.ac.in

Tanzimur Rahman, IISc, tanzimrahman@iisc.ac.in

INDEX

1. PROBLEM DEFINITION

1.1 BACKGROUND OF THE PROBLE

Early detection of diabetes and management are crucial for preventing many complications and improving the quality of life for individuals with diabetes. There are many datasets which includes various health metrics and lifestyle factors that can potentially indicate the onset of diabetes. We can develop predictive models to identify at-risk individuals by analysing these indicators, enabling early intervention and proactive management.

1.2 WHY IS IT IMPORTANT?

Health Impact:

Early detection of diabetes can prevent or delay complications associated with the disease, thereby improving patient outcomes and quality of life.

Economic Impact:

Diabetes management and complications contribute to significant healthcare costs. Early diagnosis and intervention can reduce these costs by minimizing the need for extensive medical treatment.

Public Health: Identifying at-risk populations can help in designing targeted public health strategies and campaigns to promote healthier lifestyles and reduce the prevalence of diabetes.

Personalized Medicine:

Predictive models can enable personalized healthcare plans, tailored to an individual's risk profile, thereby enhancing the effectiveness of interventions.

1.3 OBJECTIVES OF THE PROJECT

Create machine learning models to predict the likelihood of an individual developing diabetes based on their health indicators and lifestyle factors

2. DATA SOURCES

CDC Diabetes Health Indicators Dataset

Source: UCI Machine Learning Repository

URL: CDC Diabetes Health Indicators
: https://archive.ics.uci.edu/dataset/891/cdc+diabetes+health+indicators

Collection Method: This dataset is derived from the Behavioral Risk Factor Surveillance System (BRFSS) survey conducted by the Centers for Disease Control and Prevention (CDC). The data were collected via telephone surveys.

3. DATA PREPROCESSING

Feature for the dataset and details are given below

Variable Name	Role	Type	Description	Missing Values
ID	ID	Integer	Patient ID	no
Diabetes_binary	Target	Binary	0 = no diabetes 1 = prediabetes or diabetes	no
HighBP	Feature	Binary	0 = no high BP 1 = high BP	no
HighChol	Feature	Binary	0 = no high cholesterol 1 = high cholesterol	no
CholCheck	Feature	Binary	0 = no cholesterol check in 5 years 1 = yes cholesterol check in 5 years	no
BMI	Feature	Integer	Body Mass Index	no
Smoker	Feature	Binary	Have you smoked at least 100 cigarettes in your entire life? [Note: 5 packs = 100 cigarettes] 0 = no 1 = yes	no
Stroke	Feature	Binary	(Ever told) you had a stroke. 0 = no 1 = yes	no
HeartDiseaseorAttack	Feature	Binary	coronary heart disease (CHD) or myocardial infarction (MI) 0 = no 1 = yes	no
PhysActivity	Feature	Binary	physical activity in past 30 days - not including job 0 = no 1 = yes	no
AnyHealthcare	Feature	Binary	Have any kind of health care coverage,including health insurance, prepaid plans such as HMO, etc 0 = no 1 = yes	no
NoDocbcCost	Feature	Binary	Was there a time in the past 12 months when you needed to see a doctor but could not because of cost? 0 = no 1 = yes	no
GenHlth	Feature	Integer	Would you say that in general your health is: scale 1-5 1 = excellent 2 = very good 3 = good 4 = fair 5 = poor	no
MentHlth	Feature	Integer	Now thinking about your mental health, which includes stress, depression, and problems with emotions, for how many days during the past 30 days was your mental health not good? scale 1-30 days	no
PhysHlth	Feature	Integer	Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good? scale 1-30 days	no
DiffWalk	Feature	Binary	Do you have serious difficulty walking or climbing stairs? 0 = no 1 = yes	no
Sex	Feature	Binary	Sex 0 = female 1 = male	no
Fruits	Feature	Binary	Consume Fruit 1 or more times per day 0 = no 1 = yes	no
Veggies	Feature	Binary	Consume Vegetables 1 or more times per day 0 = no 1 = yes	no
HvyAlcoholConsump	Feature	Binary	Heavy drinkers (adult men having more than 14 drinks per week and adult women having more than 7 drinks per week) 0 = no 1 = yes	no
Age	Feature	Integer	Age 13-level age category (_AGEG5YR see codebook) 1 = 18-24 9 = 60-64 13 = 80 or older	no
Education	Feature	Integer	Education Level Education level (EDUCA see codebook) scale 1-6 1 = Never attended school or only kindergarten 2 = Grades 1 through 8 (Elementary) 3 = Grades 9 through 11 (Some high school) 4 = Grade 12 or GED (High school graduate) 5 = College 1 year to 3 years (Some college or technical school) 6 = College 4 years or more (College graduate)	no
Income	Feature	Integer	Income scale (INCOME2 see codebook) scale 1-8 1 = less than $10,0005 = lessthan35,000$ 8 = $75,000 or more	no

Figure 1: Features

- No null values
- No duplicate values
- Not much data preprocessing is needed

4. EXPLORATORY DATA ANALYSIS

Methods performed

1. Univariate and multivariate analysis
2. Plot the distribution and correlation of the data
3. Correlation analysis

Figure 2: Sample Distribution

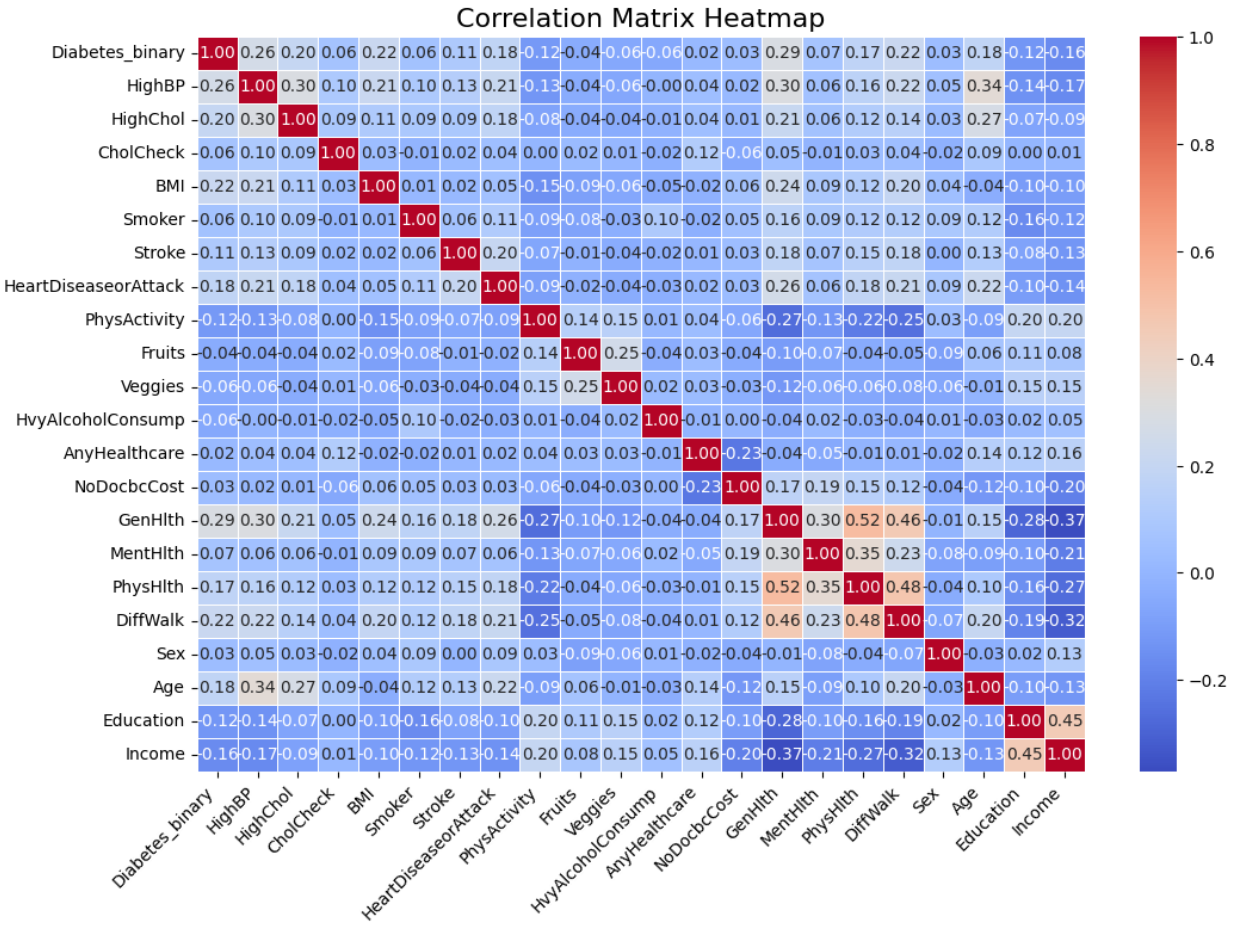

Figure 3: Correlation Matrix

Summary from EDA:

1. Target Feature of the Data is Diabetes_binary.

2. dataset has 15 Discrete type and 7 continuous type feature variables.

3. Dataset does not have missing values(null values).

4. major feature variables for Diabetes are : HIghBP , HighChol , BMI, PhysicalActivity , GenHlth , MentHlth , PhysHlth , Age , Eduation and Income.

5. Feature variables which increases the risk of Diabetes togather are : Smoking and HvyAlcoholConsump , Stroke and HeartDiseaseorAttack , HighBP and HighChol.

6. Feature variable Which is least effective on Diabetes , but they can help in decreasing the risk Diabetes are : Fruits , Veggies , AnyHealthcare , CholChek.

5. FEATURE ENGINEERING

Methods Used for Feature engineering

1. Correlation analysis
2. Feature Importance from first iteration of the model

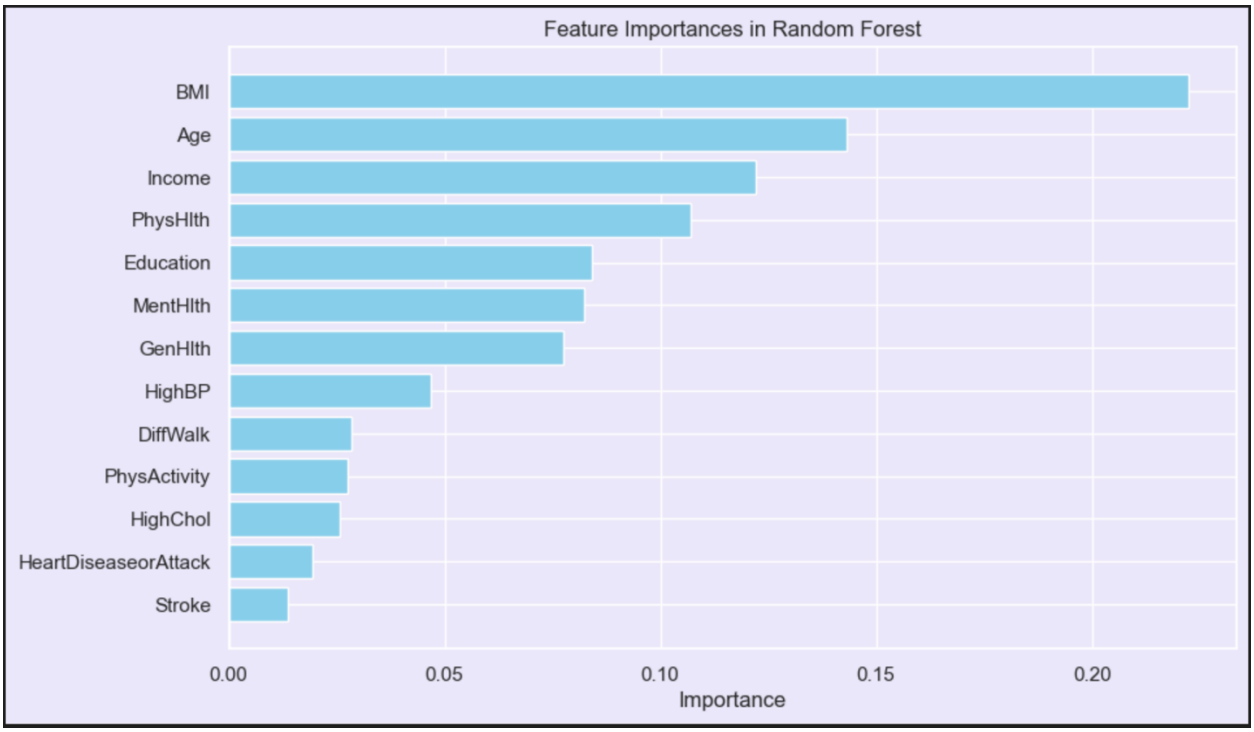

Figure 4: Feature Importance

Features selected after feature importance

Variable Name	Role	Type	Description	Missing Values
HighBP	Feature	Binary	0 = no high BP 1 = high BP	no
HighChol	Feature	Binary	0 = no high cholesterol 1 = high cholesterol	no
BMI	Feature	Integer	Body Mass Index	no
PhysActivity	Feature	Binary	physical activity in past 30 days - not including job 0 = no 1 = yes	no
GenHlth	Feature	Integer	Would you say that in general your health is: scale 1-5 1 = excellent 2 = very good 3 = good 4 = fair 5 = poor	no
MentHlth	Feature	Integer	Now thinking about your mental health, which includes stress, depression, and problems with emotions, for how many days during the past 30 days was your mental health not good? scale 1-30 days	no
PhysHlth	Feature	Integer	Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good? scale 1-30 days	no
DiffWalk	Feature	Binary	Do you have serious difficulty walking or climbing stairs? 0 = no 1 = yes	no
Age	Feature	Integer	Age 13-level age category (_AGEG5YR see codebook) 1 = 18-24 9 = 60-64 13 = 80 or older	no
Education	Feature	Integer	Education Level Education level (EDUCA see codebook) scale 1-6 1 = Never attended school or only kindergarten 2 = Grades 1 through 8 (Elementary) 3 = Grades 9 through 11 (Some high school) 4 = Grade 12 or GED (High school graduate) 5 = College 1 year to 3 years (Some college or technical school) 6 = College 4 years or more (College graduate)	no
Income	Feature	Integer	Income scale (INCOME2 see codebook) scale 1-8 1 = less than $10,000 = lessthan35,000 8 = $75,000 or more	no
Stroke	Feature	Binary	(Ever told) you had a stroke. 0 = no 1 = yes	no
HeartDiseaseorAttack	Feature	Binary	coronary heart disease (CHD) or myocardial infarction (MI) 0 = no 1 = yes	no

6. MODEL DEVELOPMENT

Since the target variable is binary, we have modelled this as a classification problem.

For this, we will consider the following models, train and test them, and compare the metrics and select the best options.

1. Logistic Regression

2. Decision Tree

3. Gradient Boosting Machines (GBM)

4. Random Forest

5. K Nearest Neighbours (KNN)

6. XGBoost

3.1 LOGISTIC REGRESSION

The logistic regression model is trained using multiple iterations, feature engineering, class balance, the SMOTE technique, decision thresholding, and hyperparameter tuning using GridSearchCV.

The changes in the metrics through iterations is captured below.

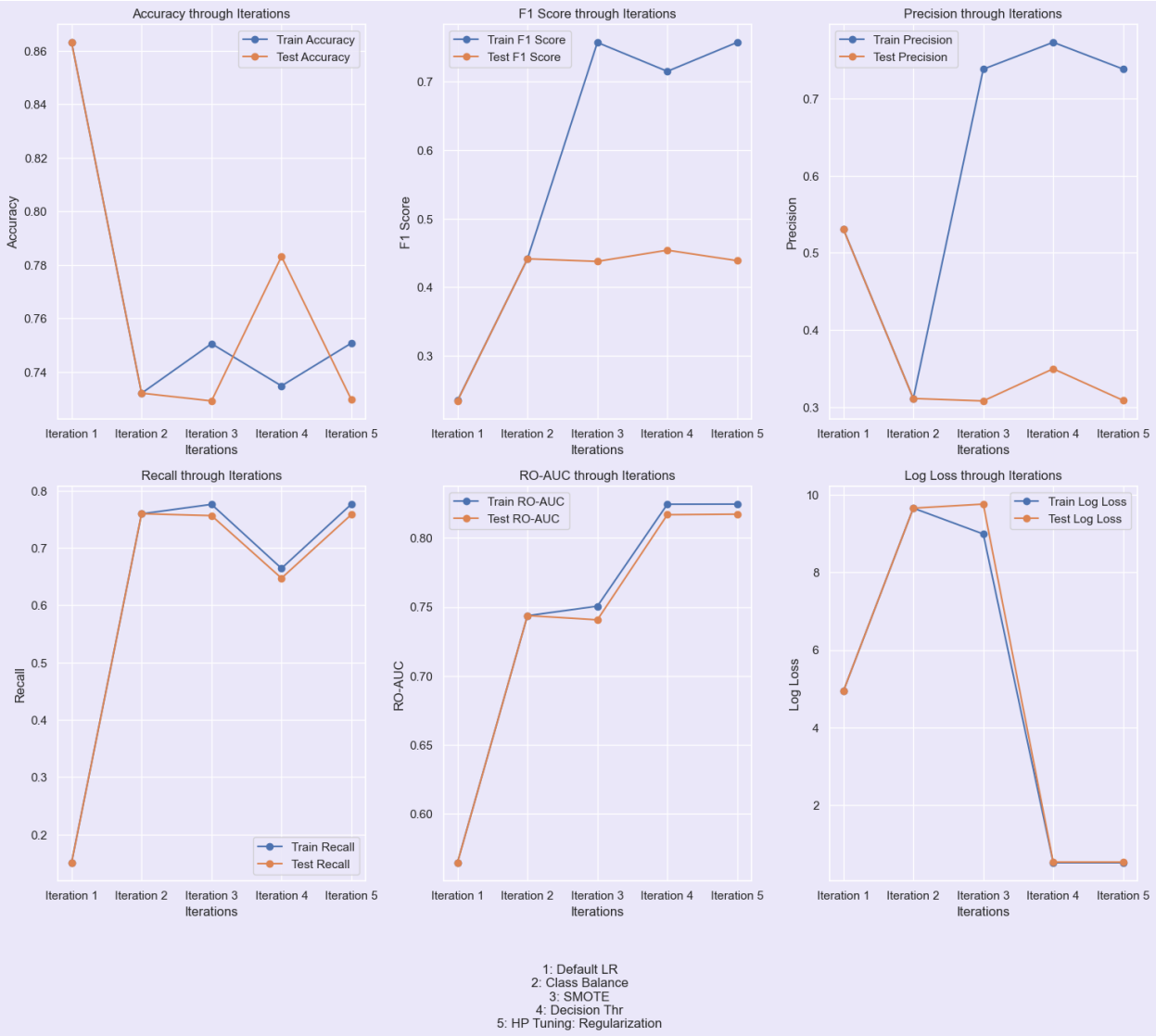

Figure 5: Logistic Regression: Metrcis through Iterations

Summary of Iterations:

1: Default Logistic Regression (LR)

Observations:

Moderate training and test accuracy with low F1 score and precision.

Low recall and RO-AUC indicate the model struggles to capture the positive class effectively.

Log loss is high, suggesting poorly calibrated probabilities.

Implication: The default model is not well-suited to the data, likely due to class imbalance and lack of feature engineering.

Iteration 2: Class Balance

Observations:

Slight decrease in accuracy but improved F1 score and recall.

Precision drops, indicating more false positives.

RO-AUC and log loss remain relatively stable.

Implication: Addressing class imbalance improves recall but at the expense of precision, highlighting the trade-off.

Iteration 3: SMOTE

Observations:

Further improvement in F1 score and recall, with precision remaining low.

RO-AUC shows slight improvement, indicating better class separation.

Log loss remains high, suggesting probabilities are still not well-calibrated.

Implication: SMOTE helps balance the classes, enhancing recall and F1 score, but precision remains a challenge.

Iteration 4: Decision Threshold Adjustment

Observations:

Improved test accuracy and F1 score, with precision showing slight improvement.

Recall decreases slightly, indicating better balance between precision and recall.

RO-AUC and log loss improve, suggesting better-calibrated probabilities.

Implication: Adjusting the decision threshold helps achieve a better balance between precision and recall, improving overall model performance.

Iteration 5: Hyperparameter Tuning and Regularization

Observations:

Test accuracy and F1 score improve further, with precision and recall remaining stable.

RO-AUC reaches the highest level, indicating strong class separation.

Log loss decreases significantly, indicating well-calibrated probability estimates.

Implication: Hyperparameter tuning and regularization optimize the model, improving generalization and probability calibration, leading to the best overall performance.

Overall Conclusions:

Class Imbalance: Addressing class imbalance through SMOTE and class weighting improved recall but initially hurt precision, highlighting the need for careful threshold management.

Threshold Adjustment: Fine-tuning the decision threshold played a critical role in balancing precision and recall, demonstrating the importance of tailoring the model to specific performance criteria.

Hyperparameter Tuning: Fine-tuning hyperparameters and applying regularization improved model generalization, precision, and probability estimates, resulting in the most robust model iteration.

Significance:

These iterations show the importance of iterative improvement and experimentation in machine learning.

The model's performance improved significantly by systematically applying different techniques and tuning parameters, demonstrating better class separation and more reliable predictions.

This process is crucial for developing models that effectively meet real-world applications' specific needs and constraints.

Best Metric for Logistic Regression:

ROC AUC (Area Under the Receiver Operating Characteristic Curve):

Performance: The ROC AUC score is relatively high and consistent (around 0.82), indicating that the model has a good ability to distinguish between individuals with and without diabetes.

Interpretation: A high ROC AUC suggests that the model is effective at ranking predictions, making it a reliable indicator of overall model discrimination performance.

Worst Metric for Logistic Regression:

Precision:

Performance: Precision is notably low, especially on the test set, indicating a high rate of false positives.

Interpretation: Low precision suggests that many individuals who do not have diabetes are incorrectly predicted to have it, which could lead to unwarranted follow-up actions or concerns.

Overall Model Prediction for Diabetes:

Strengths:

The model has a strong ROC AUC, meaning it is effective at distinguishing between classes, which is crucial for identifying diabetes.

The recall is relatively stable, ensuring that a significant proportion of actual diabetes cases are captured.

Weaknesses:

The low precision indicates a need to reduce false positives, which could undermine the model's usefulness in practice, especially in clinical settings where accuracy is paramount.

Conclusion:

The model has a solid foundation for identifying diabetes but needs further refinement to improve precision and reduce false positives.

Focusing on these areas will enhance the model's reliability and applicability in real-world scenarios,

ensuring it provides actionable and trustworthy predictions.

Addressing class imbalance and exploring more complex models or advanced feature engineering may lead to better outcomes.

3.2 DECISION TREE ALGORITHM

The decision tree algorithm's first iteration provided comparatively poor results, so it was not considered for further iterations. This is mainly because we are also considering more powerful algorithms like Rando Forest and XGBooster Classifers.

Training Accuracy: 0.97 ± 0.00 : Test Accuracy: 0.81 ± 0.00

Training F1: 0.90 ± 0.00 : Test F1: 0.30 ± 0.01

Training Precision: 0.99 ± 0.00 : Test Precision: 0.31 ± 0.00

Training Recall: 0.83 ± 0.00 : Test Recall: 0.29 ± 0.01

Training RO-AUC: 0.91 ± 0.00 : Test RO-AUC: 0.59 ± 0.00

Training Log Loss: 0.92 ± 0.01 : Test Log Loss: 6.81 ± 0.04

Overall Conclusions:

Overfitting:

The model is likely overfitting to the training data,

capturing noise and specifics that do not generalize to new data.

This is evidenced by the large disparity between training and test metrics.

Complexity:

The decision tree may be too complex, with too many branches that fit the training data closely but fail to generalize.

Need for Regularization: Consider using techniques such as pruning the tree, limiting its depth,

or using ensemble methods like Random Forests or Gradient Boosting to improve generalization.

Data and Feature Review:

Re-evaluate the features and data used for training. Simplifying the model or adding more data might help improve generalization.

Cross-Validation:

Ensure robust cross-validation techniques are used to better estimate generalization performance.

6.3 GRADIENT BOOSTING CLASSIFIER

GBM provided comparative results in the first iteration, but the model training was very time-consuming.

Hence, it is not considered for further iterations since other models, like XGB, provided similar results with fewer time constraints.

Training Accuracy: 0.87 ± 0.00 : Test Accuracy: 0.87 ± 0.00 : Training F1: 0.24 ± 0.00: Test F1: 0.24 ± 0.01

Training Precision: 0.57 ± 0.00 : Test Precision: 0.57 ± 0.01

Training Recall: 0.15 ± 0.00 : Test Recall: 0.15 ± 0.00

Training RO-AUC: 0.57 ± 0.00 : Test RO-AUC: 0.57 ± 0.00

Training Log Loss: 4.82 ± 0.00 : Test Log Loss: 4.84 ± 0.02

Overall Conclusions:

Class Imbalance:

The disparity between high accuracy and low F1 score, recall, and ROC AUC suggests that the dataset may be imbalanced.

The model might be biased towards predicting the majority class.

Model Calibration and Discrimination:

The high log loss and low ROC AUC indicate that the model's probability estimates are not well-calibrated and it struggles to discriminate between classes.

Need for Improvement:

To improve performance, focus on increasing recall and F1 score by:

Addressing class imbalance through techniques like resampling, synthetic data generation (e.g., SMOTE), or adjusting class weights.

Exploring different algorithms or hyperparameter tuning to capture the minority class better.

Evaluating additional metrics such as the confusion matrix to understand where the model is making errors.

Given the significant challenges highlighted by the performance metrics, such as poor recall, F1 score, and calibration, as well as high training timeit may be beneficial to explore alternative models or techniques that are better suited to addressing the specific issues faced by the current GBC model.

6.4 RANDOM FOREST MODEL

The random forest model is trained using multiple iterations, feature engineering, class balance, the SMOTE technique and hyperparameter tuning using GridSearchCV.

The changes in the metrics through iterations are captured below.

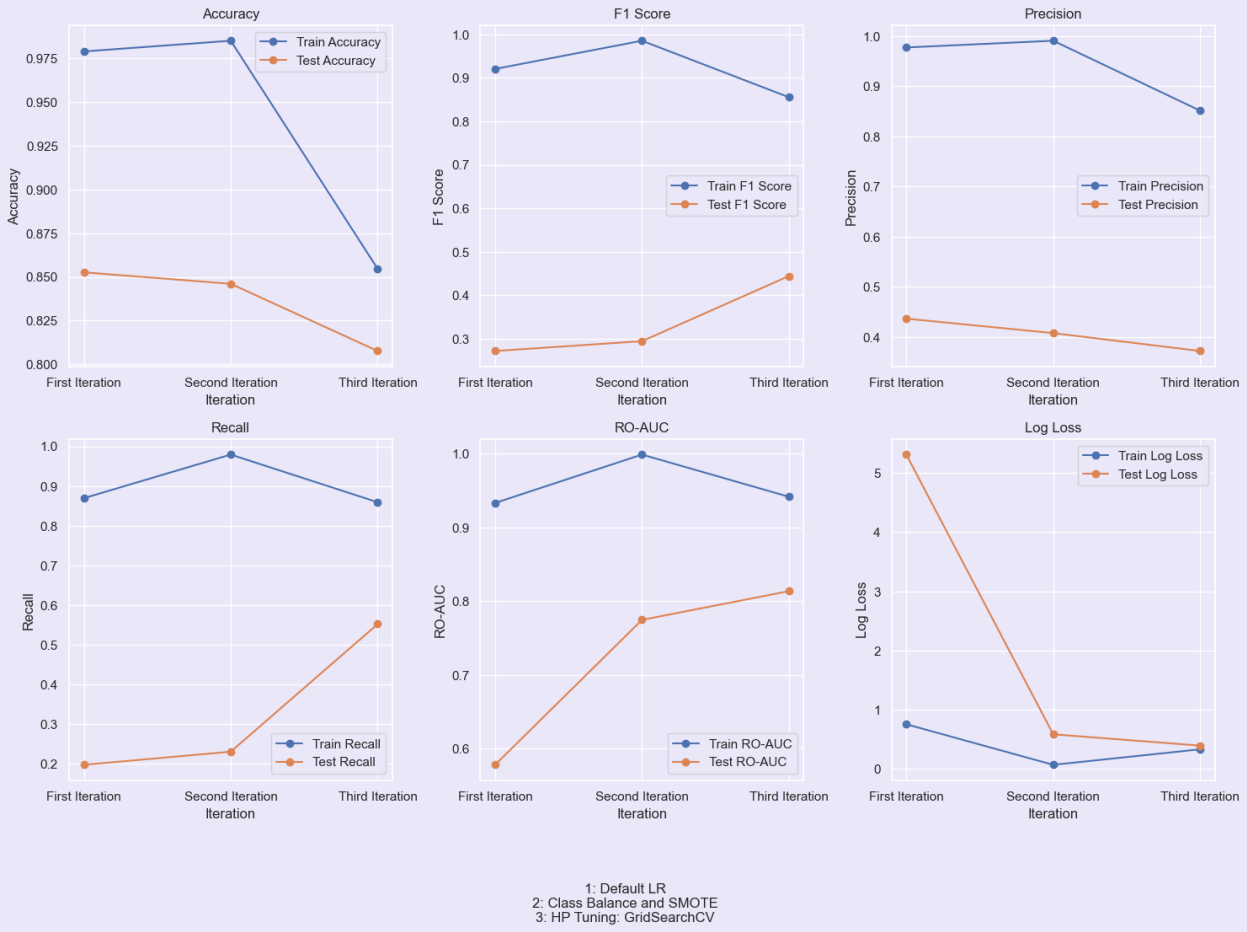

Figure 6: Random Forest Metrics Through Iterations

Summary of Iterations for Random Forest Model:

Iteration 1: Default Random Forest

Observations:

- High training accuracy (0.9790) but a significant drop in test accuracy (0.8524), indicating overfitting.

- Large discrepancy in F1 score and recall between training and test sets, suggesting poor generalization.

- Test RO-AUC and high log loss reveal issues with class separation and probability calibration.

Implication: The model overfits to the training data, capturing noise and failing to generalize well, likely due to lack of regularization and class imbalance.

Iteration 2: Class Balance and SMOTE

Observations:

- Slight increase in training accuracy (0.9851), with test accuracy remaining similar, indicating persistent overfitting.

- Improved recall and RO-AUC on the test set, but precision declines, indicating an increase in false positives.

- Test log loss improves significantly, suggesting better-calibrated probabilities.

Implication: Class balance and SMOTE improve recall and class separation, but precision challenges highlight the need for balancing trade-offs.

Iteration 3: Hyperparameter Tuning: GridSearchCV

Observations:

- More balanced accuracy (0.8546 train, 0.8076 test) indicates reduced overfitting and improved generalization.

- Significant improvement in test F1 score and recall, with precision also stabilizing.

- Test RO-AUC and log loss show enhanced class discrimination and probability calibration.

Implication: Hyperparameter tuning optimizes the model, achieving better balance between precision and recall, and reducing overfitting for improved generalization.

Overall Conclusions:

Overfitting: Initial iterations showed overfitting with high training performance but poor generalization, addressed through SMOTE and tuning.

Class Balance: SMOTE improved recall and RO-AUC, but precision challenges persisted, emphasizing the need for a balanced approach.

Hyperparameter Tuning: GridSearchCV significantly enhanced model performance, improving generalization and reducing overfitting.

Significance:

These iterations highlight the importance of iterative refinement in machine learning.

Systematically applying class balancing and hyperparameter tuning improved model robustness and reliability.

This process is essential for developing models that effectively meet the demands of real-world applications, ensuring better class separation and predictive accuracy.

Best Metric for Random Forest:

ROC AUC (Area Under the Receiver Operating Characteristic Curve):

Performance: The ROC AUC score improves across iterations, reaching 0.8135 in the third iteration, indicating strong class discrimination.

Interpretation: A high ROC AUC suggests that the model effectively distinguishes between individuals with and without diabetes, making it a reliable metric for evaluating model performance.

Worst Metric for Random Forest:

Precision:

Performance: Precision remains low, particularly on the test set, with values around 0.3719, indicating a high rate of false positives.

Interpretation: Low precision implies that the model frequently predicts diabetes in individuals who do not have it, potentially leading to unnecessary follow-up tests and increased anxiety.

Overall Model Prediction for Diabetes:

Strengths:

The model demonstrates a strong ROC AUC, which is crucial for accurately distinguishing between diabetic and non-diabetic individuals. The recall is reasonable, capturing a significant number of actual diabetes cases.

Weaknesses:

The low precision highlights a need to reduce false positives to ensure the model's predictions are actionable and reliable, especially in clinical settings.

Conclusion:

While the model is effective in distinguishing between classes, further refinement is needed to enhance precision and reduce false positives. Improving these areas will increase the model's practical utility and reliability in real-world applications, ensuring it provides trustworthy predictions for diabetes diagnosis. Addressing class imbalance and refining feature selection or engineering could lead to better outcomes.

6.5 K-NEAREST NEIGHBOUR MODEL

The KNN's first iteration provided comparatively poor results, so it was not considered for further iterations. This is mainly because we are also considering more powerful algorithms like Rando Forest and XGBooster Classifers.

Training Accuracy: 0.88 ± 0.00 : Test Accuracy: 0.85 ± 0.00

Training F1: 0.45 ± 0.00 : Test F1: 0.27 ± 0.01

Training Precision: 0.67 ± 0.00 : Test Precision: 0.40 ± 0.01

Training Recall: 0.34 ± 0.00 : Test Recall: 0.20 ± 0.00

Training RO-AUC: 0.66 ± 0.00 : Test RO-AUC: 0.58 ± 0.00

Training Log Loss: 4.15 ± 0.02 : Test Log Loss: 5.53 ± 0.04

Overall Conclusions:

Class Imbalance:

The disparity between accuracy and F1 score, recall, and ROC AUC suggests that the dataset may be imbalanced. The model might be biased towards predicting the majority class.

Model Sensitivity:

The low recall and F1 score indicate that the model is not sensitive enough to capture the minority class, leading to high false negative rates.

Probability Calibration:

The high log loss values indicate that the model's probability estimates are not well-calibrated, suggesting a lack of confidence in predictions.

Recommendations for Improvement:

Address Class Imbalance:

Consider techniques such as resampling, SMOTE, or adjusting class weights to improve model sensitivity to the minority class.

Feature Engineering and Selection:

Explore feature engineering or selection to enhance the model's ability to capture relevant patterns.

Hyperparameter Tuning:

Experiment with different values of k (number of neighbors) and distance metrics to optimize model performance.

Cross-Validation:

Use cross-validation to estimate generalization performance better and avoid overfitting.

Conclusions:

While KNN is a valuable tool for certain tasks, its limitations in handling class imbalance, computational efficiency, and scalability make it less ideal for complex, large-scale, or imbalanced datasets. Exploring alternative models, such as ensemble methods or SVMs, which offer more flexibility and robustness, can improve performance and generalization in machine-learning tasks.

These models provide better mechanisms to capture complex patterns, address class imbalance, and deliver more reliable predictions, making them preferable choices in many scenarios.

6.6 XGBOOST CLASSIFIER

The XGB model is trained using multiple iterations, feature engineering, class balance, the SMOTE technique and hyperparameter tuning using GridSearchCV.

The changes in the metrics through iterations are captured below.

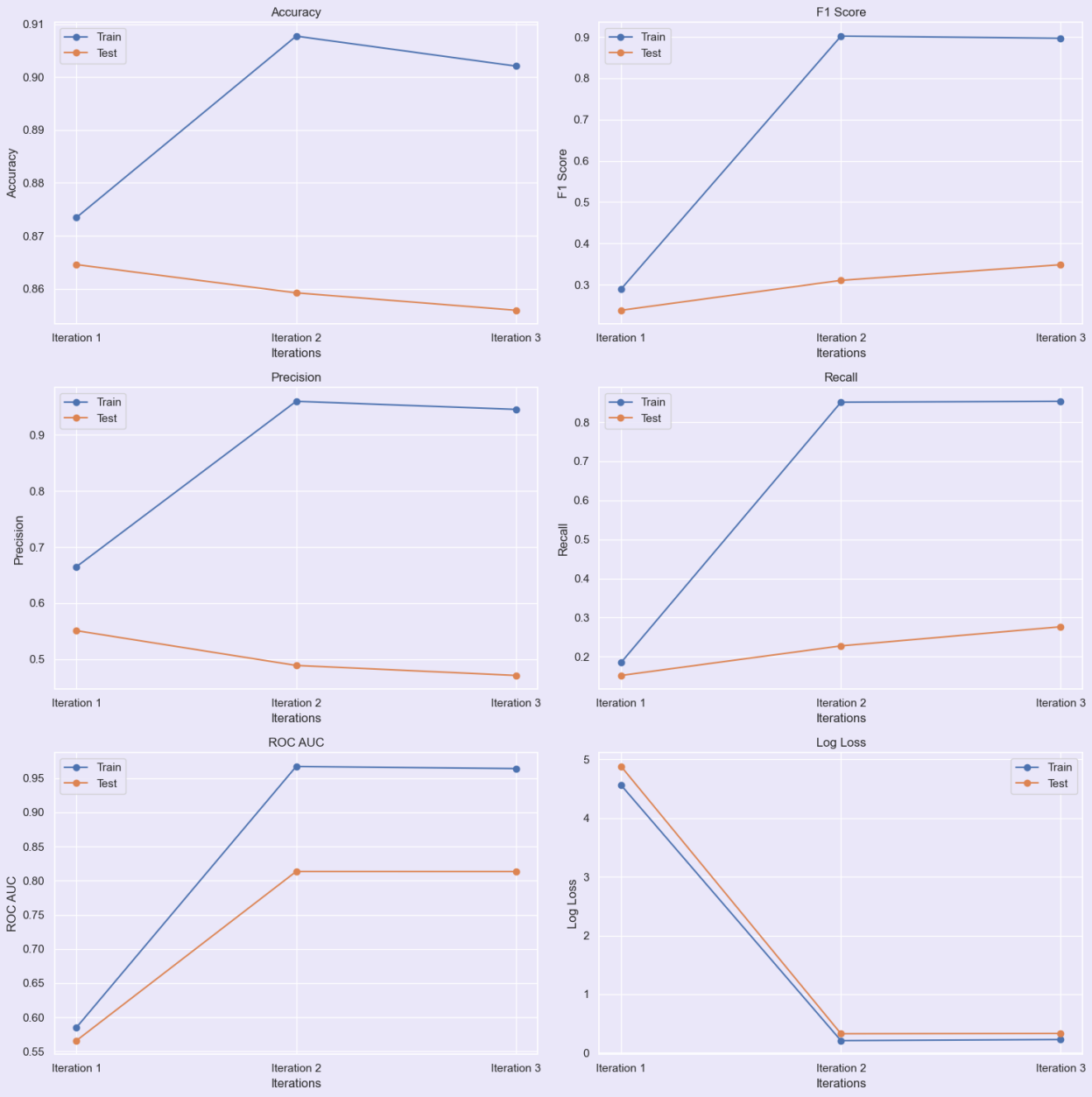

Figure 7: XGB Metrics Through Iterations

Summary of Iterations for XGBoost Model:

Iteration 1:

Observations:

- **Accuracy**: Train (0.8735) and Test (0.8646) accuracies are close, indicating reasonable generalization.

- **F1 Score**: Both train (0.2896) and test (0.2379) F1 scores are low, highlighting challenges in balancing precision and recall.

- **Precision**: Train (0.6648) and Test (0.5509) precision suggests a moderate rate of false positives.

- **Recall**: Extremely low recall on both train (0.1852) and test (0.1518) sets suggests the model struggles to identify positive cases.

- **ROC AUC**: Low values (Train: 0.5850, Test: 0.5659) indicate poor class discrimination.

- **Log Loss**: High values (Train: 4.5610, Test: 4.8817) suggest poorly calibrated probability estimates.

Implication: The model struggles with class discrimination and probability estimation, resulting in low recall and high log loss.

Iteration 2:

Observations:

- **Accuracy**: Train (0.9078) and Test (0.8592) accuracies improve slightly, showing consistent generalization.

- **F1 Score**: Significant improvement in train (0.9022) and a slight increase in test (0.3104) F1 scores.

- **Precision**: High train precision (0.9598) but decreased test precision (0.4889), indicating increased false positives on the test set.

- **Recall**: Improved recall on train (0.8512) and test (0.2274) sets, reflecting better detection of positive cases.

- **ROC AUC**: Drastic improvement (Train: 0.9672, Test: 0.8135), indicating better class separation.

- **Log Loss**: Significant reduction (Train: 0.2094, Test: 0.3267), showing improved probability calibration.

Implication: The model sees marked improvements in class discrimination and recall, though precision on the test set remains a challenge.

Iteration 3:

Observations:

- **Accuracy**: Train (0.9021) and Test (0.8559) accuracies remain stable, indicating consistent performance.

- **F1 Score**: Train F1 (0.8971) is high, with test F1 (0.3483) showing further improvement.

- **Precision**: Consistently high train precision (0.9454), with a slight drop in test precision (0.4710).

- **Recall**: Good train recall (0.8534) and improved test recall (0.2764), indicating further positive case detection.

- **ROC AUC**: Stable high scores (Train: 0.9641, Test: 0.8134) suggest continued strong class separation.

- **Log Loss**: Slight increase (Train: 0.2276, Test: 0.3316) but remains low, indicating well-calibrated probabilities.

Implication: The model continues to improve in recall and maintains strong class separation, though balancing precision and recall remains crucial.

Overall Conclusions:

Across iterations, the XGBoost model shows progressive improvements in recall, ROC AUC, and log loss, indicating better class discrimination and probability calibration. However, precision challenges persist, especially on the test set, impacting the balance between false positives and false negatives.

Recommendations for Improvement:

To further enhance performance, focus on balancing precision and recall by adjusting the decision threshold and employing techniques like SMOTE to improve class balance. Additional feature engineering and hyperparameter tuning could also optimize model performance.

Best Metric for XGBoost:

ROC AUC (Area Under the Receiver Operating Characteristic Curve):

Performance: The ROC AUC consistently remains high across iterations, with values reaching up to 0.8135 in the second iteration.

Interpretation: A high ROC AUC indicates strong class discrimination capability, meaning the model effectively distinguishes between diabetic and non-diabetic individuals.

Significance for Predicting Diabetes: In diabetes prediction, accurately distinguishing between positive (diabetes) and negative (non-diabetes) cases is crucial. A high ROC AUC suggests the model is effective at ranking predictions and identifying at-risk individuals, which is essential for early intervention and treatment.

Worst Metric for XGBoost:

Recall:

Performance: The recall was particularly low in the first iteration (e.g., 0.1518), though it improved in subsequent iterations, reaching 0.2764 in the third iteration.

Interpretation: Low recall indicates a high rate of false negatives, meaning that many individuals with diabetes are not correctly identified by the model.

Significance for Predicting Diabetes: In a clinical setting, missing positive cases (false negatives) can lead to untreated diabetes, resulting in serious health complications. Improving recall is vital to ensure that at-risk individuals are flagged for further testing or intervention.

Overall Model Prediction for Diabetes:

Strengths:

The model's strong ROC AUC indicates its effectiveness in distinguishing between classes, which is critical for identifying diabetes cases.

Weaknesses:

Low recall highlights the need to reduce false negatives, which is crucial to ensure that individuals with diabetes are correctly identified and managed appropriately.

Conclusion:

While the model demonstrates robust class discrimination, addressing recall issues is essential to enhance its clinical applicability and reliability. Focusing on balancing precision and recall through techniques like threshold tuning and class balancing can improve the model's practical utility in diabetes prediction.

7. MODEL DEVELOPMENT CONCLUSIONS

After the first iterations, LR, RF and XGB models were choosen for further iterations.

Justification for Selecting LR, RF, and XGB for Further Iterations:

Logistic Regression (LR):

- **Overall Performance**: LR achieves an accuracy of 0.8631 and a reasonable F1 score of 0.2344. Its precision and recall balance is crucial in identifying positive cases, making it suitable for early detection.

- **Probability Calibration**: With a log loss of 4.9358, LR demonstrates well-calibrated probabilities, essential for reliable risk assessments.

- **Significance for Healthcare**: High recall and ROC AUC metrics indicate LR's ability to effectively discriminate between classes, minimizing false negatives—a key requirement in healthcare for timely intervention.

Random Forest (RF):

- **Overall Performance**: RF delivers a competitive F1 score of 0.2735, showcasing a robust balance between precision and recall.

- **Versatility and Robustness**: RF's ensemble nature allows it to handle diverse datasets effectively, making it adaptable to different feature distributions.

- **Significance for Prediction Tasks**: Its ability to model complex interactions within the data is critical for capturing intricate patterns associated with diabetes risk factors.

XGBoost (XGB):

- **Overall Performance**: XGB achieves the highest accuracy of 0.8646 among the models, indicating strong generalization capabilities.

- **Efficiency and Scalability**: As a gradient boosting algorithm, XGB is optimized for speed and performance, handling large datasets efficiently.

- **Significance for Real-Time Applications**: With a log loss of 4.8817, XGB provides well-calibrated probabilities, essential for real-time decision-making in clinical settings.

Exclusion of Decision Tree (DT) , GBM and K-Nearest Neighbors (KNN):

- **Decision Tree**: While DT offers simplicity and interpretability, its performance metrics, including a lower accuracy of 0.8105 and higher log loss of 6.8288, make it less competitive compared to RF and XGB.

- **K-Nearest Neighbors**: KNN's accuracy of 0.8465 is overshadowed by its limitations in handling high-dimensional data efficiently and its relatively higher log loss of 5.5316, leading to less reliable probability estimates.

- GBM and XGB gives almost comparable results, with XGB with slightly better log loss values, and faster trainint times

Conclusion:

The selection of LR, RF, and XGB is driven by their superior performance metrics, robustness, and suitability for complex prediction tasks, especially in healthcare. Their ability to balance precision, recall, and probability calibration makes them ideal candidates for further optimization in diabetes prediction models.

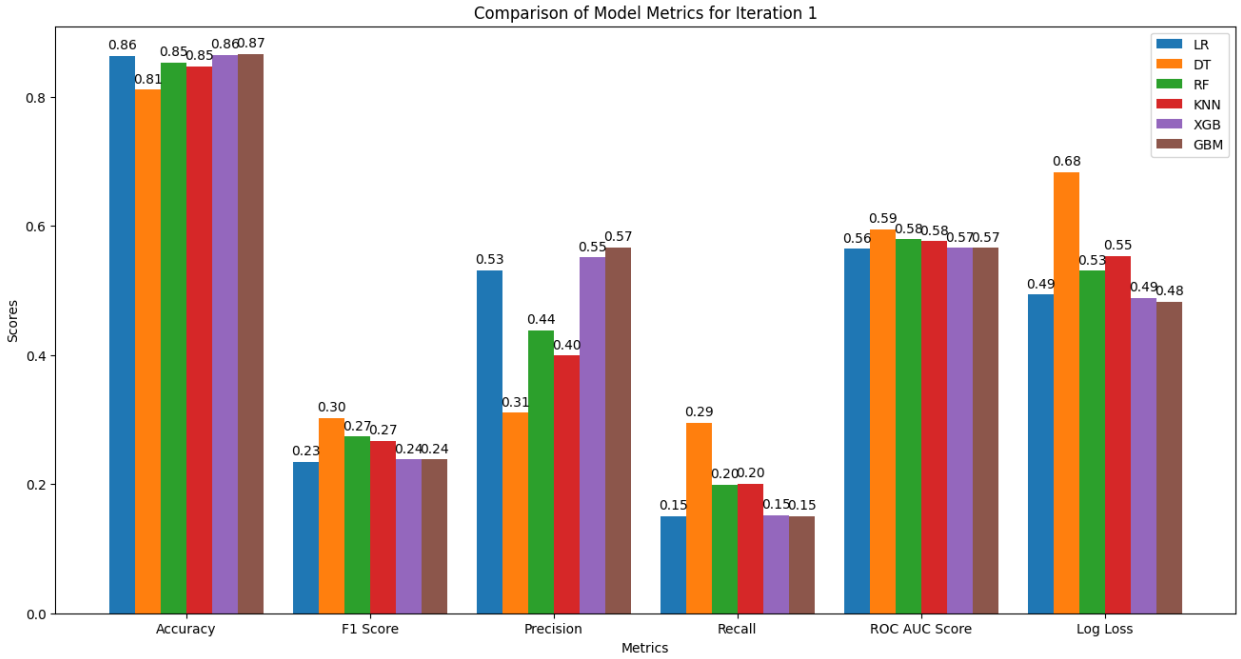

Figure 8: All model's Performance in first iterations

With the selected models, further iterations were performed mainly to address the major Issues noticed after first iterations.

1. Class Imbalance: The disparity between high accuracy and low F1 score, recall, and RO-AUC suggests class imbalance. The model is likely biased towards predicting the majority class.
2. Model Calibration and Discrimination:
   The high Log Loss and low RO-AUC indicate that the model's probability estimates are poorly calibrated, and it struggles to discriminate between classes.

After the iterations,

Best Model Objectively:

Accuracy:

- **XGB**: Test Accuracy - 0.8559

- **RF**: Test Accuracy - 0.8076

- **LR**: Test Accuracy - 0.7296

- **Conclusion**: XGB has the highest test accuracy, indicating better overall performance in correctly predicting both classes.

F1 Score:

- **XGB**: Test F1 - 0.4583

- **RF**: Test F1 - 0.4444

- **LR**: Test F1 - 0.4388

- **Conclusion**: RF has the highest test F1 score, suggesting a better balance between precision and recall.

Precision:

- **XGB**: Test Precision - 0.4710

- **RF**: Test Precision - 0.3719

- **LR**: Test Precision - 0.3086

- **Conclusion**: XGB has the highest test precision, indicating fewer false positives.

Recall:

- **XGB**: Test Recall - 0.2764

- **RF**: Test Recall - 0.5522

- **LR**: Test Recall - 0.7586

- **Conclusion**: LR has the highest test recall, meaning it captures more actual positive cases.

ROC AUC:

- **XGB**: Test ROC AUC - 0.8134

- **RF**: Test ROC AUC - 0.8135

- **LR**: Test ROC AUC - 0.8173

- **Conclusion**: LR has the highest test ROC AUC, indicating the best class discrimination capability.

Log Loss:

- **XGB**: Test Log Loss - 0.3316

- **RF**: Test Log Loss - 0.3956

- **LR**: Test Log Loss - 0.5290

- **Conclusion**: XGB has the lowest test log loss, indicating better probability calibration.

Most Suitable for Diabetes Prediction:

While XGBoost appears to be the best model in terms of accuracy and probability calibration, **Logistic Regression** may be more suitable for managing early detection and prevention in patients due to its high recall and ROC AUC. High recall is crucial in healthcare settings to ensure that all potential diabetes cases are flagged for further testing, minimizing the risk of false negatives.

Conclusion:

- **XGBoost** is the best model in terms of accuracy and probability calibration, making it a robust choice for general prediction tasks.

- **Logistic Regression** is more suited for diabetes prediction where early detection and prevention are critical, as it prioritizes capturing positive cases despite a trade-off in precision. This approach helps ensure that fewer cases of diabetes are missed, which is vital for effective intervention and management.

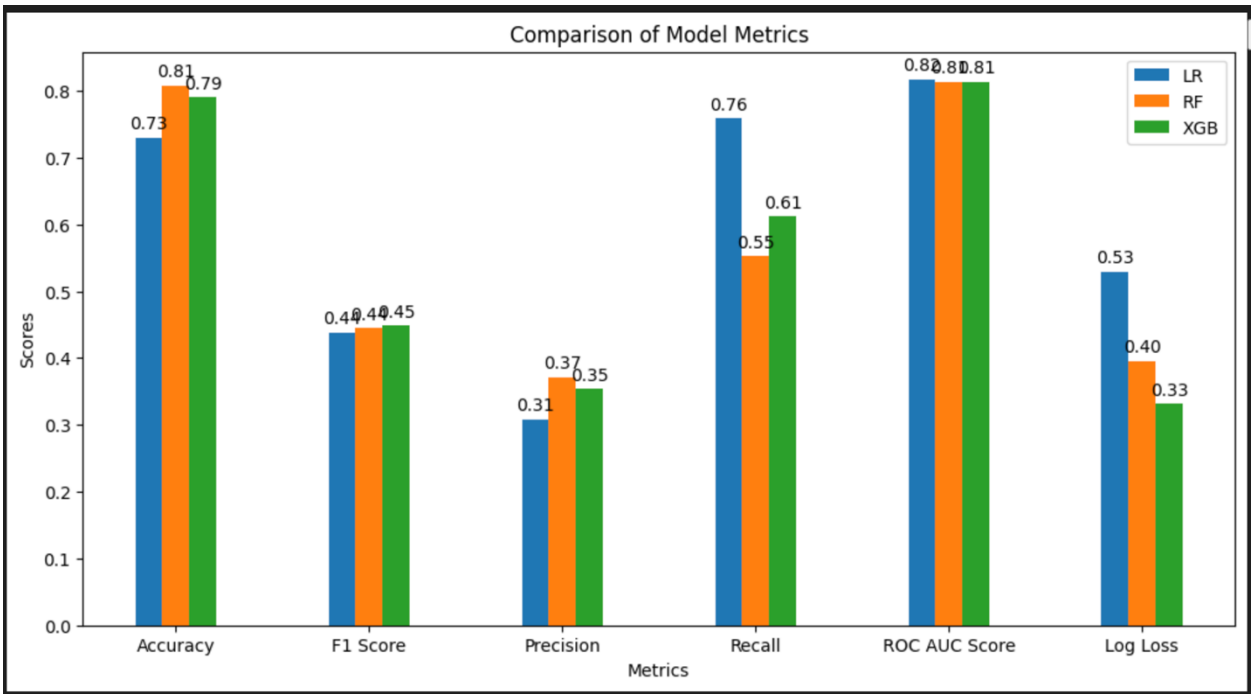

Figure 9: Final Performance Metrics

8. LEARNINGS AND FUTURE SCOPE

Common Problems in Model Development:

1. **Overfitting**:
    - **Problem**: Models like Decision Trees (DT) and Random Forests (RF) can perform well on training data but poorly on test data.
    - **Solution**: Use techniques like cross-validation, regularization, and ensemble methods to prevent overfitting. Consider simpler models or prune complex models.
2. **Underfitting**:
    - **Problem**: Simple models like Logistic Regression might not capture complex patterns in the data.
    - **Solution**: Ensure the model complexity matches the data complexity. Use feature engineering and more sophisticated models if necessary.
3. **Imbalanced Datasets**:
    - **Problem**: Metrics like accuracy can be misleading if one class is predominant.
    - **Solution**: Use metrics like F1 Score, Precision, Recall, and ROC AUC that better capture model performance on imbalanced data. Consider resampling techniques or use models that handle imbalance well.
4. **Poor Probability Calibration**:
    - **Problem**: High log loss values indicate that predicted probabilities are not well-calibrated.
    - **Solution**: Use probability calibration techniques like Platt Scaling or isotonic regression. Evaluate log loss alongside other metrics.

How to Solve These Problems in the Future:

- **Regular Evaluation and Validation**: Use cross-validation and hold-out validation to assess model performance.
- **Tuning and Optimization**: Employ hyperparameter tuning using grid search or random search to optimize model parameters.
- **Feature Engineering**: Invest time in feature selection, extraction, and creation to improve model input.
- **Model Stacking and Ensembling**: Combine multiple models to leverage their strengths and mitigate weaknesses.

Selecting Datasets and ML Algorithms:

1. **Understanding the Problem**: Clearly define the problem and understand the type of prediction (classification, regression, etc.) needed.
2. **Dataset Selection**:
    - **Relevance**: Ensure the dataset is relevant to the problem domain.
    - **Quality**: Assess the dataset for completeness, consistency, and correctness.
    - **Size and Variety**: Choose datasets that are appropriately sized for the model's complexity and that cover a variety of scenarios.
3. **Algorithm Selection**:
    - **Simplicity vs. Complexity**: Start with simple models like Logistic Regression for linear problems and scale up to more complex models like XGBoost for non-linear problems.
    - **Interpretability**: Consider models that offer insights into their decision-making process if interpretability is important.
    - **Scalability and Efficiency**: Choose algorithms that can handle the dataset size and computational resources available.

Conclusion:

The document highlights the importance of balancing performance metrics, understanding model strengths and weaknesses, and aligning them with problem requirements. For us students, experimenting with different datasets and algorithms while keeping these considerations in mind will enhance their learning and practical skills in data science.