Predicting Diabetes Using Machine Learning: A Comparative Analysis of Classification Algorithms

(Development Phase)

Table of Contents

Abstract

In this project, the researcher has been using machine learning techniques to predict diabetes, which is a chronic condition with an increased global prevalence. The secondary data collection method is used to collect the Pima Indians Diabetes Dataset, which has various clinical and demographic variables such as glucose levels, BMI, age, and family history. The major agenda is to determine the supervised learning models' effectiveness in developing early diagnosis. The reseracher has been using ***"Logistic Regression, Random Forest, and Gradient Boosting models"***, which were developed with the help of Python and scikit-learn. The performance measure metrics, such as ***"accuracy, precision, recall, F1-score, and ROC-AUC"***, are to be used to underscore the model performance and also use cross-validation to determine the best model. The unit testing has helped to define whether the developed pipeline works correctly or not, and end-to-end workflow validation helps to confirm that the pipeline is accurate for developing predictions.

1. Introduction

Diabetes mellitus is a persistent metabolic disorder that is increasing at an alarming rate all over the world. The World Health Organisation states that over 422 million individuals are already living with diabetes, and this figure is expected to rise considerably in the next few decades. Cardiovascular disease, kidney failure, blindness and amputation of lower limbs are some of the severe complications linked to diabetes, necessitating early diagnosis and treatment to enhance patient outcomes and minimise the cost of healthcare. The research shows that traditional diagnostic tools, including fasting blood glucose and oral glucose tolerance tests, are effective but might be resource-intensive, invasive, and may not give any information besides the binary response to the question of whether a patient is diabetic or not. The development of computational techniques and data-driven solutions gives the possibility to boost clinical decision-making. Using machine learning, we can analyse a vast amount of patient data and determine patterns and risk factors (Tasin et al., 2023). It is underscored from the research that a predictive model has the ability to help healthcare professionals in screening individuals who are high risk, as well as to develop preventive interventions before major complications develop. The main agenda of developing this project is to develop ***"supervised machine learning algorithms"*** with the help of the ***"Pima Indians Diabetes Dataset",*** *a*s well as build logistic regression as a baseline and advanced classifiers such as random forests and gradient boosting. The objective is to determine whether computational methods can reliably assist in the early detection of diabetes and also to assess the possibility of applying them in real-world healthcare processes.

2. Related work

This section is established to critically analyse the published journal articles from Google Scholar and underscore the major technological evaluation of machine learning in predicting diabetes, and discover major research gaps. It is underscored from the research that machine learning is a coherent tool for healthcare and giving the ability to analyse critical datasets as well as help clinicians in developing early disease detection. The critical analysis of journals shows that various author uses the Pima Indians Diabetes Dataset for developing the classification models and benchmark. Although significant advancements have occurred, the literature shows the existence of recurrent conflicts between predictive performance, interpretability and clinical applicability. The study developed by Naz and Ahuja (2020) uses deep learning methods for predicting diabetes with the help of the Pima dataset, as well as results of the study show cogent accuracy in comparison to the traditional model. The author's work shows the coherent capability of neural networks to determine nonlinear patterns in medical data. Also, this study has some limitations, such as being dependent on the Pima dataset decreases external validity. Also, deep learning methods were considered as black

box models, and it was not discussed much about the interpretability to clinicians, which decreases the chances of use in practice.

Also, it is underscored from the research that the machine learning model has cogent capability to accelerate generalizability. Tasin et al. (2023) used several machine learning methods on a real clinical dataset that was gathered in Bangladesh, focusing on accuracy, as well as recall and ROC-AUC. Their work revealed the relevance of recall, specifically in the case of healthcare, where false negatives are especially risky. Although they demonstrated a cogent methodology, but clinical workflow integration was not discussed properly, as well as raising questions about how these systems can be used in day-to-day practice. The other study also shows the efficiency of ensemble methods for diabetes prediction. The study developed by Abnoosian et al. (2023) shows the effectiveness of feature selection and stacked ensembles to gather high accuracy on the Pima dataset. The literature delivers cogent insights on how the study can be methodologically enhanced, mainly through the application of Boruta in feature selection. Also, overfitting and generalizability were concerns due to the use of a small benchmark set and the lack of cross-population testing.

The author Li et al. (2024) show a coherent method for diabetes prediction, such as merging genetic algorithms for hyperparameter optimisation with XGBoost stacking. The critical analysis of the literature shows that hybrid modelling shows coherent performance, which highlights the importance of systematic optimisation methods. Also, the complexity introduced to the algorithm is a possible threat to transparency and reproducibility. It is underscored that the model achieves high performance, but without explainability, making it hard to justify in a clinical environment. Hoyos et al. (2024), on the contrary, directly discussed the problem of explainability in diabetes prediction. Their research integrated statistical analysis with AI techniques and used the tools of interpretability, including ranking features of importance, to confirm that the outputs would be comprehensible to the clinicians. Their approach has helped to eliminate the gaps in technical performance and clinical trust. Also, their data were still small and narrow, implying that they should be expanded with larger, multi-site validation. Collectively, these studies show the development of diabetes prediction research. Older experiments have focused on accuracy, especially on the Pima dataset, but more recent works focus on more advanced optimisation and explainability. Also, the limitations remain: excessive dependence on benchmark data, a lack of focus on workflow integration, and a lack of attention to ethical and bias problems. Our research aims to fill these gaps by balancing performance and interpretability. Here, we compare Logistic Regression, Random Forest, and Gradient Boosting. It aims to not only assess such metrics of prediction as accuracy and AUC but also recall, which is essential in medical diagnosis. Moreover, ROC curves and cross-validation are used to determine the best model. The unit testing has helped to define whether

the developed pipeline works correctly or not, and end-to-end workflow validation helps to confirm that the pipeline is accurate for developing predictions.

3. Technical Background

In this project, the researcher has used a supervised classification method, such as the model that learn a mapping from patient features X, which are glucose, BMI, to a binary label Y, which states diabetes.

Here, the sigmoid function has been used in Logistic Regression models. The equation outlined below,

$$"P(y=1| X)= (1/1+e^{-(\beta 0+\beta 1x1+...\beta nxn)})"$$

Also, the ensemble of decision trees, such as Random Forest, is trained on the diabetes dataset as well as uses random feature selection to boost accuracy and decrease overfitting. The Gradient Boosting sequentially develop the trees, and every tree corrects the errors of the previous one and decreases loss with the help of gradient descent. To evaluate the model performance, various kinds of performance measure metrics have been used, which are "accuracy, precision, recall, F1-score, and ROC-AUC".

Accuracy

- *"(TP+TN)/(TP+TN+ FP+FN)"*

Precision and recall

- *"(TP)/(TP+FP), (TP)/(TP+FN)"*

F1-score and ROC & AUC

- It is the harmonic mean of Precision and Recall.
- Also, the ROC curve and AUC measure discrimination ability.

4. Method

4.1 Dataset

The methodology of the project was developed to perform a cogent evaluation of machine learning models to predict diabetes on the Pima Indians Diabetes Dataset. The dataset has 768 records of patients, which are characterised by eight numeric health variables and a binary outcome variable that shows whether the patient has diabetes. Such a structure renders the task appropriate to a supervised binary classification problem.

4.2 Preprocessing

The critical analysis shows that the dataset has various missing values, which are replaced with the help of median imputation to confirm that the model was not biased with biologically implausible values. The features were then scaled with a z-score to normalise features to a similar scale, which is important when using a model that is sensitive to the magnitude of the features, as is the case with Logistic Regression. Also, the data had a moderate class imbalance, whereby there were fewer positive cases of diabetes corresponded to negative

cases. To eliminate this problem, stratified sampling has been used in the process of splitting and model evaluation.

4.3 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) was performed to obtain information about the distributions of features and their relationships. The descriptive statistics gave a generalised idea about the range of the variables, whereas the visuals, like the histograms, displayed that the features like insulin and BMI had skewed distributions. The correlation heatmap was used to determine the possibility of multicollinearity, and it is underscored that glucose and BMI were coherent predictors of diabetes outcome.

4.4 Workflow

The researcher uses a cogent pipeline, which has helped to load the dataset, develop preprocessing, as well as use a stratified train–test split (80:20), model training, cross-validation, model evaluation on the test set and uses unit testing to underscore the efficiency of the workflow. The reseracher has been using Logistic Regression as a baseline, as well as Random Forest as a bagging-based ensemble, and Gradient Boosting as a boosting-based ensemble.

4.5 Tools

The researcher has been using the Jupyter notebook environment to develop the implementation and testing. Also, uses Python libraries such as Pandas and NumPy for data handling, matplotlib and seaborn for visualisation, such as EDA, ROC curves, scikit-learn helps in model training and evaluation as well as pytest is used for unit testing. This methodological structure made it possible to conduct a systematic and reproducible study of the performance of machine learning models in predicting diabetes and preserving clinical interpretability by feature analysis.

5. Implementation

In this section, the researcher will show the implementation process via screenshots, and a detailed explanation will be developed in the final phase.

5.1 Data loading and overview

```
[2]: df = pd.read_csv("diabetes (1).csv")
     df.head()
```

[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
[3]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Figure 1: Dataset loading and overview

```
[4]: df.describe()
```

[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

```
[5]: # Check missing or zero values
     (df == 0).sum()
```

```
[5]: Pregnancies                 111
     Glucose                     5
     BloodPressure              35
     SkinThickness             227
     Insulin                   374
     BMI                        11
     DiabetesPedigreeFunction    0
     Age                         0
     Outcome                   500
     dtype: int64
```

Figure 2: Descriptive statistics and checking missing values

It can be underscored from Figure 1 that the researcher uses the Panda library to load the Pima Indians Diabetes Dataset, which has 768 observations and 9 attributes. Also, it is underscored from Figure 2 that the dataset has missing values, which need to be eliminated to maintain data consistency.

5.2 Exploratory data analysis

The exploratory data analysis is developed to underscore the major insights from the dataset. This graphical analysis helps to determine patterns, trends as well and distribution.

```
[6]: # Glucose Distribution by Outcome
plt.figure(figsize=(8,5))
sns.histplot(data=df, x="Glucose", hue="Outcome", bins=30, kde=True, element="step")
plt.title("Glucose Distribution by Outcome")
plt.show()
```

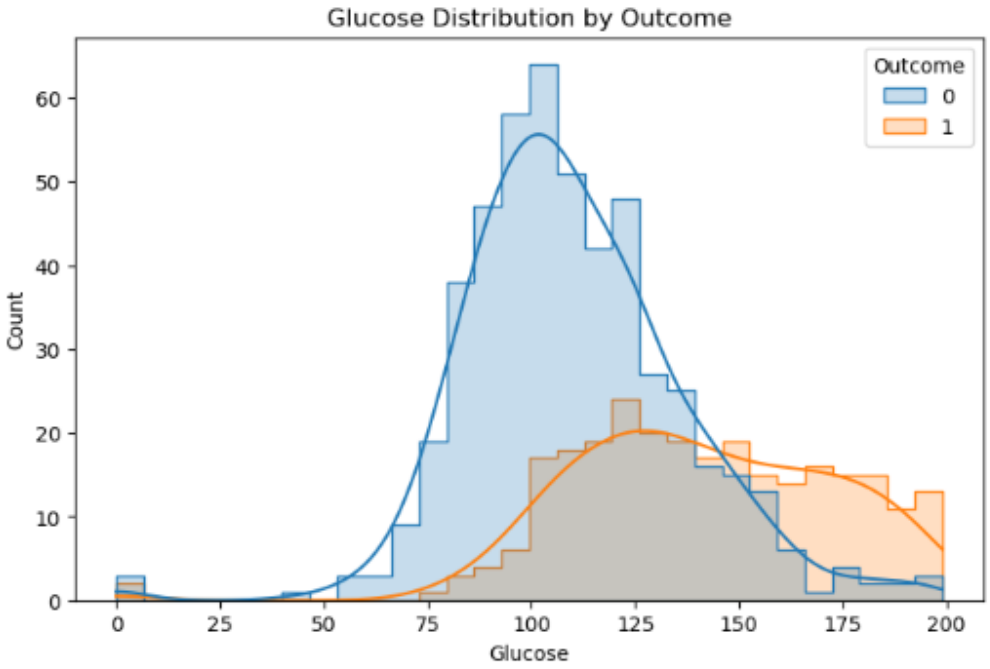

Figure 3: Glucose Distribution by Outcome

It has been underlined from Figure 3 that the histogram shows the distribution of the levels of glucose in diabetic (Outcome = 1) and non-diabetic (Outcome = 0) patients. The values of non-diabetic people are concentrated nearer to lower glucose levels (≈ 100), whereas those of diabetic patients are higher, showing an average and more distributed (>125). It means that glucose is a powerful predictor to identify diabetes.

```
[7]: # BMI Distribution by Outcome
plt.figure(figsize=(8,5))
sns.histplot(data=df, x="BMI", hue="Outcome", bins=30, kde=True, element="step")
plt.title("BMI Distribution by Outcome")
plt.show()
```

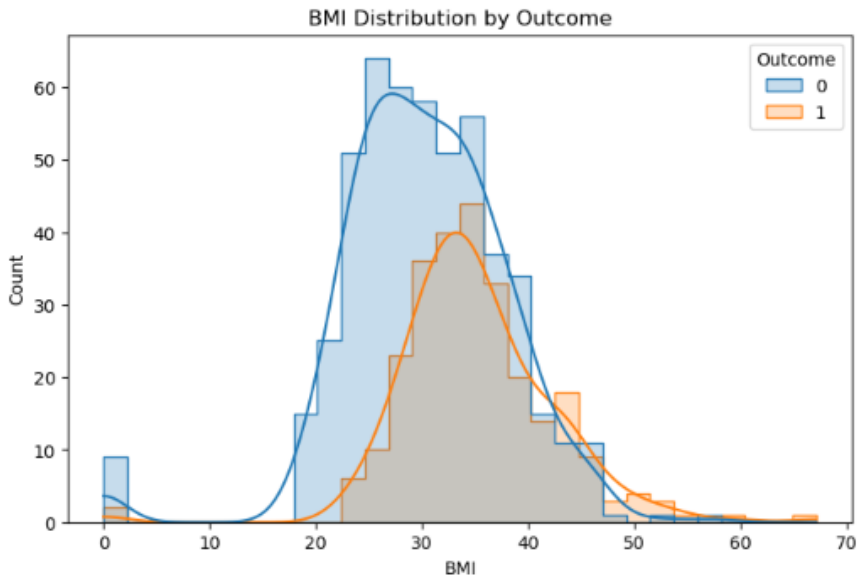

Figure 4: BMI Distribution by Outcome

The graphical analysis shows BMI distribution, which underscores that diabetic patients (Outcome = 1) have high BMI as well as non-diabetic patients have lower BMI. The non-diabetics are clustered around 25 BMI, with diabetics centred above 30+. This means that high BMI is a significant variable that correlates with diabetes.

```
[8]: # Age Distribution by Outcome
plt.figure(figsize=(8,5))
sns.kdeplot(data=df, x="Age", hue="Outcome", fill=True)
plt.title("Age Distribution by Diabetes Outcome")
plt.show()
```

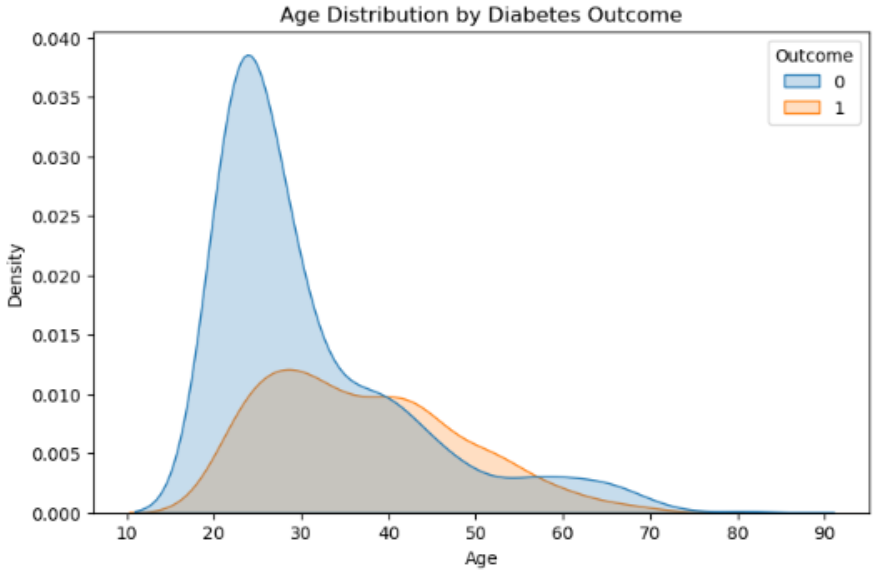

Figure 5: Age Distribution by Outcome

It has been underscored from the age distribution that younger age individuals do not have diabetes, and people who are aged 35+ show they suffer from diabetes. This analysis states that age is a major factor, such as diabetes risk rising as age rises.

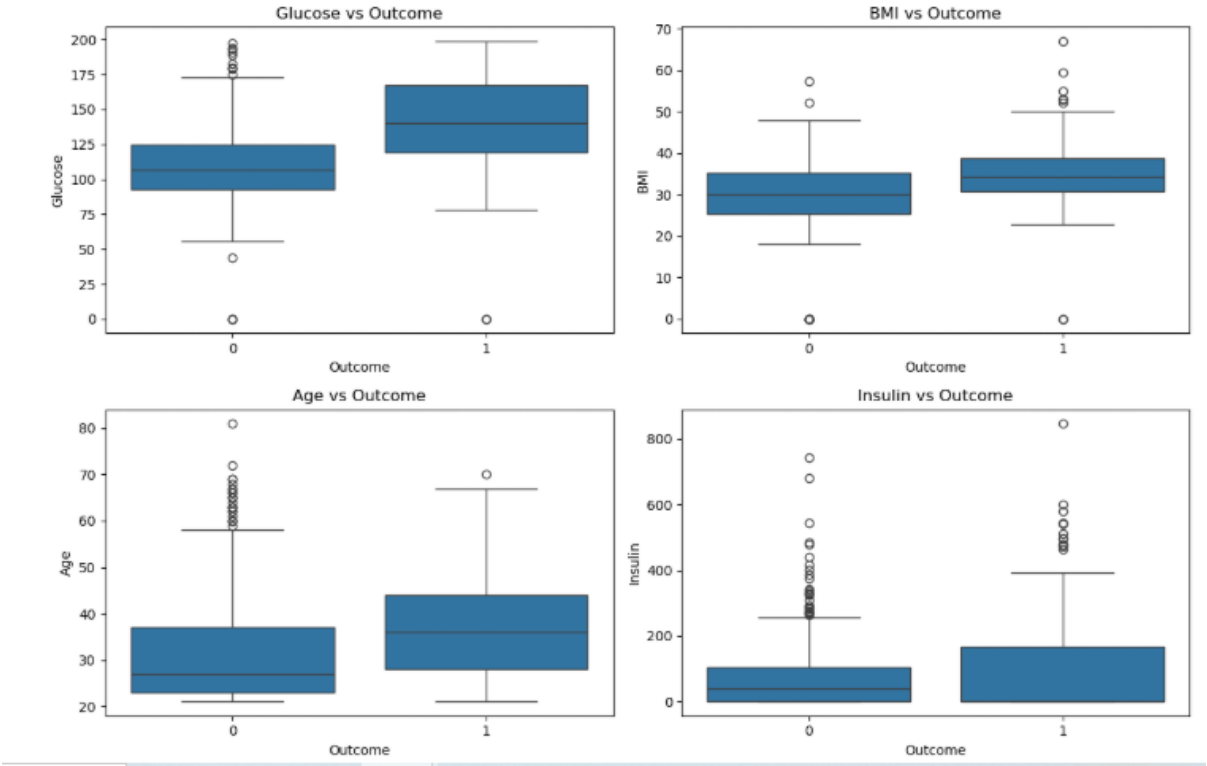

Figure 6: Boxplots for Key Features

```
[10]: # Correlation Heatmap
corr = df.corr()
plt.figure(figsize=(8,6))
sns.heatmap(corr[['Outcome']].sort_values(by='Outcome', ascending=False), annot=True, cmap="coolwarm", vmin=-1, vmax=1)
plt.title("Feature Correlation with Diabetes Outcome")
plt.show()
```

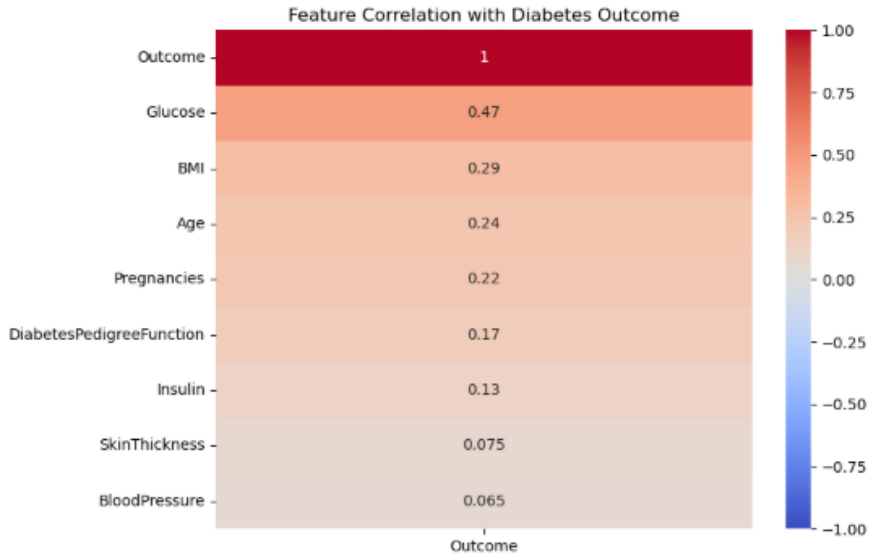

Figure 7: Feature Correlation with Diabetes Outcome

The above figure (7) shows the correlation heatmap and states that glucose has a positive correlation (0.47) with diabetes outcome, as well as other are BMI, age, and number of pregnancies. Features such as blood pressure and skin thickness show weak relationships. This states that glucose and BMI are effective predictors in determining diabetes.

5.3 Data Preprocessing

```
[12]: # Replace 0 with NaN for some columns
cols_with_zero = ['Glucose','BloodPressure','SkinThickness','Insulin','BMI']
df[cols_with_zero] = df[cols_with_zero].replace(0, np.nan)

[13]: imputer = SimpleImputer(strategy="median")
df[cols_with_zero] = imputer.fit_transform(df[cols_with_zero])

[14]: df.isnull().sum()

[14]: Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64

[15]: # Split features/target
X = df.drop('Outcome', axis=1)
y = df['Outcome']

[16]: # Preprocessing pipeline
preprocessor = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

X_processed = preprocessor.fit_transform(X)
```

Figure 8: Data Preprocessing

Here, the researcher uses cogent preprocessing to prepare the dataset for ML modelling. The missing values are replaced with NAN and imputed with the help of median values to maintain cogent distributions. For maintaining consistency, the pipeline is developed with imputation and scaling with all features. This method is very beneficial for the Logistic Regression algorithm. The output shows that the dataset is clean, normalised, has no missing values and is suitable for training machine learning models.

5.4 Data splitting and model training

Train/Test Split

```
[17]: X_train, X_test, y_train, y_test = train_test_split(
          X_processed, y, test_size=0.2, stratify=y, random_state=42
      )
```

Model Training

```
[18]: models = {
          "Logistic Regression": LogisticRegression(max_iter=1000, random_state=42),
          "Random Forest": RandomForestClassifier(n_estimators=200, random_state=42),
          "Gradient Boosting": GradientBoostingClassifier(n_estimators=200, random_state=42)
      }

      results = {}

      for name, model in models.items():
          model.fit(X_train, y_train)
          preds = model.predict(X_test)
          proba = model.predict_proba(X_test)[:,1]

          acc = accuracy_score(y_test, preds)
          roc = roc_auc_score(y_test, proba)

          print(f"### {name} ###")
          print("Accuracy:", acc)
          print("ROC-AUC:", roc)
          print(classification_report(y_test, preds))

          results[name] = {"model": model, "accuracy": acc, "roc_auc": roc}
```

Figure 9: Data splitting and model training

Figure 9 shows that the researcher develops three classifiers in the preprocessed dataset as well as the models are "Logistic Regression, Random Forest, and Gradient Boosting". The dataset is also split into an 80:20 ratio and uses a stratified split to confirm balanced class distribution. It is also determined from the above figure that the model will be evaluated with the help of accuracy, ROC-AUC and classification metrics.

6. Testing

This section is established to test the model performance and perform unit testing.

6.1 Model evaluation and cross-validation

```
### Logistic Regression ###
Accuracy: 0.7012987012987013
ROC-AUC: 0.8127777777777777
              precision    recall  f1-score   support

           0       0.75      0.81      0.78       100
           1       0.59      0.50      0.54        54

    accuracy                           0.70       154
   macro avg       0.67      0.66      0.66       154
weighted avg       0.69      0.70      0.70       154

### Random Forest ###
Accuracy: 0.7402597402597403
ROC-AUC: 0.8173148148148148
              precision    recall  f1-score   support

           0       0.78      0.84      0.81       100
           1       0.65      0.56      0.60        54

    accuracy                           0.74       154
   macro avg       0.71      0.70      0.70       154
weighted avg       0.73      0.74      0.73       154

### Gradient Boosting ###
Accuracy: 0.7597402597402597
ROC-AUC: 0.8251851851851851
              precision    recall  f1-score   support

           0       0.81      0.83      0.82       100
           1       0.67      0.63      0.65        54

    accuracy                           0.76       154
   macro avg       0.74      0.73      0.73       154
weighted avg       0.76      0.76      0.76       154
```

Figure 10: Model evaluation of Logistic Regression, Random Forest, and Gradient Boosting

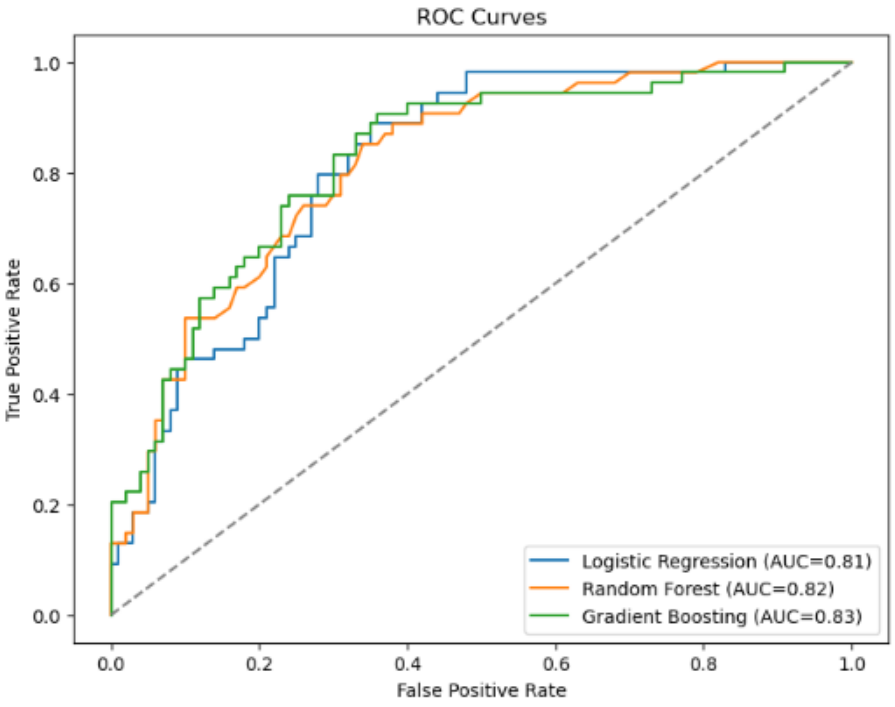

Figure 11: ROC curves

The model evaluation and comparison show that Gradient Boosting outperforms all models in terms of the highest accuracy, such as 76% and ROC-AUC (0.83). The model shows balanced performance to predict both classes. The Random Forest model also shows cogent performance, such as 0.74% accuracy and ROC-AUC (0.82), but can not outperform Gradient

Boosting. As a baseline, logistic regression shows moderate performance, but its is recommended that management of healthcare use the Gradient Boosting model to accurately predict diabetes and gather data-driven decisions.

```
[20]: cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

for name, info in results.items():
    scores = cross_val_score(info['model'], X_processed, y, cv=cv, scoring='roc_auc')
    print(f"{name} CV ROC-AUC Mean:", scores.mean())
```

```
Logistic Regression CV ROC-AUC Mean: 0.8366268343815513
Random Forest CV ROC-AUC Mean: 0.8261194968553459
Gradient Boosting CV ROC-AUC Mean: 0.8251243885394828
```

Save Best Model

```
[21]: import joblib

best_model = max(results.items(), key=lambda x: x[1]['roc_auc'])[1]['model']
joblib.dump(best_model, "best_model.joblib")
print("Best model saved as best_model.joblib")
```

```
Best model saved as best_model.joblib
```

Figure 12: Cross-Validation

It has been underlined from the cross-validation that the Logistic Regression model achieves the highest mean ROC-AUC (≈ 0.84), and others are close to the LR model, such as the boosting model and the ensemble model achieve ≈ 0.83. Also, the best model has been saved using joblib, which confirms reproducibility and its use in the future to perform diabetes prediction exercises.

6.2 Unit testing

```
[22]: # Test 1: Preprocessing replaces zeros with NaN
df_test = pd.DataFrame({
    "Pregnancies":[1,2],
    "Glucose":[0,150],
    "BloodPressure":[0,80],
    "SkinThickness":[0,20],
    "Insulin":[0,90],
    "BMI":[0,32],
    "DiabetesPedigreeFunction":[0.3,0.5],
    "Age":[25,40],
    "Outcome":[0,1]
})

df_test_replaced = df_test.copy()
cols_with_zero = ['Glucose','BloodPressure','SkinThickness','Insulin','BMI']
df_test_replaced[cols_with_zero] = df_test_replaced[cols_with_zero].replace(0, np.nan)

assert df_test_replaced.isnull().sum().sum() > 0, "Zeros were not replaced with NaN!"

print(" Test 1 passed: preprocessing replaces zeros correctly.")
```

```
Test 1 passed: preprocessing replaces zeros correctly.
```

Figure 13: Test 1: Preprocessing replaces zeros with NaN

```
[23]: # Test 2: Preprocessing pipeline output shape
pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

X_test = df_test_replaced.drop("Outcome", axis=1)
X_trans = pipeline.fit_transform(X_test)
assert X_trans.shape[1] == X_test.shape[1], "Shape mismatch after preprocessing!"

print("Test 2 passed: preprocessing pipeline keeps correct feature shape.")
```

```
Test 2 passed: preprocessing pipeline keeps correct feature shape.
```

Figure 14: Test 2: Preprocessing pipeline output shape

```
[24]: #  Test 3: Model can fit and predict
y_test = df_test_replaced["Outcome"]

model = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler()),
    ('clf', LogisticRegression(max_iter=500))
])

model.fit(X_test, y_test)
preds = model.predict(X_test)

assert len(preds) == len(y_test), "Prediction length mismatch!"
print(" Test 3 passed: model can train and predict.")
```

```
Test 3 passed: model can train and predict.
```

Figure 15: Test 3: Model can fit and predict

The preprocessing and modelling steps are trustworthy, as verified in the unit tests. Test 1 confirmed the presence of zeros that were substituted with NaNs in an accurate manner. The fact that the pipeline maintained feature dimensions following scaling and imputation was tested and confirmed in test 2. Test 3 validated the model in the ability to train and give predictions. Such tests suggest a coherent, reproducible workflow that is fit to be used in clinical data applications.

6.3 Workflow Validation (End-to-End Check)

```
[25]: def full_workflow_check(df):
          """
          Mini end-to-end test of workflow: preprocess -> split -> train -> predict
          """
          from sklearn.model_selection import train_test_split

          X = df.drop("Outcome", axis=1)
          y = df["Outcome"]

          pipeline = Pipeline([
              ('imputer', SimpleImputer(strategy='median')),
              ('scaler', StandardScaler()),
              ('clf', LogisticRegression(max_iter=500))
          ])

          X_train, X_test, y_train, y_test = train_test_split(
              X, y, test_size=0.2, random_state=42, stratify=y
          )

          pipeline.fit(X_train, y_train)
          acc = pipeline.score(X_test, y_test)
          return acc

      # Run on main dataset
      acc = full_workflow_check(df)
      print(f" End-to-end workflow test passed. Accuracy: {acc:.3f}")

       End-to-end workflow test passed. Accuracy: 0.708
```

Figure 16: Workflow Validation

It is noted from the end-to-end workflow validation that the pipeline runs accurately from preprocessing through model training and finally to prediction, which gives an accuracy of 0.708. This workflow validation states the accurate functionality of the developed software, as well as confirming that there is an appropriate integration of the parts into real practice in diabetes prediction activities.

7. Conclusion

It can be concluded that this research showed that machine learning has the ability to effectively help early diabetes detection. It is underscored from the model performance that Logistic Regression uses as a baseline, as well as ensemble methods such as Random Forest and Gradient Boosting, gathered coherent predictive power, also Gradient Boosting performed best and outperforms all models. Cross-validation and workflow validation confirmed trustworthiness. Also, due to the size of the datasets, the work demonstrates the potential of the computational tools to speed up the process of clinical decision-making and highlights the necessity of recall in medical predictions.

References

Naz, H. and Ahuja, S., 2020. Deep learning approach for diabetes prediction using PIMA Indian dataset. *Journal of Diabetes & Metabolic Disorders*, *19*(1), pp.391-403.

Tasin, I., Nabil, T.U., Islam, S. and Khan, R., 2023. Diabetes prediction using machine learning and explainable AI techniques. *Healthcare technology letters*, *10*(1-2), pp.1-10.

Abnoosian, K., Farnoosh, R. and Behzadi, M.H., 2023. Prediction of diabetes disease using an ensemble of machine learning multi-classifier models. *BMC bioinformatics*, *24*(1), p.337.

Li, W., Peng, Y. and Peng, K., 2024. Diabetes prediction model based on GA-XGBoost and stacking ensemble algorithm. *PloS one*, *19*(9), p.e0311222.

Hoyos, W., Hoyos, K., Ruiz, R. and Aguilar, J., 2024. An explainable analysis of diabetes mellitus using statistical and artificial intelligence techniques. *BMC medical informatics and decision making*, *24*(1), p.383.