Team No. : 109 **VIT Chennai Campus**

Team Members:

1. **Aimee Grace Ronny - 20BEC1267**
2. **Archana Jayakrishnan - 20BEC1216**
3. **Nayanthara P S - 20BEC1175**

Diabetes Prediction Using Machine Learning

1 INTRODUCTION

1.1 Overview

The aim of the project is early detection and prediction of diabetes using machine learning algorithms and to develop a predictive model that can accurately identify individuals at high risk of developing diabetes based on their health records and other relevant factors.

1.2 Purpose

Early detection and management of diabetes can improve healthcare outcomes, reduce costs, and benefit healthcare providers and insurance companies. Since it can lead to more personalised healthcare, improving patient outcomes and adherence to treatment plans .Thus developing an accurate and reliable predictive model for diabetes detection can have a significant impact on healthcare outcomes and costs.

2 LITERATURE SURVEY

2.1 Existing problem

"Diabetes Prediction using Machine Learning Algorithms" by Aishwarya Mujumdar et al.This paper reviews several commonly used machine learning algorithms, including decision trees, support vector machines (SVM), logistic regression, random forests, and artificial neural networks, for diabetes prediction. They discuss the strengths and limitations of each algorithm, considering factors such as handling imbalanced datasets, feature selection, and model interpretability.They addresses challenges related to data quality and missing values in diabetes prediction and discuss techniques to handle these challenges effectively, ensuring the reliability of the predictive models

"'A Review on Current Advances in Machine Learning-Based Diabetes Prediction" by Varun Jaiswal et al. This paper addresses the challenges associated with real-world implementation, including data quality, model interpretability, and deployment in healthcare settings. They highlight the need for robust validation and standardisation of machine learning models for reliable and scalable diabetes prediction.

2.2 Proposed solution

The proposed solution for diabetes prediction using machine learning algorithms addresses several key challenges. Firstly, it tackles the issue of imbalanced datasets commonly encountered in medical datasets. Techniques such as oversampling, undersampling, or SMOTE can be employed to balance the dataset and ensure that minority class samples are adequately represented. Secondly, feature selection plays a crucial role in identifying the most relevant features and reducing dimensionality. Methods like correlation analysis and feature importance analysis can be utilised to select the most informative features, improving model performance and interpretability.

Moreover, ensuring data quality and reliability is essential for accurate diabetes prediction. It is crucial to address data accuracy, completeness, and consistency, and robust validation techniques like cross-validation or bootstrapping should be employed to assess model performance and generalizability. Standardisation of machine learning models and their implementation should also be considered to ensure consistency and scalability in real-world applications.

3 THEORETICAL ANALYSIS

3.1 Block diagram

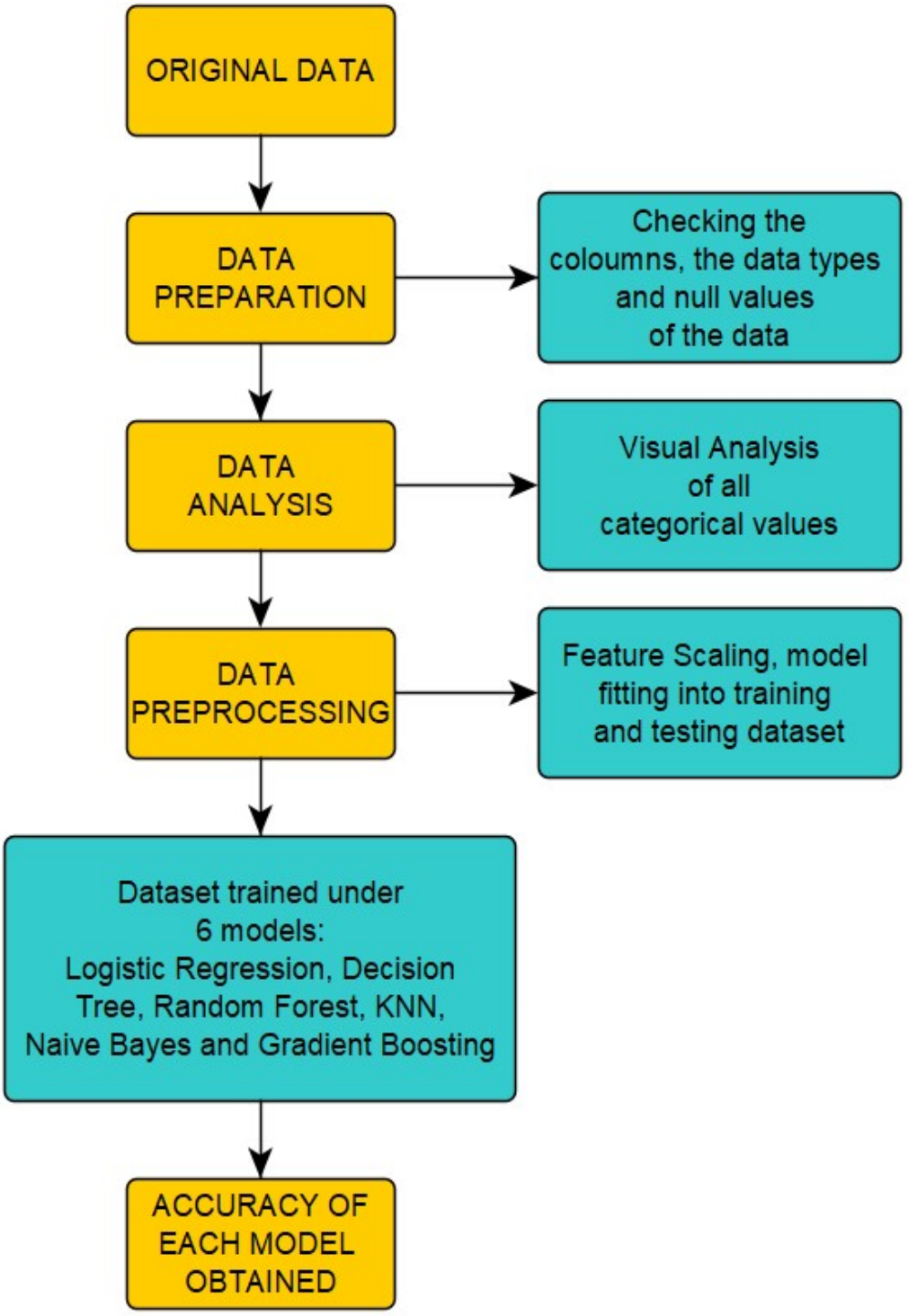
3.2 Hardware / Software designing

- Google Colab
- Flask

4 EXPERIMENTAL INVESTIGATIONS

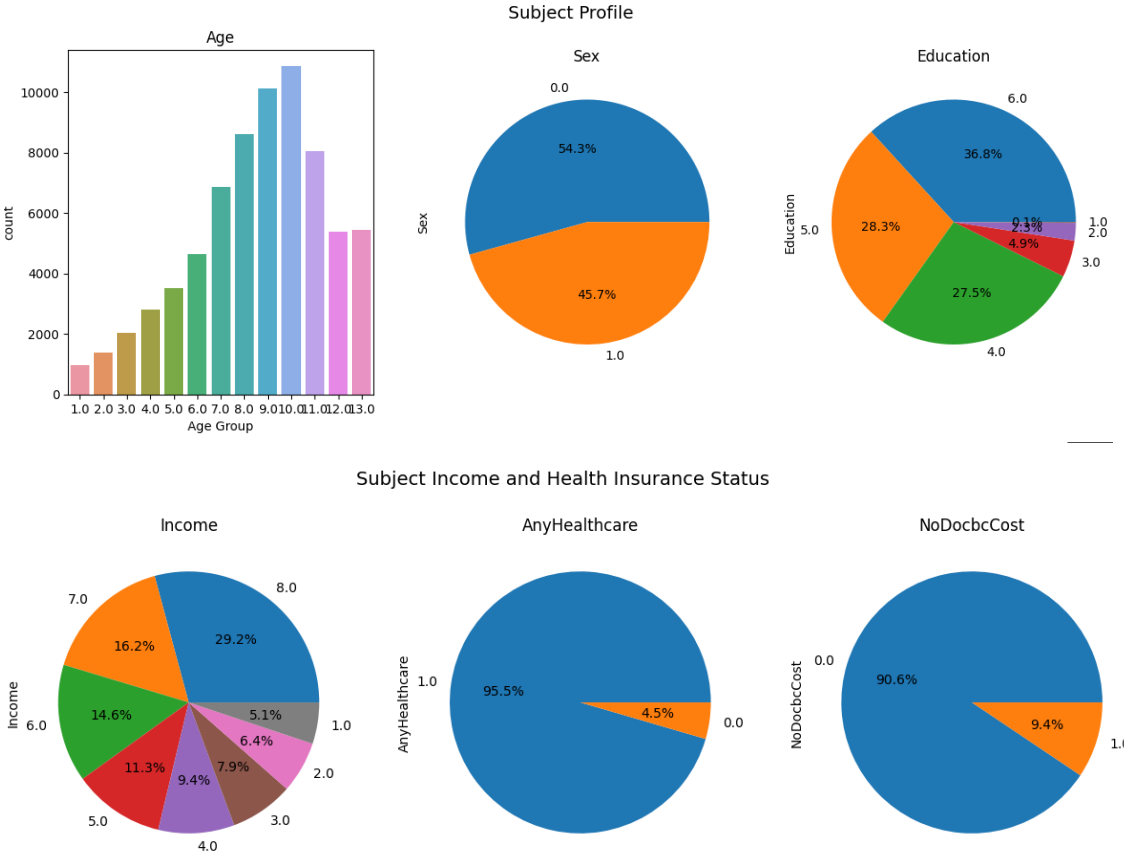
- The code defines a function evaluate_model() to evaluate the performance of different machine learning models.
- Several classification models (Logistic Regression, Decision Tree, Random Forest, KNN, Naive Bayes, Gradient Boosting) are trained and evaluated using the function.
- The results of each model, including accuracy, precision, recall, F1-score, and ROC AUC score, are stored in a dataframe and displayed.
- The Gradient Boosting model performs the best based on the validation accuracy and ROC AUC score.
- GridSearchCV is used to tune the hyperparameters of the Gradient Boosting model.
- The grid of hyperparameters to search over is defined in param_grid.
- The best parameters found by GridSearchCV are printed.
- The model is re-initialized with the best parameters and evaluated again.

5 FLOWCHART

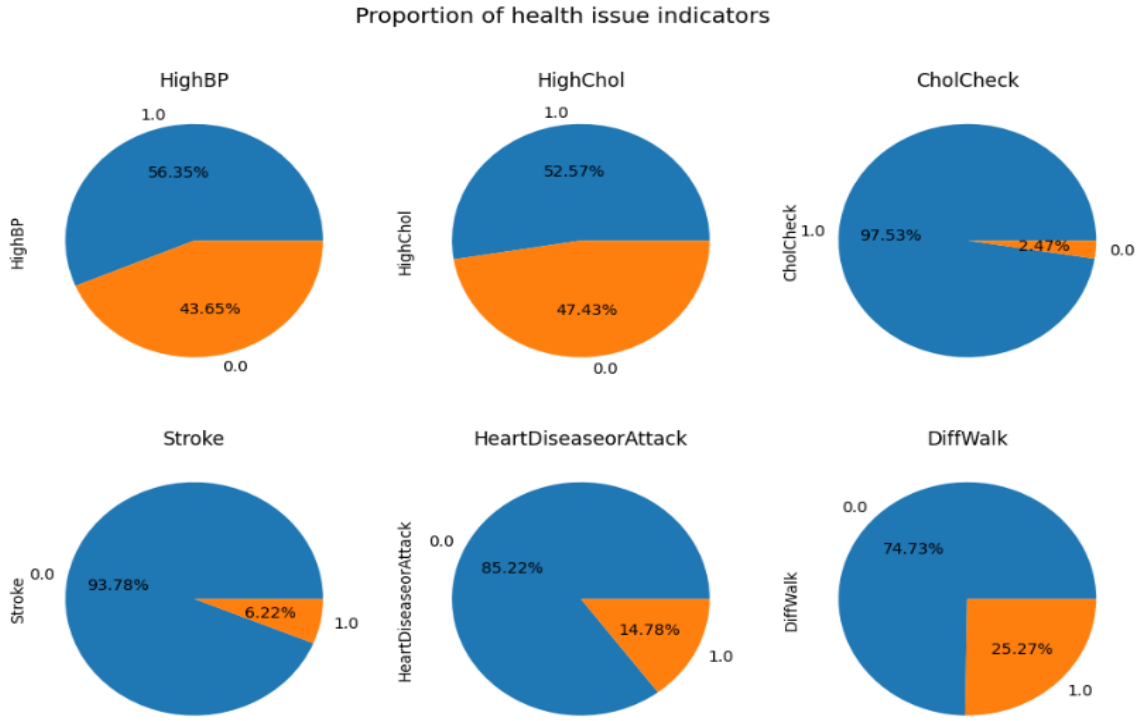

ORIGINAL DATA
DATA PREPARATION
Checking the coloumns, the data types and null values of the data
DATA ANALYSIS
Visual Analysis of all categorical values
DATA PREPROCESSING
Feature Scaling, model fitting into training and testing dataset
Dataset trained under 6 models: Logistic Regression, Decision Tree, Random Forest, KNN, Naive Bayes and Gradient Boosting
ACCURACY OF EACH MODEL OBTAINED

6 RESULT

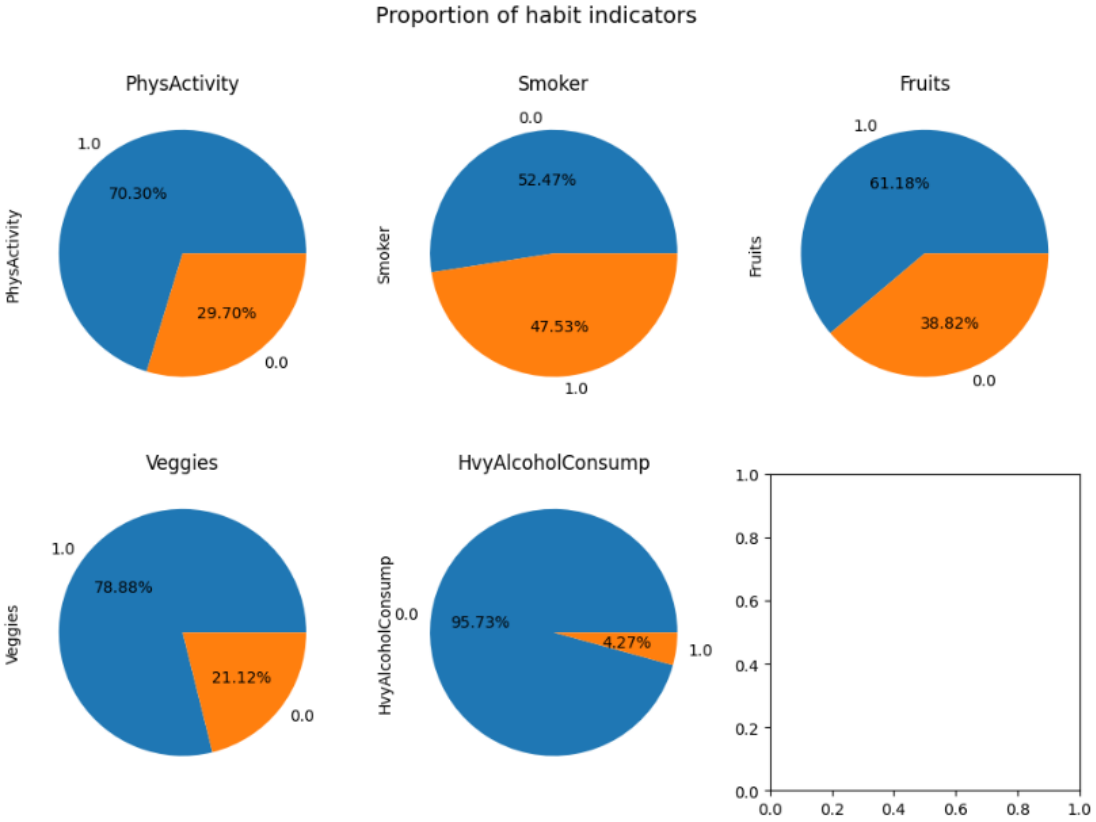
1. Dataset distribution based on particular variables



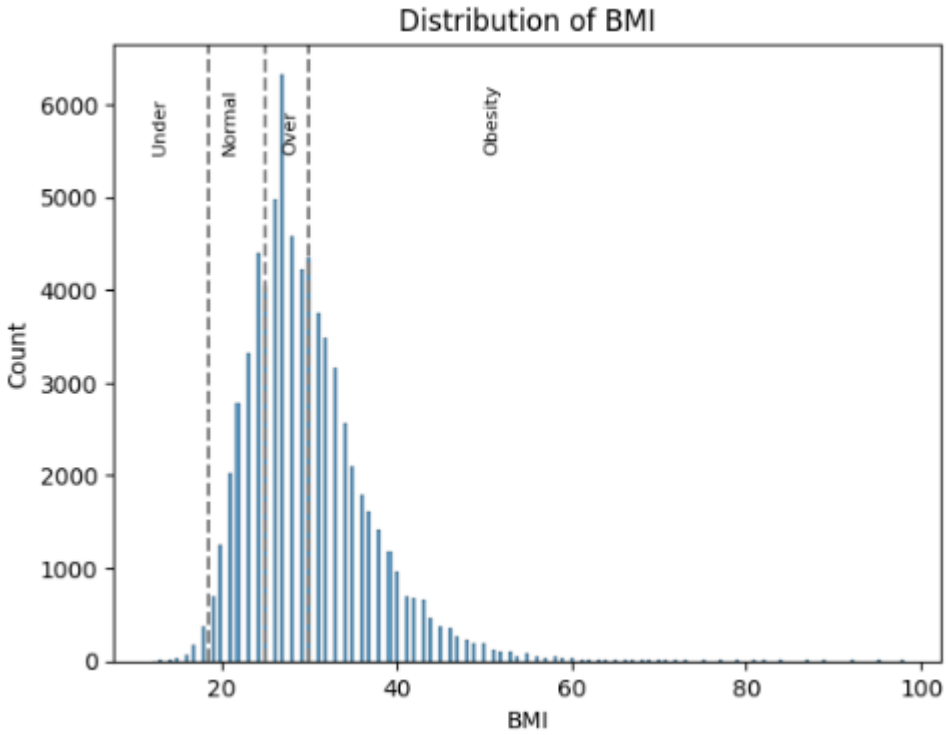
2. Proportion oh health issue indicators



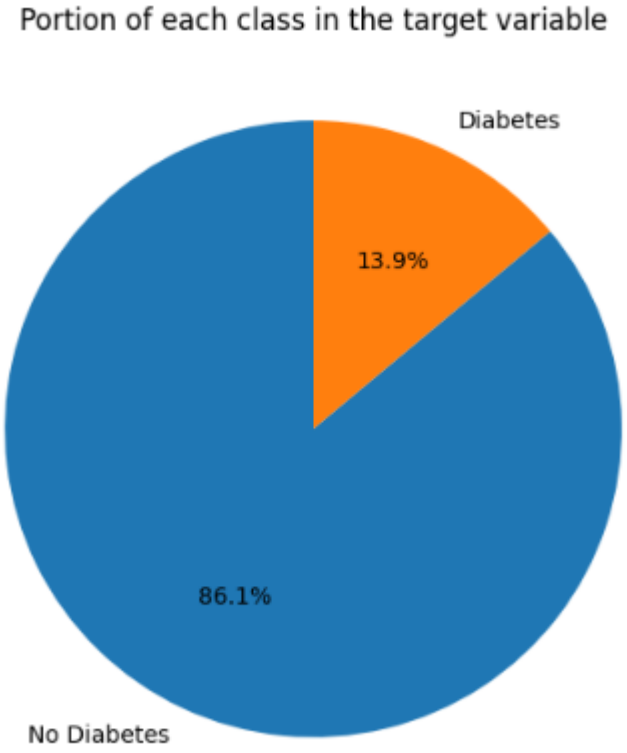
3. Proportion of habit indicators



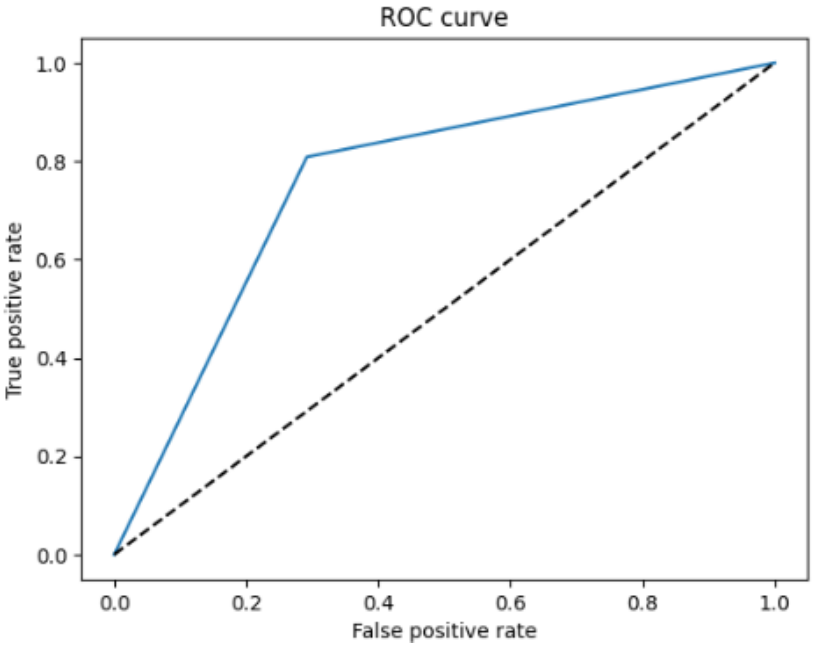
4. Distribution of BMI



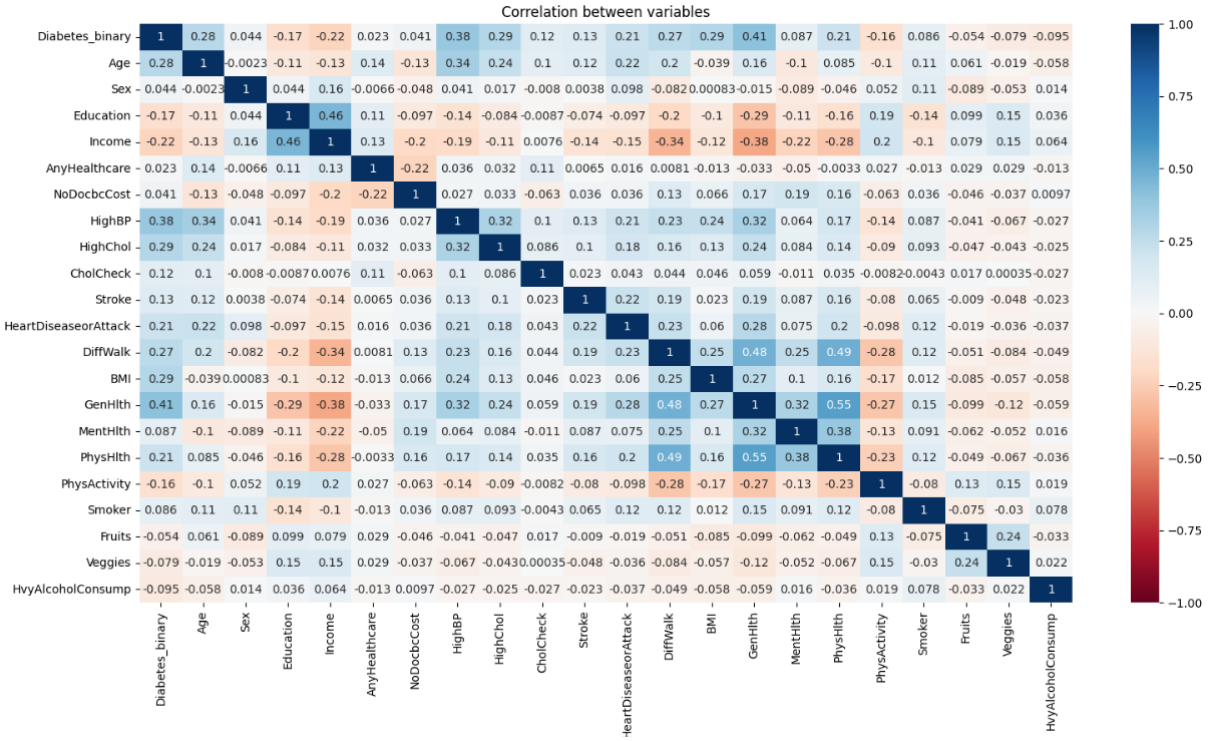
5. **Portion of each class Diabetes and No Diabetes**



6. **ROC curve**



7. Correlation between the variables



Inference from the results:

It seems that Diabetes looks positively correlated with age and negatively correlated with income.The physical measures High BP, high BMI, high Cholesterol are correlated with having diabetes. Among the health habits, whether having physical activity in the past 30 days is the most correlated with diabetes (in a negative way).

Thus by training a Gradient Boosting model, we can predict diabetes with 75.5% accuracy score after model selection and hyperparameter tuning.

7 ADVANTAGES & DISADVANTAGES

7.1 Advantages

1. **Improved accuracy:** It can analyse complex patterns and relationships within diabetes-related data, leading to improved prediction accuracy compared to traditional diagnostic methods.
2. **Early detection:** It can identify subtle patterns in data that may indicate early signs of diabetes and thus help prevent or manage the condition effectively.
3. **Personalised predictions**: It can analyse individual patient data, including medical history, lifestyle factors, and genetic information, to provide personalised predictions and tailored recommendations for diabetes management.
4. **Handling large datasets**: it allows for the incorporation of a wide range of variables and factors that influence diabetes prediction.
5. **Real-time monitoring:** It can be integrated with wearable devices and other monitoring systems to continuously track patient data and provide real-time predictions.

7.2 Disadvantages

1. **Data requirements:** It requires large and high-quality datasets to train accurate models. Acquiring and curating such datasets, especially in healthcare settings, can be challenging due to issues like data privacy, data availability, and data quality.
2. **Generalisation limitations:** It may struggle to generalise well to diverse populations or when faced with data from different healthcare settings. Models trained on specific patient populations or data sources may not perform as effectively when applied to different populations or settings.
3. **Imbalanced data:** Imbalanced datasets can lead to biased predictions. Ensuring balanced data representation and appropriate handling of class imbalance is crucial for accurate predictions.
4. **Model complexity and expertise:**. Adequate resources, skills, and collaborations are necessary to build and maintain robust and reliable models.

8 APPLICATIONS

1. **Early Detection and Diagnosis:** It can be used as a screening tool to identify individuals at risk of diabetes and thus enable timely interventions and lifestyle modifications to prevent or manage the condition effectively.
2. **Personalised Treatment Plans:** It can help healthcare professionals create personalised treatment plans for individuals diagnosed with diabetes.
3. **Remote Monitoring and Telemedicine:**.Patients can regularly input their health indicators, and the model can provide real-time risk assessments and feedback. This allows healthcare providers to remotely monitor patients with diabetes, detect any concerning patterns or changes, and intervene promptly when necessary.
4. **Preventive Interventions:** Public health organisations and policymakers can identify high-risk populations and implement targeted preventive interventions. Thus for individuals of higher risk of diabetes in a specific region or demographic group, public health campaigns and initiatives can be developed to raise awareness, encourage healthy lifestyle choices, and provide resources for diabetes prevention.
5. **Research and Insights:** Itcan aid researchers in studying the relationships between different health indicators and diabetes and can gain insights into the underlying mechanisms and risk factors associated with diabetes.

9. CONCLUSION

The ML model for predicting diabetes using the Pima Indians Diabetes Dataset was implemented and evaluated. The model, based on the Random Forest algorithm, was trained on historical data to learn patterns and relationships between various features and the presence or absence of diabetes. The model offers valuable insights and predictions that can assist in managing and preventing diabetes at both the individual and population levels.The performance of the model was assessed using different evaluation metrics such as accuracy, precision, recall, and F1-score. The results indicate that the model achieved good performance, demonstrating its ability to effectively predict the presence or absence of diabetes in new data
The correlation analysis show that the most important indicators are:

1. Self-reported health status
2. High blood pressure
3. High BMI
4. High cholesterol
5. Age
6. DIfficulties in walking
7. Having heart disease or attack

10. FUTURE SCOPE

The future scope of the model lies in further enhancing its predictive accuracy and usability. This can be achieved through feature engineering, exploring alternative machine learning algorithms, evaluating additional performance metrics, and augmenting the dataset. External validation ,that is the model can be tested on external datasets to assess its generalizability. Using diverse datasets from different populations and demographics will help evaluate the model's performance across different contexts and ensure its reliability. Furthermore, integrating the ML model into a real-time system or web application would make it readily available to healthcare professionals and individuals, enabling timely and accurate diabetes prediction.

11 BIBLIOGRAPHY

[1] ***Diabetes Prediction using Machine Learning Algorithms*****. (2020, February 27). Diabetes Prediction Using Machine Learning Algorithms - ScienceDirect. https://doi.org/10.1016/j.procs.2020.01.047**

[2] ***A review on current advances in machine learning based diabetes prediction*****. (2021, February 26). A Review on Current Advances in Machine Learning Based Diabetes Prediction - ScienceDirect. https://doi.org/10.1016/j.pcd.2021.02.00**

APPENDIX A.

Source Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, roc_curve
# train data
train = pd.read_csv('/content/diabetes_binary_5050split_health_indicators_BRFSS2015.csv')
# test data
test = pd.read_csv('/content/diabetes_binary_health_indicators_BRFSS2015.csv')
print(f"train shape: {train.shape}")
print(f"test shape: {test.shape}")
# check the columns
print(train.columns == test.columns)
print(train.dtypes)
train.isnull().sum()
train.describe()
cat_socialecom = ['Age', 'Sex', 'Education', 'Income', 'AnyHealthcare', 'NoDocbcCost']
```

```
  cat_disease = ['HighBP', 'HighChol', 'CholCheck', 'Stroke', 'HeartDiseaseorAttack',
'DiffWalk']
  cat_health = ['GenHlth', 'MentHlth', 'PhysHlth']
  cat_habit = ['PhysActivity', 'Smoker', 'Fruits', 'Veggies', 'HvyAlcoholConsump']
  fig, axes = plt.subplots(1, 3, figsize=(15, 5))
  axes = axes.flatten()
  sns.countplot(x='Age', data=train, ax=axes[0])
  axes[0].set_title('Age')
  axes[0].set_xlabel('Age Group')
  for i, col in enumerate(['Sex', 'Education']):
     train[col].value_counts().plot(kind='pie', autopct='%1.1f%%', ax=axes[i+1], title=col)
  plt.suptitle('Subject Profile', fontsize=14)
  plt.show()
  fig, axes = plt.subplots(1, 3, figsize=(15, 5))
  axes = axes.flatten()
  for i, col in enumerate(['Income', 'AnyHealthcare', 'NoDocbcCost']):
     train[col].value_counts().plot(kind='pie', autopct='%1.1f%%', ax=axes[i], title=col)
  plt.suptitle('Subject Income and Health Insurance Status', fontsize=14)
  plt.show()
  fig, axes = plt.subplots(1, 3, figsize=(15, 4))
  axes = axes.flatten()
  for i, col in enumerate(cat_health):
     g = sns.countplot(x=col, data=train, ax=axes[i])
     if len(train[col].unique()) > 5:
        g.set_xticks(np.arange(0, len(train[col].unique()), 5))
  plt.suptitle('Subject health self-evaluation', fontsize=14)
  plt.tight_layout()
  plt.show()
  fig, ax = plt.subplots(2, 3, figsize=(12, 8))
  for i, col in enumerate(cat_disease):
     train[col].value_counts().plot.pie(ax=ax[i//3, i%3], autopct='%.2f%%', title=col)
  plt.suptitle('Proportion of health issue indicators', fontsize=14)
  plt.show()
  fig, ax = plt.subplots(2, 3, figsize=(12, 8))
  for i, col in enumerate(cat_habit):
     train[col].value_counts().plot.pie(ax=ax[i//3, i%3], autopct='%.2f%%', title=col)
  plt.suptitle('Proportion of habit indicators', fontsize=14)
  plt.show()
  plt.axvline(18.5, color='gray', linestyle='--')
  plt.axvline(24.9, color='gray', linestyle='--')
  plt.axvline(29.9, color='gray', linestyle='--')
  plt.text(12, 5500, 'Under', rotation=90, size=8)
```

```
  plt.text(20, 5500, 'Normal', rotation=90, size=8)
  plt.text(27, 5500, 'Over', rotation=90, size=8)
  plt.text(50, 5500, 'Obesity', rotation=90, size=8)
  sns.histplot(train['BMI'], kde=False)
  plt.title('Distribution of BMI')
  plt.show()
  cols = ['Diabetes_binary'] + cat_socialecom + cat_disease +  ['BMI'] + cat_health +
cat_habit
  plt.figure(figsize=(18, 9))
  sns.heatmap(train[cols].corr(), annot=True, cmap='RdBu', vmin=-1, vmax=1)
  plt.title('Correlation between variables')
  plt.show()
  # split the data into X and y
  X = train.drop('Diabetes_binary', axis=1)
  y = train['Diabetes_binary']
  # split the data into train and validation set
  X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)
  X_test = test.drop('Diabetes_binary', axis=1)
  y_test = test['Diabetes_binary']
  scaler = StandardScaler()
  X_train = scaler.fit_transform(X_train)
  X_val = scaler.transform(X_val)
  X_test = scaler.transform(X_test)
  def evaluate_model(model, X_train, y_train, X_val, y_val):
    model.fit(X_train, y_train)
    y_train_pred = model.predict(X_train)
    y_val_pred = model.predict(X_val)
    df = pd.DataFrame({'train_accuracy': [accuracy_score(y_train, y_train_pred)],
                       'train_precision': [precision_score(y_train, y_train_pred)],
                       'train_recall': [recall_score(y_train, y_train_pred)],
                       'train_f1': [f1_score(y_train, y_train_pred)],
                       'train_roc_auc': [roc_auc_score(y_train, y_train_pred)],
                       'val_accuracy': [accuracy_score(y_val, y_val_pred)],
                       'val_precision': [precision_score(y_val, y_val_pred)],
                       'val_recall': [recall_score(y_val, y_val_pred)],
                       'val_f1': [f1_score(y_val, y_val_pred)],
                       'val_roc_auc': [roc_auc_score(y_val, y_val_pred)]})
    return df
  models = {  'Logistic Regression': LogisticRegression(),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'KNN': KNeighborsClassifier(),
```

```
    'Naive Bayes': GaussianNB(),
    'Gradient Boosting': GradientBoostingClassifier()
  }
  results_l = []
  for name, model in models.items():
    model_results = evaluate_model(model, X_train, y_train, X_val, y_val)
    model_results['model'] = name
    results_l.append(model_results)
  results = pd.concat(results_l, axis=0).reset_index(drop=True)
  results.sort_values(by='val_accuracy', ascending=False)
  param_grid = {
    'max_depth': [4, 5, 6],
    'max_features': ['sqrt'],
    'min_samples_split': [10, 15, 20],
    'min_samples_leaf': [1, 2, 3],
    'n_estimators': [50, 100, 150],}
  gb = GradientBoostingClassifier()
  grid_search = GridSearchCV(gb, param_grid, cv=3, scoring='accuracy', n_jobs=-1,
verbose=1)
  # fit the model
  grid_search.fit(X_train, y_train)
  # print the best parameters
  print(grid_search.best_params_)
  best_model = GradientBoostingClassifier(**grid_search.best_params_)
  evaluate_model(best_model, X_train, y_train, X_val, y_val)
  plt.figure(figsize=(6, 6))
  plt.pie(y_test.value_counts(), labels=['No Diabetes', 'Diabetes'], autopct='%1.1f%%',
startangle=90)
  plt.title('Portion of each class in the target variable')
  plt.show()
  # predict the test set
  y_test_pred = best_model.predict(X_test)
  # print the classification report
  print(classification_report(y_test, y_test_pred))
  print(f'ROC AUC score: {roc_auc_score(y_test, y_test_pred)}')
  # plot the confusion matrix
  cm = confusion_matrix(y_test, y_test_pred)
  sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
  plt.xlabel('Predicted')
  plt.ylabel('Actual')
  plt.xticks([0.5, 1.5], ['No Diabetes', 'Diabetes'])
  plt.yticks([0.5, 1.5], ['No Diabetes', 'Diabetes'])
```

```
plt.title('Confusion matrix')
plt.show(
fpr, tpr, thresholds = roc_curve(y_test, y_test_pred)
plt.plot(fpr, tpr, label='ROC curve')
plt.plot([0, 1], [0, 1], 'k--', label='Random guess')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve')
plt.show()
```