# Diabetes Detection Using Machine Learning

## Team Members

Fawad Arabzada ▪ Maryam Timorian ▪ Zabehullah Nasiri

**Model Refinement**

**1. Overview**

The model refinement phase focuses on enhancing the performance of the machine learning model developed to predict diabetes using the Pima Indians Diabetes dataset. By leveraging advanced techniques such as hyperparameter tuning and feature selection, this phase ensures that the model generalizes well and performs optimally.

**2. Model Evaluation**

The initial evaluation of the model revealed areas for improvement. Metrics such as accuracy, precision, recall, and F1-score were analyzed to identify underperforming aspects. The baseline logistic regression model achieved an accuracy of 76%, indicating room for optimization.

**3. Refinement Techniques**

Several refinement techniques were applied to improve the model:

- **Hyperparameter Tuning:** Grid search and random search were employed to optimize parameters such as C and solver settings in logistic regression.
- **Algorithm Comparison:** Random Forest and Gradient Boosting were explored for potential performance improvements.
- **Feature Engineering:** Polynomial features and interaction terms were introduced for better representation of relationships between variables.

**4. Hyperparameter Tuning**

Hyperparameter tuning for logistic regression included:

- Regularization strength (C): Tested values ranged from 0.01 to 10.
- Solver: Compared solvers like liblinear and saga.

The optimal parameters improved the F1-score from 0.74 to 0.78.

**5. Cross-Validation**

K-fold cross-validation was used with k=10 to ensure the model's performance was consistent across different subsets of the data. This reduced the risk of overfitting observed in earlier validation phases.

**6. Feature Selection**

Recursive Feature Elimination (RFE) identified the most significant predictors, including glucose levels, BMI, and age. By focusing on these features, the model became more interpretable and efficient without a noticeable drop in performance.

**Test Submission**

### 1. Overview

The test submission phase evaluated the final refined model on unseen test data to simulate real-world performance and ensure its readiness for deployment.

### 2. Data Preparation for Testing

The test dataset underwent the following preprocessing steps:

- Imputation of missing values using the median.
- Scaling numerical features using StandardScaler.
- Encoding categorical variables with One-Hot Encoding.

### 3. Model Application

The final model was applied to the test dataset. Predictions were generated using the optimized logistic regression model.

Code snippet:

```
# Making predictions on the test data
test_predictions = final_model.predict(test_data)
```

## 4. Test Metrics

On the test dataset:

- **Accuracy:** 77.5%
- **Precision:** 0.79
- **Recall:** 0.74
- **F1-Score:** 0.76

These results align closely with the validation metrics, confirming the model's reliability.

## 5. Model Deployment

The trained model was deployed as a web application using Flask. The app allows users to input data and receive real-time predictions about diabetes risk. Integration with a front-end interface made the tool accessible to non-technical users.

## 6. Code Implementation

**Model Refinement Code:**

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression

# Hyperparameter tuning
param_grid = {'C': [0.01, 0.1, 1, 10], 'solver': ['liblinear', 'saga']}
grid_search = GridSearchCV(LogisticRegression(), param_grid, cv=10)
grid_search.fit(X_train, y_train)
best_model = grid_search.best_estimator_
```

**Test Submission Code:**

```
# Predictions and evaluation
y_test_pred = best_model.predict(X_test)
accuracy = accuracy_score(y_test, y_test_pred)
print(f"Test Accuracy: {accuracy}")
```

# Conclusion

The model refinement phase enhanced the predictive power and stability of the diabetes prediction model. Deploying the model as a web application showcased its practical utility, bridging the gap between machine learning and healthcare.

## References


- Pima Indians Diabetes Database: Kaggle
- Machine Learning Best Practices

- **Python Libraries**:
    - Scikit-learn Documentation: *https://scikit-learn.org/stable/documentation.html*
    - Pandas Documentation: *https://pandas.pydata.org/docs/*
- **Online Tutorials**:
- Logistic Regression Overview: *https://towardsdatascience.com/logistic-regression-explained-and-implemented-in-python-8809557c28e7*
- **Evaluation Metrics**:
- Accuracy Score and Classification Report Documentation: *https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html* *https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html*