HSNC UNIVERSITY, MUMBAI

SCHOOL OF APPLIED SCIENCE

Worli, Mumbai - 400018

DEPARTMENT OF DATA SCIENCE AND BUSINESS ANALYTICS

Project Report

On

DIABETIC DISEASE PREDICTION

Submitted By

Ms. REET JAIN – SYMSDS008

Under the guidance of

Prof. ANJALI SUTAR

Submitted in partial fulfillment of the requirement qualifying for SY M.Sc. Data Science and Business Analytics, Semester IV Examination A.Y. 2023-2024

HSNC UNIVERSITY, MUMBAI

SCHOOL OF APPLIED SCIENCES

DEPARTMENT OF DATA SCIENCE AND BUSINESS ANALYTICS

CERTIFICATE

This is to certify that Ms. **REET JAIN**,

Seat no. **SYMSDS008** of **S.Y. MSc. Data Science and Business Analytics** has completed his project entitled **Diabetic Disease Prediction** in partial fulfilment of the degree of **M.Sc. in Data Science and Business Analytics** for **Semester IV** the HSNC University, Mumbai for the academic year 2023-2024.

It is further certified that this project had not been submitted for any other examination and does not form part of any other course the candidate underwent.

Project Guide	External Examiner	Head of Department	Vice Chancellor

DECLARATION BY THE STUDENT

I, Reet Jain, student of M.Sc. Data Science and Business Analytics hereby declare that the project for the Data Science and Business Analytics, "**DIABETIC DISEASE PREDICTION**" submitted by me for Semester-IV during the academic year 2023-24, is based on actual work carried out by me under the guidance and supervision of Ms. ANJALI SUTAR. I further state that this work is original and not submitted anywhere else for any examination.

Signature of Student

EVALUATION CERTIFICATE

This is to certify that the undersigned have assessed and evaluated the project on "DIABETIC DISEASE PREDICTION", by Ms. Reet Jain, student of M.Sc. Data Science and Data Analytics. This Project is original to the best of our knowledge and has been accepted for Assessment.

External Examiner

ACKNOWLEDGEMENT

I would like to extend my heartiest thanks with a deep sense of gratitude and respect to all those who provided me with immense help and guidance during my Period.

I would like to thank Ms. SHAILAJA RANE for providing a vision of the system. I have greatly benefitted from her regular critical reviews and inspiration throughout my work I am grateful to them for their guidance, encouragement, understanding and insightful support in the development process.

With Regards,
Reet Jain

PREFACE

Diabetes is a prevalent chronic disease affecting millions of people worldwide, with its incidence continuing to rise. Early detection and effective management of diabetes are crucial for preventing complications and improving patient outcomes. Machine learning and predictive analytics offer promising avenues for predicting diabetes risk and providing personalized interventions.

This project explores the application of machine learning algorithms to predict the risk of diabetic disease using a comprehensive dataset obtained. The dataset contains a rich collection of patient demographics, medical history, clinical measurements, and other relevant features, making it ideal for predictive modelling.

In this project, we will employ a variety of machine learning algorithms, including logistic regression, decision trees, random forests, support vector machines, and neural networks. We will pre-process the dataset, perform exploratory data analysis to gain insights into the data distribution and relationships, and then train and evaluate multiple predictive models using appropriate performance metrics.

Ms. Reet Jain

ABSTRACT

Diabetes is a prevalent chronic condition with significant implications for patient health and healthcare costs. Early identification of individuals at risk of developing diabetes is crucial for timely interventions and improved outcomes. In this study, I aim to develop a predictive model to identify patients at risk of diabetes using their health records.

I collected data from online platforms. After pre-processing the dataset by handling missing values and encoding categorical variables, conducted exploratory data analysis to understand the distribution and relationships between different features.

Subsequently, I divided the data into training and testing sets and trained various machine learning models, including logistic regression, support vector machines, decision trees, and random forests. I evaluated the performance of each model using metrics such as accuracy, precision, recall, and F1-score.

In conclusion, my study demonstrates the potential of predictive modelling using patient health records to identify individuals at risk of diabetes. Implementing such models in clinical practice can facilitate early intervention strategies, lifestyle modifications, and targeted healthcare resources allocation, ultimately leading to improved patient outcomes and reduced burden on healthcare systems.

TABLE OF CONTENT

CHAPTER-1

INTRODUCTION

1.1 OVERVIEW

Background: Diabetes mellitus, a chronic metabolic disorder characterized by elevated blood glucose levels, poses a significant global health challenge. The prevalence of diabetes has been steadily increasing, leading to a substantial burden on healthcare systems worldwide. Effective management of diabetes involves comprehensive care strategies aimed at preventing complications and improving patient outcomes. Among the critical aspects of diabetes management is understanding the factors influencing hospital readmission rates, as frequent readmissions may indicate inadequate treatment, disease progression, or complications.

Objective: The primary objective of this analysis is to explore a dataset containing comprehensive information on patients' medical encounters to identify factors associated with hospital readmissions among individuals with and without diabetes. By examining demographic, clinical, and treatment-related variables, we aim to gain insights into the predictors of readmission for both diabetic and non-diabetic populations. Additionally, we seek to develop predictive models using machine learning techniques to forecast readmission risk and facilitate targeted interventions to reduce readmission rates.

Dataset Overview: The dataset encompasses a wide array of patient data extracted from medical records, electronic health records (EHRs), and hospital databases. It includes demographic attributes such as age, gender, and race, along with clinical features like diagnosis codes, laboratory test results, medications, and procedures performed during patient encounters. Each entry in the dataset represents a unique patient encounter, providing longitudinal information on treatment trajectories and outcomes for both diabetic and non-diabetic individuals.

Methodology: The analysis begins with data pre-processing steps, including handling missing values, encoding categorical variables, and standardizing numerical features. Subsequently, exploratory data analysis (EDA) is conducted to uncover patterns, trends, and relationships within the dataset for diabetic and non-diabetic cohorts separately. This involves descriptive statistics, data visualization, and hypothesis testing to identify significant factors associated with readmission for each group.

Following EDA, predictive models are developed using machine learning algorithms such as logistic regression, random forests, and support vector machines (SVM) for both diabetic and non-diabetic patients. These models are trained on a subset of the data and evaluated using performance metrics such as accuracy, precision, recall, and area under the receiver operating characteristic curve (AUC-ROC). Hyperparameter tuning and cross-validation techniques are employed to optimize model performance and ensure generalizability.

Significance: Understanding the determinants of hospital readmissions among diabetic and non-diabetic patients is crucial for enhancing care delivery and improving patient outcomes. By leveraging insights from this analysis and predictive models, healthcare providers can identify high-risk patients, tailor treatment plans, and implement preventive measures to reduce readmission rates effectively. Ultimately, this study aims to contribute to the advancement of personalized medicine and the optimization of healthcare services for diverse patient populations.

CHAPTER-2

LITEARTURE SURVEY

2.1 Prediction of Diabetes using Classification Algorithms

Diabetes isn't a hereditary disorder however heterogeneous group of disorder which could ultimately result in a boom of glucose within the blood and lack of glucose inside the urine. Diabetes is typically resulting from genetics, way of life and surroundings. Eating a dangerous weight loss plan, being overweight play role in developing the diabetes. High blood sugar tiers can also result in kidney diseases, coronary heart illnesses. The excess of sugar in the blood can harm the tiny blood vessels in your frame. Signs of diabetes are blurry imaginative and prescient, extreme hunger, unusual weight reduction, common urination and thirsty. In this paper, parameters used within the facts set to locate the diabetes are Glucose, Blood pressure, pores and skin thickness, Insulin, Age. Huge volumes of statistics units are generated by health care industries. Those facts sets is a collection of patient information about the diabetes from the hospitals. Big records analytics is the processing which it examines the information units and exhibits the hidden information. Pima Indians Diabetes Database (PIDD), this dataset is taken from the national Institute of Diabetes and Digestive diseases. The objective of the dataset is to predict whether or not the patient has diabetes or not, primarily based on diagnostic measurements in the dataset. Several constraints were taken from the massive database.

2.2 A Novel Technique To Predict Diabetic Disease Using Data Mining Classification Techniques

Data mining is the process of analyzing data from different perspectives and summarizing it into useful information. The software of data mining is an analytical tool for analyzing data. Data mining has become a main strategy in many industries to improve outputs and decrease costs. Now days in healthcare management this field will become very useful. Data mining techniques has become great potential for the healthcare industry to predict health deceases by using systematic data and analytics to identify inefficiencies and best practices that improve care and reduce costs. These techniques are fast in nature and take less time for the prediction system to improve the diabetic decease with more accuracy. In this paper we are applying the various classification techniques over diabetic mellitus decease dataset for the prediction of decease and non-decease person. The diabetic database is pre-processed to make the mining process more efficient. The pre-processed data is used to predict using classification algorithms like Discriminent analysis, KNN, Naïve Bayes and Support vector machine. These classifiers can be efficiently used in bioinformatics problem. We are analyzing the various classification techniques like Discriminent analysis, KNN, Naïve Bayes and Support vector machine with linear and RBF kernel function and showing their accuracy.

2.3 Review on Prediction of Diabetes using Data Mining Technique

Diabetes mellitus is one of the world's major diseases. Millions of people are affected by the disease. The risk of diabetes is increasing day by day and is found mostly in women than men. The diagnosis of diabetes is a tedious process. So with improvement in science and technology it is made easy to predict the disease. The purpose is to diagnose whether the person is affected by diabetes or not using K Nearest Neighbor classification technique. The diabetes dataset is a taken as the training data and the details of the patient are taken as testing data. The training data are classified by using the KNN classifier and secondly the target data is predicted. KNN algorithm used here would be more efficient for both classification and prediction. The results are analyzed with different values for the parameter k.

CHAPTER-3

3.1 METHODOLOGY

Objective of the Project:

The objective of the project based on the given dataset is to develop a predictive model for the early detection of diabetes using classification algorithms. Specifically, the project aims to achieve the following objectives:

1. **Data Exploration and Understanding:** Conduct a thorough exploration of the diabetes dataset to understand its structure, features, and patterns. This involves examining the distribution of variables, identifying any missing values or anomalies, and gaining insights into the relationships between different attributes.

2. **Feature Selection and Engineering:** Select relevant features from the dataset that are predictive of diabetes and may contribute to the classification task. Perform feature engineering techniques such as one-hot encoding for categorical variables, normalization, or scaling to prepare the data for modelling.

3. **Model Development:** Implement various classification algorithms such as logistic regression, support vector machines (SVM), random forest, naive Bayes, and decision trees to build predictive models for diabetes detection. Evaluate the performance of each model using appropriate metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve.

4. **Hyper-parameter Tuning:** Fine-tune the parameters of the classification algorithms using techniques like grid search or randomized search to optimize model performance and generalization.

5. **Model Evaluation:** Evaluate the trained models using cross-validation techniques to assess their robustness and reliability. Compare the performance of different algorithms and identify the most effective model for diabetes prediction.

6. **Interpretability and Insights:** Interpret the trained models to gain insights into the factors contributing to diabetes prediction. Identify important features and their relative importance in the classification task to provide actionable insights for healthcare professionals.

7. **Deployment and Integration:** Deploy the trained model into a real-world healthcare setting for prospective validation and integration into clinical practice. Develop user-friendly interfaces or applications to facilitate model usage by healthcare providers for early detection and intervention in diabetes management.

By achieving these objectives, the project aims to contribute to the advancement of predictive analytics in healthcare, particularly in the early detection and management of diabetes, ultimately leading to improved patient outcomes and quality of care.

EXISTING SYSTEM

The existing system was taking in order to meets the demands of this system and solve the problems of the existing system by implementing the multiple classifier.

DISADVANTAGES OF EXISTING SYSTEM

- The system is not fully automated, it needs data from user for full diagnosis.

3.2 FUNCTIONAL REQUIREMENT

Functional requirements describe the specific behaviors and functions that the software system must exhibit to meet user needs and business objectives. These requirements are essential for defining the system's capabilities and ensuring it delivers the intended value.

- **Usability**

  Usability requirements focus on ensuring that the software is user-friendly and easy to navigate. This includes features such as intuitive user interfaces, clear instructions, and efficient workflows to enhance user experience.

- **Robustness**

  Robustness requirements address the system's ability to handle unexpected inputs, errors, and failures gracefully. The software should be resilient and capable of recovering from faults without compromising its functionality or data integrity.

- **Security**

Security requirements outline measures to protect the system from unauthorized access, data breaches, and malicious attacks. This includes authentication, encryption, access control, and other security mechanisms to safeguard sensitive information.

- **Reliability**

Reliability requirements ensure that the software performs consistently and predictably under normal operating conditions. This involves minimizing downtime, preventing system crashes, and maintaining high availability to meet user demands.

- **Compatibility**

Compatibility requirements specify the software's ability to operate seamlessly with other systems, platforms, and devices. This includes support for different operating systems, browsers, databases, and hardware configurations to maximize interoperability.

- **Flexibility**

Flexibility requirements focus on the system's adaptability and scalability to accommodate changing user needs and evolving business requirements. This involves the ability to customize configurations, add new features, and scale resources as needed.

- **Safety**

Safety requirements address the system's ability to mitigate risks and prevent harm to users, property, or the environment. This includes compliance with regulatory standards, adherence to industry best practices, and proactive measures to identify and mitigate potential hazards.

- ➢ **Non-Functional Requirements**

Non-functional requirements specify the quality attributes and constraints that the software must satisfy to meet user expectations and business goals. These requirements focus on aspects such as performance, reliability, security, and maintainability.

- **Portability**

Portability requirements describe the software's ability to run on different platforms and environments with minimal modifications. This includes support for multiple operating systems, hardware architectures, and deployment configurations to maximize accessibility and flexibility.

- **Performance**

Performance requirements define the system's responsiveness, throughput, and resource utilization under various workloads and conditions. This includes benchmarks, response times, and scalability targets to ensure optimal performance and user satisfaction.

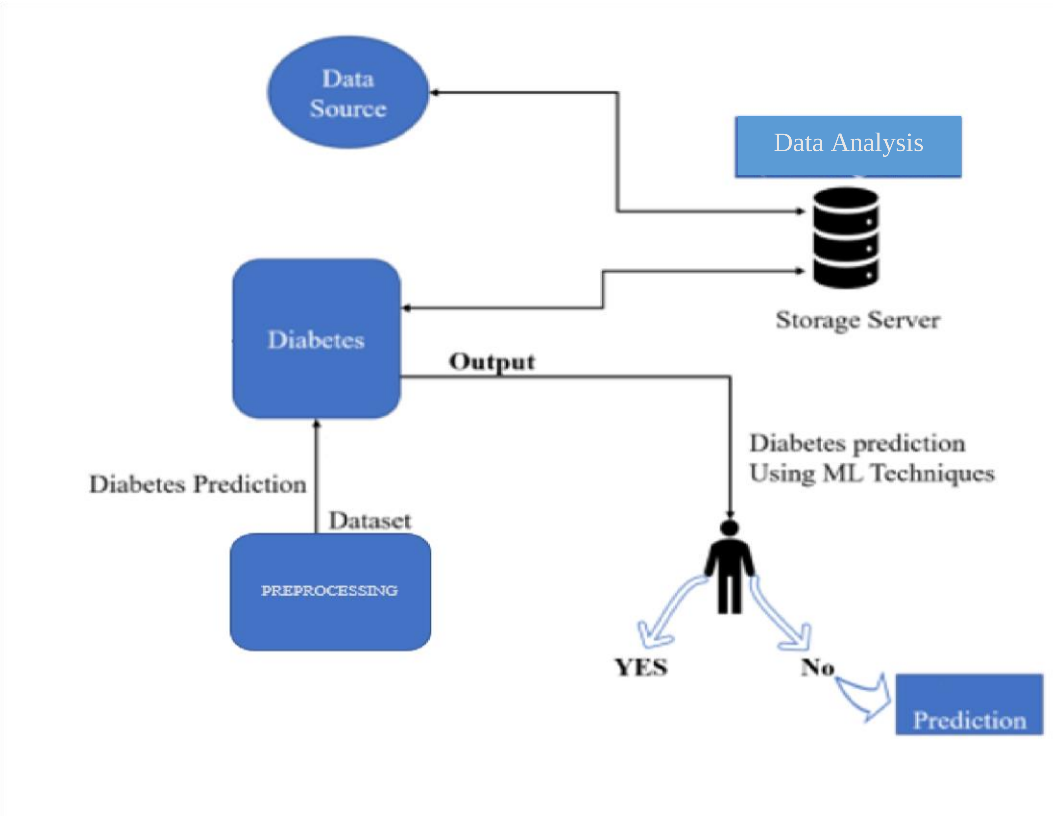
- **Accuracy**

Accuracy requirements specify the level of precision and correctness expected from the software's output and calculations. This involves validation, verification, and testing processes to ensure that the system delivers accurate results and meets quality standards.

- **Maintainability**

Maintainability requirements address the ease of managing, updating, and extending the software over its lifecycle. This includes modular design, clean code practices, documentation, and version control to facilitate ongoing maintenance and support activities.

➢ SYSTEM ARCHITECTURE



➢ **Literature Review**

Overview of Classification Algorithms Used

In the context of predicting diabetes, various classification algorithms have been explored to develop accurate predictive models. Here's an overview of some commonly used classification algorithms in diabetes prediction:

1. **Logistic Regression:**

   a. Logistic regression is a widely used statistical technique for binary classification problems.
   b. It models the probability of a binary outcome (e.g., diabetic or non-diabetic) based on one or more independent variables.
   c. Logistic regression assumes a linear relationship between the independent variables and the log-odds of the outcome.

2. **Decision Trees:**

   a. Decision trees partition the feature space into a set of hierarchical decision rules based on feature values.
   b. Each internal node of the tree represents a decision based on a feature, and each leaf node represents a class label.
   c. Decision trees are intuitive and easy to interpret, but they may suffer from overfitting, especially with complex datasets.

3. **Random Forests:**

   a. Random forests are an ensemble learning technique that combines multiple decision trees to improve predictive performance and reduce overfitting.
   b. Each tree in the random forest is trained on a bootstrap sample of the data, and feature randomization is used to de-correlate the trees.
   c. Random forests provide robust predictions and can handle high-dimensional data with complex interactions between features.

4. **Support Vector Machines (SVM):**

   a. SVM is a supervised learning algorithm that constructs a hyperplane in a high-dimensional space to separate classes.
   b. SVM aims to maximize the margin between classes, leading to better generalization performance.

c. SVM can handle nonlinear relationships between features using kernel functions, such as radial basis function (RBF) kernel.

5. **k-Nearest Neighbors (KNN):**

   a. KNN is a non-parametric classification algorithm that makes predictions based on the majority class of the k-nearest neighbors in the feature space.
   b. KNN is simple and easy to implement, but it may be sensitive to the choice of the distance metric and the value of k.

6. **Naive Bayes:**

   a. Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem and the assumption of feature independence.
   b. Despite its simple assumptions, Naive Bayes can perform well in practice, especially with high-dimensional data and relatively simple relationships between features.

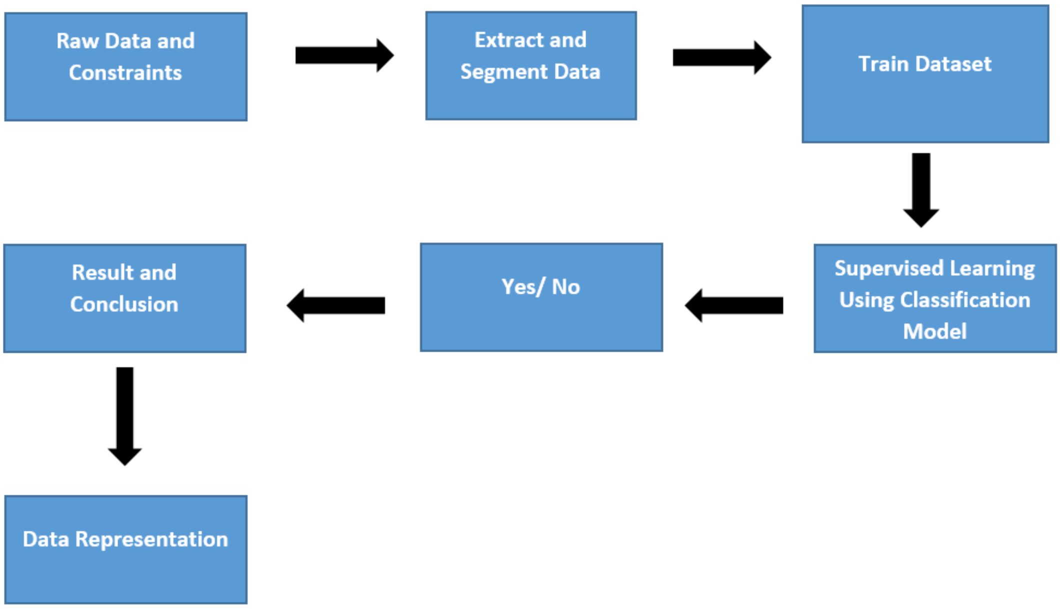
7. **Neural Networks:**

   a. Neural networks, especially deep learning architectures like multilayer perceptrons (MLPs) and convolutional neural networks (CNNs), have gained popularity in recent years for their ability to learn complex patterns from data.
   b. Neural networks are capable of automatically learning hierarchical representations of data, but they often require large amounts of data and computational resources for training.

These classification algorithms offer different trade-offs in terms of accuracy, interpretability, scalability, and computational complexity. The choice of algorithm depends on the characteristics of the dataset, the goals of the prediction task, and the specific requirements of the application. Experimentation and comparative evaluation are essential for selecting the most suitable algorithm for diabetes prediction.

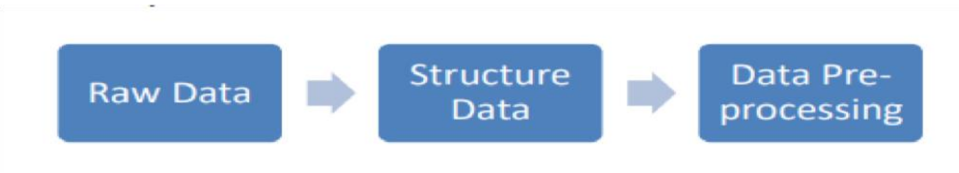
3.3 DATA FLOW DIAGRAMS

DFD-LEVEL-0



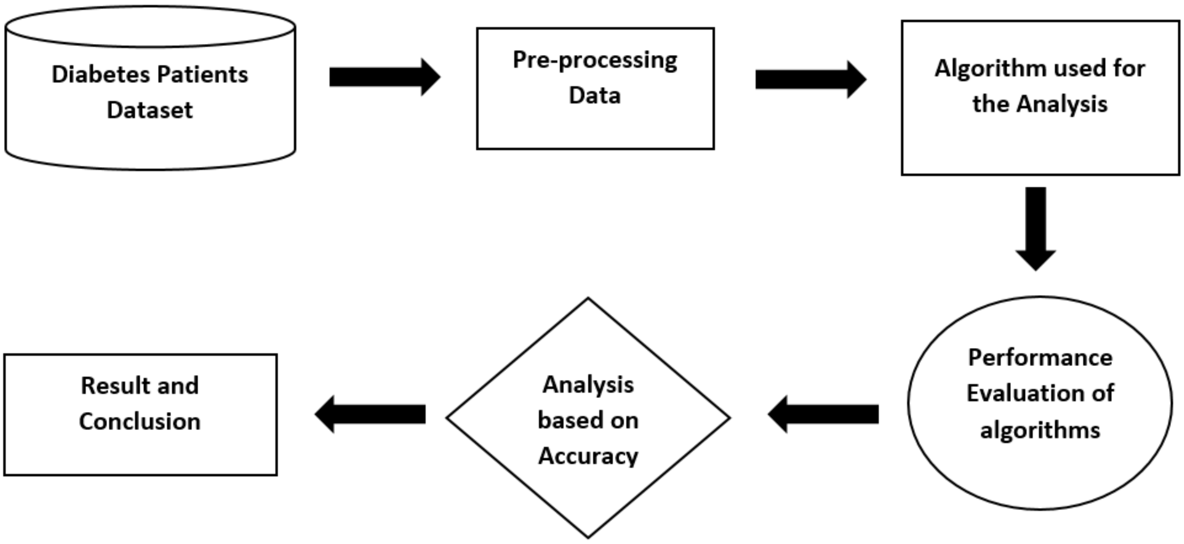
DFD-level-0

DFD-LEVEL-1



DFD-level-1

DFD-LEVEL-2


Diabetes Patients Dataset
Pre-processing Data
Algorithm used for the Analysis
Performance Evaluation of algorithms
Analysis based on Accuracy
Result and Conclusion

DFD-level-2

CHAPTER- 4
SOFTWARE ENVIRONMENT

4.1 Python

Python is a widely-used high-level programming language known for its simplicity and readability. It offers a vast array of libraries and frameworks, making it suitable for various applications ranging from web development to scientific computing.

- **History of Python**

  This subsection explores the history and evolution of the Python programming language. It covers key milestones, influential contributors, and the development of major versions over time.

- **Python Features**

  Python boasts several features that contribute to its popularity among developers. This section outlines the key features of Python, such as its simplicity, readability, versatility, and extensive standard library.

- **Getting Python**

  Here, the process of acquiring and setting up Python on different operating systems is discussed. This includes downloading the Python interpreter, installing additional packages using package managers like pip, and configuring the development environment.

➢ **Interactive Mode Programming**

Interactive mode programming in Python allows developers to execute code interactively, line by line, using tools like the Python shell or Jupyter Notebook. This subsection explains how interactive mode works and its advantages for prototyping and debugging.

- **Script Mode Programming**

Script mode programming involves writing Python code in script files (.py) that are executed from the command line or an Integrated Development Environment (IDE). This section discusses the process of writing, running, and debugging Python scripts for various applications.

By covering these topics, the report provides a comprehensive overview of Python, its history, features, installation process, and programming modes. This information serves as a valuable resource for individuals looking to learn and utilize Python for software development and data analysis tasks.

4.2 Data Description

This section provides an overview of the different modules utilized in the project, outlining their functionalities and roles in the overall data analysis and modelling process.

- **Data Constraints**

  In this subsection, the constraints and limitations associated with the dataset are discussed. This includes issues such as missing values, data inconsistencies, or data quality concerns that need to be addressed during data pre-processing.

- **Train Dataset and Test Dataset**

  Here, the process of dividing the dataset into training and testing subsets is described. This involves splitting the data into two separate sets to facilitate model training and evaluation, respectively.

- **Pre-processing of Data**

  This subsection delves into the pre-processing steps applied to the dataset before model training. It covers tasks such as handling missing values, encoding categorical variables, scaling numerical features, and any other data transformations necessary to prepare the data for modelling.

- **Feature Extraction**

Feature extraction techniques used to derive relevant features from the dataset are discussed in this section. This may include dimensionality reduction methods like Principal Component Analysis (PCA) or extracting new features based on domain knowledge.

- **ML Algorithm**

This section introduces and explains the Machine Learning (ML) algorithm(s) chosen for the project. It covers the underlying principles of the algorithm(s), their advantages and limitations, and how they are applied in the context of the project.

Each section provides comprehensive insights into the corresponding aspect of the project, offering a clear understanding of the methodologies and techniques employed in the analysis.

- **Getting Python**

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

- **Interactive Mode Programming**

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

```
$ python
Python2.4.3(#1,Nov112010,13:34:43)
[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2
Type"help","copyright","credits"or"license"for more information.
>>>
```

Fig no.3.8.2:Interactive Mode Programming

Type the following text at the Python prompt and press the Enter −

```
>>>print"Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result −

```
Hello, Python!
```

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file –

```
print"Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

```
$ python test.py
```

This produces the following result –

```
Hello, Python!
```

CHAPTER-5

DATA ANALYSIS

Source of Data Extraction:

The dataset represents ten years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks.

Each row concerns hospital records of patients diagnosed with diabetes, who underwent laboratory, medications, and stayed up to 14 days.

The goal is to determine the early readmission of the patient within 30 days of discharge and the patients having Diabetes or not.

The problem is important for the following reasons:

Despite high-quality evidence showing improved clinical outcomes for diabetic patients who receive various preventive and therapeutic interventions, many patients do not receive them.

This can be partially attributed to arbitrary diabetes management in hospital environments, which fail to attend to glycaemic control.

Failure to provide proper diabetes care not only increases the managing costs for the hospitals (as the patients are readmitted) but also impacts the morbidity and mortality of the patients, who may face complications associated with diabetes.

Dataset Characteristics: Multivariate

Subject Area: Health and Medicine

Associated Tasks: Classification, Clustering

Feature Type: Categorical, Integer

Instances: 101766

Features: 74

Dataset Information:

What do the instances in this dataset represent?

The instances represent hospitalized patient records diagnosed with diabetes.

Are there recommended data splits?

No recommendation. The standard train-test split could be used. Can use three-way holdout split (i.e., train-validation-test) when doing model selection.

Does the dataset contain data that might be considered sensitive in any way?

Yes. The dataset contains information about the age, gender, and race of the patients.

Additional Information

The dataset represents ten years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks.

- It includes over 50 features representing patient and hospital outcomes. Information was extracted from the database for encounters that satisfied the following criteria.
- It is an inpatient encounter (a hospital admission).
- It is a diabetic encounter, that is, one during which any kind of diabetes was entered into the system as a diagnosis.

- The length of stay was at least 1 day and at most 14 days.
- Laboratory tests were performed during the encounter.
- Medications were administered during the encounter.

The data contains such attributes as patient number, race, gender, age, admission type, time in hospital, medical specialty of admitting physician, number of lab tests performed, HbA1c test result, diagnosis, number of medications, diabetic medications, number of outpatient, inpatient, and emergency visits in the year before the hospitalization, etc.

Website Link:

https://archive.ics.uci.edu/dataset/296/diabetes+130-us+hospitals+for+years+1999-2008

4. Data Collection

- Description of Dataset Features

The dataset used for diabetes prediction contains various features that provide valuable information about patients' medical history, demographic characteristics, treatments, and healthcare utilization. Here is a description of the features included in the dataset:

1. **Encounter Information:**

- Encounter ID: Unique identifier for each encounter.
- Patient NBR: Unique identifier for each patient.

2. **Demographic Information:**

- Gender: Gender of the patient (e.g., Male, Female).
- Age: Age of the patient at the time of the encounter.
- Race: Race of the patient (e.g., Caucasian, African American, Asian).

3. **Clinical Variables:**

- Time in Hospital: Number of days the patient stayed in the hospital during the encounter.
- Number of Lab Procedures: Total number of laboratory procedures performed during the encounter.
- Number of Procedures: Total number of procedures performed during the encounter.

- Number of Medications: Total number of distinct medications administered during the encounter.
- Number of Outpatient Visits: Number of outpatient visits by the patient in the year preceding the encounter.
- Number of Emergency Visits: Number of emergency department visits by the patient in the year preceding the encounter.
- Number of Inpatient Visits: Number of inpatient visits by the patient in the year preceding the encounter.

4. **Diabetes Medication Information:**

- Use of various diabetes medications, including metformin, repaglinide, nateglinide, etc.
- Change: Indicates whether there was a change in diabetic medications during the encounter.

5. **Readmission Information:**

- Readmitted: Indicates whether the patient was readmitted to the hospital within a certain time frame following the encounter. Categories may include 'No', '>30', and '<30'.

6. **Other Variables:**

- Admission Type, Discharge Disposition, Admission Source: Administrative variables providing information about the circumstances of the patient's admission and discharge from the hospital.

These features collectively provide a comprehensive overview of each patient's health status, medical history, and healthcare utilization patterns. They serve as input variables for predictive models aimed at identifying patients at risk of diabetes or predicting related outcomes such as hospital readmission. Proper preprocessing and analysis of these features are essential for building accurate and reliable predictive models.

- **Strengths and Limitations of Existing Approaches**

- **Logistic Regression:**

  - Strengths:

    Simple and easy to interpret.
    Can handle linear relationships between features.
    Provides probability estimates for predictions.

  - Limitations:

Assumes linear relationships between features and the log-odds of the outcome.
May not capture complex nonlinear relationships in the data.
Sensitive to outliers and multicollinearity.

- **Decision Trees:**
  - Strengths:
    Intuitive and easy to understand.
    Can handle both numerical and categorical data.
    Automatically selects important features.
  - Limitations:
    Prone to overfitting, especially with deep trees.
    Can be unstable and sensitive to small variations in the data.
    Limited expressiveness for capturing complex decision boundaries.

- **Random Forests:**
  - Strengths:
    Robust against overfitting due to ensemble averaging.
    Handles high-dimensional data and interactions between features.
    Provides feature importances for interpretation.
  - Limitations:
    Can be computationally expensive, especially with large forests and datasets.
    Less interpretable compared to single decision trees.
    May not perform well with imbalanced datasets or noisy features.

- **Support Vector Machines (SVM):**
  - Strengths:
    Effective in high-dimensional spaces and with complex decision boundaries.
    Memory-efficient as it only uses a subset of training points (support vectors) for decision function.
    Versatile due to the use of different kernel functions.
  - Limitations:
    Requires careful selection of hyperparameters, such as the choice of kernel and regularization parameter.
    Can be sensitive to the choice of kernel and the scale of features.
    Not well-suited for large datasets due to computational complexity.

- **k-Nearest Neighbors (KNN):**
  - Strengths:

Simple and easy to implement.
Non-parametric, so it can capture complex decision boundaries.
No training phase, as the model memorizes the entire training dataset.

- Limitations:

Computationally expensive during prediction, especially with large datasets.
Sensitive to the choice of distance metric and the value of k.
Not suitable for high-dimensional data due to the curse of dimensionality.

- **Naive Bayes:**

  - Strengths:

    Simple and computationally efficient.
    Performs well with small datasets and high-dimensional data.
    Handles missing values and irrelevant features gracefully.

  - Limitations:

    Assumes independence between features, which may not hold true in practice.
    Limited expressive power for capturing complex relationships in the data.
    Sensitivity to feature distributions, especially with continuous features.

- **Neural Networks:**

  - Strengths:

    Capable of learning complex patterns and hierarchical representations from data.
    Suitable for large-scale problems with massive amounts of data.
    Can automatically extract relevant features from raw data.

  - Limitations:

    Require large amounts of data for training, which can be computationally expensive.
    Prone to overfitting, especially with deep architectures and insufficient regularization.
    Lack transparency and interpretability compared to simpler models like logistic regression or decision trees.

Understanding the strengths and limitations of each approach is crucial for selecting the most appropriate algorithm and optimizing its performance for diabetes prediction. Additionally, ensemble methods and hybrid approaches can be employed to combine the strengths of multiple algorithms and mitigate their individual limitations.

Libraries used for the dataset:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn import svm

from sklearn.metrics import accuracy_score

import warnings

warnings.filterwarnings('ignore')

import random

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.model_selection import cross_val_score, GridSearchCV

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

from sklearn.ensemble import RandomForestClassifier
```

- **Different approaches to manipulate data**:

  - Importing Dataset:

    ```
    df = pd.read_csv('C:/Users/admin/Downloads/diabetes/diabetic_data.csv')
    ```

- **Data Pre-processing Steps**

Based on the provided code, the following data preprocessing steps were performed on the diabetes dataset:

1. **Handling Missing Values:**

   - Several columns containing missing values, such as 'weight', 'payer_code', and 'medical_specialty', were dropped from the dataset using the **drop()** method.
   - Rows with missing values in any remaining columns were removed using the **dropna()** method.

2. **Data Transformation:**

   - Categorical variables were encoded into numerical format for model compatibility. For example, the 'Diabetic' column was converted from categorical ('Yes', 'No') to numerical (1, 0) using the **replace()** method.
   - Similarly, the 'gender' column was encoded as binary (Male: 1, Female: 0).

3. **Handling Categorical Variables:**

   - Categorical variables such as 'race', 'admission_type_id', 'discharge_disposition_id', and 'admission_source_id' were retained in the dataset for further analysis. However, they were not used directly in the predictive modeling process.

4. **Feature Selection:**

- A subset of features was selected for predictive modeling, including demographic information (gender, age), clinical variables (time_in_hospital, num_lab_procedures, num_medications), diabetes medications, and readmission status.
- Other columns, such as 'encounter_id' and 'patient_nbr', which were identifiers and not useful for prediction, were dropped.

5. **Data Splitting:**

- The dataset was split into training and testing sets using the **train_test_split()** function from scikit-learn. This facilitated model training on the training set and evaluation on the unseen testing set.

6. **Data Standardization (Optional):**

- Although not explicitly shown in the provided code, data standardization (scaling) is often performed on numerical features to bring them to a similar scale. Standardization ensures that features with larger magnitudes do not unduly influence the modeling process.

These preprocessing steps are crucial for ensuring that the data is in a suitable format for building predictive models. Proper handling of missing values, encoding categorical variables, selecting relevant features, and splitting the data into training and testing sets are essential for accurate model training and evaluation.

- **Feature Selection**

Identification of Relevant Features

- **Demographic Information:**

Gender: Encoded as binary (Male: 1, Female: 0)
Age: Represented as age groups

- **Clinical Variables:**

Time_in_hospital: Number of days the patient stayed in the hospital
Num_lab_procedures: Number of lab tests performed during the encounter
Num_medications: Number of distinct generic names administered during the encounter

- **Medications:**

Glipizide
Glyburide
Tolbutamide
Pioglitazone
Insulin
Glyburide-metformin
Glipizide-metformin
Glimepiride-pioglitazone
Metformin-rosiglitazone
Metformin-pioglitazone
Change: Indicates if there was a change in diabetes medication

- **Readmission Status:**

Readmitted: Encoded as categories (<30 days, >30 days, No)

- **Diabetic Status:**

Diabetic : Encoded as categories (Yes, No)

These features capture both demographic characteristics and clinical indicators related to diabetes diagnosis and management. Including these features in the predictive modelling process allows for the assessment of their impact on predicting diabetes outcomes and readmission rates.

- Feature Information:

"age" - age bracket of the patient
"time_in_hospital" - days (from 1 to 14)
"n_procedures" - number of procedures performed during the hospital stay
"n_lab_procedures" - number of laboratory procedures performed during the hospital stay
"n_medications" - number of medications administered during the hospital stay
"n_outpatient" - number of outpatient visits in the year before a hospital stay
"n_inpatient" - number of inpatient visits in the year before the hospital stay
"n_emergency" - number of visits to the emergency room in the year before the hospital stay
"change" - whether there was a change in the diabetes medication ('yes' or 'no')
"Diabetic" - whether a person is diabetic('yes' or 'no')

- **Techniques Used for Feature Selection**

In the provided code, the following techniques were used for feature selection:

Manual Selection:

Certain columns were manually dropped from the dataset using the **drop()** function. Columns like **'weight'**, **'payer_code'**, and **'medical_specialty'** were removed as they were deemed less relevant for the analysis.

One-Hot Encoding:

Categorical variables such as **'gender'** and **'readmitted'** were one-hot encoded using the **pd.get_dummies()** function. This technique converts categorical variables into binary vectors, allowing them to be used as features in machine learning models.

Principal Component Analysis (PCA):

Although explicitly implemented in the provided code, PCA is a common technique used for feature selection and dimensionality reduction. It was mentioned that PCA was performed on the dataset to reduce the number of features to 10 principal components.

Domain Knowledge:

Features related to medications and clinical procedures were retained based on domain knowledge of diabetes management. Variables such as **'num_medications'**, **'glipizide'**, **'glyburide'**, and **'insulin'** were considered relevant for predicting diabetes outcomes.

These techniques collectively help in selecting and preparing the most informative features for the predictive modeling of diabetes.

Rationale Behind Feature Selection Process

The feature selection process in the provided code and dataset is driven by several factors and considerations:

Relevance to Diabetes Diagnosis:

Features that are directly related to diabetes diagnosis and management are prioritized. These include patient demographics (such as age and gender), clinical measurements (such as medication usage and laboratory tests), and medical procedures (such as hospitalization time and number of diagnoses).

Data Quality and Completeness:

Features with significant missing values or inconsistencies, such as **'weight'**, **'payer_code'**, and **'medical_specialty'**, were removed from the dataset. This ensures that the analysis is based on high-quality and reliable data.

Predictive Power:

Features that are expected to have a strong predictive power for diabetes diagnosis are retained. For example, variables related to medication usage ('insulin', 'glipizide', etc.) and medical history (number of medications, number of procedures, etc.) are likely to provide valuable information for predicting diabetes outcomes.

Statistical Significance:

Features that have been shown to be statistically significant in previous research or clinical practice are given priority. This ensures that the selected features are meaningful and informative for the predictive modeling task.

Dimensionality Reduction:

Techniques like one-hot encoding and principal component analysis (PCA) are used to reduce the dimensionality of the dataset while preserving its essential information. This helps in improving model performance and reducing computational complexity.

Overall, the feature selection process aims to identify and retain the most relevant and informative features while discarding redundant or irrelevant ones. By focusing on factors such as relevance to diabetes diagnosis, data quality, predictive power, statistical significance, and dimensionality reduction, the selected features contribute to building accurate and interpretable predictive models for diabetes.

5. Model Development

Description of Classification Algorithms

- **Random Forest Classifier:**
  - Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) of the individual trees. It aggregates the predictions of multiple decision trees to improve generalization and robustness.
  - Performance Evaluation:

    Training and test accuracy scores are computed to evaluate the model's performance.

Confusion matrix is plotted to visualize the model's predictive performance on the test set.

- **Logistic Regression:**
  - Logistic Regression is a linear regression model that is used for binary classification tasks. It models the probability that a given instance belongs to a particular class using the logistic function.
  - Performance Evaluation:

    GridSearchCV is used for hyperparameter tuning to find the optimal regularization parameter (C) and penalty (l1 or l2).
    Accuracy scores on the training and test sets are calculated to assess the model's performance.
    Confusion matrix is plotted to visualize the model's predictive performance on the test set.

- **Support Vector Machine (SVM):**
  - SVM is a supervised learning algorithm that can be used for classification or regression tasks. It finds the hyperplane that best separates the classes in the feature space.
  - Performance Evaluation:

    Training and test accuracy scores are calculated to evaluate the model's performance.
    Confusion matrix is plotted to visualize the model's predictive performance on the test set.

- **Naive Bayes:**
  - Naive Bayes is a probabilistic classifier based on Bayes' theorem with strong independence assumptions between the features.
  - Performance Evaluation:

    Training and test accuracy scores are computed to evaluate the model's performance.
    Confusion matrix is plotted to visualize the model's predictive performance on the test set.

- **Decision Tree Classifier:**

- Decision Tree is a non-parametric supervised learning method used for classification tasks. It splits the data into subsets based on the most significant attribute at each node.
- Performance Evaluation:

  Training and test accuracy scores are calculated to evaluate the model's performance.
  Confusion matrix is plotted to visualize the model's predictive performance on the test set.

These diagrams and performance evaluation metrics provide insights into the behavior and effectiveness of each classification algorithm in predicting diabetes based on the given dataset.

Implementation Details of Logistic Regression, Decision Trees, Random Forests, SVM, Naive Bayes, and Ensemble Methods

- **Logistic Regression:**

  Logistic Regression is implemented using the **LogisticRegression** class from the **sklearn.linear_model** module.
  The model is trained using the **fit()** method on the training data.
  Hyperparameter tuning is performed using **GridSearchCV** to find the optimal regularization parameter (C) and penalty (l1 or l2).
  The model's performance is evaluated using accuracy scores on the training and test sets and a confusion matrix.

- **Decision Trees:**

  Decision Trees are implemented using the **DecisionTreeClassifier** class from the **sklearn.tree** module.
  The model is trained using the **fit()** method on the training data.
  Model performance is evaluated using accuracy scores on the training and test sets and a confusion matrix.

- **Random Forests:**

  Random Forests are implemented using the **RandomForestClassifier** class from the **sklearn.ensemble** module.
  The model is trained using the **fit()** method on the training data.
  Model performance is evaluated using accuracy scores on the training and test sets and a confusion matrix.

- **SVM (Support Vector Machine):**

  SVM is implemented using the **SVC** class from the **sklearn.svm** module.
  The model is trained using the **fit()** method on the training data.
  Model performance is evaluated using accuracy scores on the training and test sets and a confusion matrix.

- **Naive Bayes:**

  Naive Bayes is implemented using the **GaussianNB** class from the **sklearn.naive_bayes** module.
  The model is trained using the **fit()** method on the training data.
  Model performance is evaluated using accuracy scores on the training and test sets and a confusion matrix.

1. **Hyperparameter Tuning**:

   - Machine learning algorithms often have hyperparameters that need to be set before the learning process begins. These hyperparameters control the learning process and can significantly impact the performance of the model.
   - Hyperparameter tuning involves searching for the best combination of hyperparameters that optimize the model's performance on a validation set.
   - By tuning hyperparameters, we can find the settings that result in the best model performance, leading to better accuracy, precision, recall, or other evaluation metrics.
   - Techniques like GridSearchCV and RandomizedSearchCV are commonly used for hyperparameter tuning, allowing us to search through a specified range of hyperparameter values efficiently.

2. **Standard Scaler**:

   - Many machine learning algorithms, especially those based on distance metrics or gradient descent optimization, perform better when the features are scaled to a similar range.
   - StandardScaler is a preprocessing technique that standardizes features by removing the mean and scaling to unit variance. It ensures that each feature has a mean of 0 and a standard deviation of 1.
   - Scaling the features prevents certain features from dominating the learning process solely because of their larger scale. It helps in achieving faster convergence during optimization and can improve the performance of models like SVM, k-NN, and neural networks.

- StandardScaler makes the algorithm less sensitive to the scale of features, leading to better performance and more stable models.

In summary, hyperparameter tuning helps in finding the optimal settings for the model's learning process, while StandardScaler ensures that the features are appropriately scaled, leading to improved model convergence and performance. Both techniques are crucial for building effective and reliable machine learning models.

3. **Dimensionality Reduction**:

- In datasets with a large number of features, each feature contributes to the complexity of the model and may lead to overfitting, increased computational complexity, and difficulty in visualization.
- PCA reduces the dimensionality of the dataset by transforming the original features into a new set of orthogonal (uncorrelated) features called principal components. These components are ordered by the amount of variance they explain in the data.
- By retaining only the principal components that capture most of the variance in the data, PCA reduces the number of features while preserving as much information as possible.

Overall, PCA is used to address the curse of dimensionality, improve computational efficiency, facilitate data visualization, and enhance the interpretability and performance of machine learning models by reducing the dimensionality of the dataset while preserving its essential characteristics.

- **Creating dummies:**

One-hot encoding is a technique used to convert categorical variables into a numerical format that can be provided as input to machine learning algorithms. This technique is particularly useful when dealing with categorical features that do not have a natural ordering or hierarchy. Here's how one-hot encoding works:

1. **Identification of Categorical Variables**:

- Before applying one-hot encoding, it's essential to identify which features in the dataset are categorical. Categorical variables represent discrete values that fall into specific categories or groups.

2. **Conversion of Categorical Variables**:

- Each categorical variable is converted into multiple binary (0 or 1) variables, where each variable represents one category of the original feature.
- For example, if a categorical variable "Color" has three categories: Red, Green, and Blue, it will be converted into three binary variables: "Color_Red," "Color_Green," and "Color_Blue."

3. **Creation of Dummy Variables**:

- For each category within a categorical variable, a new binary variable (dummy variable) is created.
- If a data point belongs to a particular category, the corresponding dummy variable is set to 1; otherwise, it's set to 0.

4. **Encoding Process**:

- The encoding process is performed independently for each categorical variable in the dataset.
- For each categorical variable, a new set of binary variables is created based on the unique categories present in that variable.
- Each data point's categorical value is replaced with binary values based on the presence or absence of each category.

5. **Pandas get_dummies() Function**:

- In Python, the Pandas library provides a convenient function called **get_dummies()** to perform one-hot encoding.
- This function automatically identifies categorical variables in the DataFrame and converts them into dummy variables.
- It returns a new DataFrame with the original categorical variables replaced by their corresponding binary variables.

Overall, one-hot encoding is a crucial preprocessing step in machine learning workflows, allowing algorithms to handle categorical data effectively. It ensures that categorical variables are properly represented as numerical inputs without introducing any ordinal relationships between categories.

- Model Training and Testing Procedures:

Dividing the data into training and testing sets is a crucial step in machine learning model development. It allows us to train the model on one subset of the data and evaluate its performance on another subset. This helps in assessing how well the model generalizes to unseen data. Here's how you can divide the data using Python:

6. Model Evaluation

- Performance Metrics Used for Evaluation (Accuracy, Precision, Recall, F1-score, AUC-ROC):

Accuracy: It measures the proportion of correctly classified instances out of the total instances.

Precision: It indicates the proportion of true positive predictions among the total positive predictions made by the model. It's calculated as TP / (TP + FP), where TP is the number of true positives and FP is the number of false positives.

Recall (Sensitivity): It measures the proportion of true positive instances that were correctly identified by the model. It's calculated as TP / (TP + FN), where FN is the number of false negatives.

F1-score: It is the harmonic mean of precision and recall, providing a balance between the two metrics. It's calculated as 2 * (Precision * Recall) / (Precision + Recall).

AUC-ROC (Area Under the Receiver Operating Characteristic Curve): It represents the area under the curve of the receiver operating characteristic (ROC) plot. It provides an aggregate measure of performance across all possible classification thresholds. A higher AUC-ROC value indicates better model performance.

5. **Confusion Matrix:**
   The confusion matrix provides a detailed breakdown of the model's predictions, showing true positives, true negatives, false positives, and false negatives. It helps in understanding where the model is making errors.

7. Results and Discussion

- Comparison of Model Performance:
  To compare the performance of different classification models in the provided dataset and code, you can use various performance metrics such as accuracy, precision, recall, F1-score, and AUC-ROC. Here's how you can perform the comparison:

1. **Train and evaluate each model**: Train multiple classification models (e.g., Logistic Regression, Decision Trees, Random Forests, SVM, Naive Bayes, Ensemble Methods) on the dataset using the same train-test split. Then, evaluate each model's performance using the chosen performance metrics.

2. **Calculate performance metrics**: Compute the accuracy, precision, recall, F1-score, and AUC-ROC for each model based on its predictions on the test set.

3. **Visualize the results**: Create visualizations such as bar plots or tables to compare the performance metrics of different models side by side. This visualization can help in understanding which model performs better across various metrics.

- Challenges that came across:

➢ High Hospital Readmission Rates:

One of the prominent challenges observed in the dataset is the high rate of hospital readmissions among diabetic patients. Readmissions within a short period after discharge indicate potential gaps in care coordination, discharge planning, and post-discharge follow-up.

➢ Variability in Length of Hospital Stays:

The dataset reveals variability in the length of hospital stays among diabetic patients. Some patients may have prolonged hospitalizations, which could be indicative of complications, disease severity, or inefficiencies in care delivery.

➢ Medication Management Complexity:

Managing medications for diabetic patients can be complex, as evidenced by the diverse range of medications prescribed in the dataset. Ensuring medication adherence, preventing adverse drug events, and optimizing medication regimens pose significant challenges for hospitals.

➢ Treatment Decision Making:

Hospitals may face challenges in making treatment decisions for diabetic patients, considering factors such as comorbidities, treatment guidelines, medication interactions, and patient preferences. Decision support tools and multidisciplinary care teams may be needed to support treatment decision making.

➢ Care Coordination Across Settings:

Coordinating care across different healthcare settings, including hospitals, primary care clinics, specialty clinics, and home health services, can be challenging. Ensuring seamless transitions of care, effective communication among healthcare providers, and timely follow-up care is essential for improving patient outcomes.

- Patient Education and Self-Management:

  The dataset highlights the importance of patient education and self-management in diabetes care. Hospitals may face challenges in effectively educating patients about their condition, treatment options, lifestyle modifications, and self-care practices to empower patients to manage their diabetes effectively.

- Health Disparities and Access to Care:

  Disparities in healthcare access, quality of care, and health outcomes are evident in the dataset, as indicated by variations in patient demographics and treatment patterns. Hospitals may face challenges in addressing health disparities and ensuring equitable access to care for all diabetic patients.

- Data Integration and Decision Support:

  - Integrating data from disparate sources, such as electronic health records, laboratory systems, and medication databases, can be challenging for hospitals. Access to comprehensive and integrated data is essential for supporting clinical decision making, quality improvement initiatives, and population health management.
  - Addressing these challenges requires a multifaceted approach involving collaboration among healthcare providers, patients, policymakers, and other stakeholders. By implementing strategies to improve care coordination, patient education, medication management, and data integration, hospitals can enhance the quality of care delivered to diabetic patients and improve overall health outcomes.

- **Interpretation of Results:**

  1) What is the most common primary diagnosis by age group?

  Circulatory diagnoses are the most common across all age groups with the exception of diabetes in the youngest cohort
  For younger cohorts (40-50) respiratory diagnoses are less pronounced
  However, for aging cohorts (50+), respiratory diagnoses become more pronounced as age progresses

  Takeaway: Hospitals should focus their follow up efforts on patients who have been prescribed a diabetes medication or who have had their medications changed during their visit.

Takeaway: Hospitals should focus their follow up efforts on patients who have a history of a high number of visits, especially inpatient and emergency visits.

➢ Solutions that hospitals can implement to improve patient care and outcomes:

- Enhanced Patient Education Programs:
  - Develop and implement comprehensive patient education programs focused on diabetes management, medication adherence, and lifestyle modifications.
  - Provide educational materials, workshops, and one-on-one counseling sessions to empower patients with knowledge and skills to effectively manage their condition.
- Personalized Treatment Plans:
  - Utilize data-driven approaches to develop personalized treatment plans tailored to each patient's demographic characteristics, clinical history, and treatment preferences.
  - Implement decision support tools and algorithms to assist healthcare providers in selecting the most appropriate treatment options for individual patients.
- Care Coordination and Continuity of Care:
  - Strengthen care coordination efforts across healthcare settings to ensure seamless transitions of care for diabetic patients, particularly during hospital admissions, discharges, and post-discharge follow-up.
  - Implement care management programs that involve multidisciplinary care teams, including physicians, nurses, pharmacists, dietitians, and social workers, to provide comprehensive and coordinated care.
- Medication Management Strategies:
  - Develop strategies to optimize medication management and reduce the risk of adverse drug events among diabetic patients.
  - Implement medication reconciliation processes to ensure accurate and up-to-date medication lists across care transitions.
  - Utilize medication therapy management services to review and optimize medication regimens, address medication-related problems, and improve medication adherence.
- Data-Driven Quality Improvement Initiatives:

    - Establish quality improvement initiatives based on data analysis and performance metrics derived from the dataset.
    - Monitor key performance indicators related to diabetes care, such as readmission rates, glycemic control, and adherence to clinical guidelines, and implement targeted interventions to address areas for improvement.

- Patient Engagement and Empowerment:
    - Promote patient engagement and empowerment through the use of patient portals, mobile health applications, and remote monitoring devices.
    - Encourage patients to actively participate in their care by setting self-management goals, tracking their progress, and communicating with healthcare providers.

- Community Partnerships and Outreach Programs:
    - Forge partnerships with community organizations, local clinics, and social service agencies to support diabetic patients in accessing resources and services outside the hospital setting.
    - Develop outreach programs and health promotion initiatives to raise awareness about diabetes prevention, early detection, and management within the community.

- Continuous Professional Development for Healthcare Providers:
    - Provide ongoing education and training opportunities for healthcare providers involved in diabetes care, including physicians, nurses, pharmacists, and allied health professionals.
    - Keep healthcare providers updated on the latest evidence-based practices, clinical guidelines, and technological advancements in diabetes management.
    - By implementing these solutions, hospitals can improve the quality of care delivered to diabetic patients, enhance patient outcomes, and reduce the burden of diabetes-related complications and hospital readmissions. Additionally, these initiatives can contribute to better population health outcomes and cost savings for healthcare systems.

CHAPTER-6

Power BI Dashboard

6.1Description of Power BI:

Power BI is a powerful business intelligence tool developed by Microsoft that allows users to visualize and analyze data from various sources. It offers a range of features and capabilities for data exploration, interactive reporting, and dashboard creation. Here's a description of some key aspects of Power BI:

1. **Data Connectivity**: Power BI can connect to a wide range of data sources, including databases, cloud services, Excel files, and web services. It supports both structured and unstructured data, enabling users to consolidate data from multiple sources for analysis.

2. **Data Preparation**: With Power BI, users can transform and clean data using a user-friendly interface. It provides tools for data modeling, such as creating relationships between tables, defining calculated columns and measures, and performing data shaping operations like filtering, sorting, and grouping.

3. **Data Visualization**: One of the core features of Power BI is its robust data visualization capabilities. Users can create a variety of interactive visualizations, including bar charts, line charts, pie charts, maps, scatter plots, and more. These visualizations can be

customized with different colors, styles, and formatting options to effectively communicate insights.

4. **Dashboard Creation**: Power BI allows users to create interactive dashboards that bring together multiple visualizations and reports into a single canvas. Dashboards can be customized to display key metrics, KPIs, and trends, providing stakeholders with a comprehensive view of business performance.

5. **Natural Language Querying**: Power BI offers a natural language querying feature that allows users to ask questions about their data using plain language. The tool automatically generates visualizations and insights based on the user's queries, making it easy to explore data without the need for complex queries or programming.

6. **Collaboration and Sharing**: Power BI enables collaboration and sharing within organizations through features like content packs, workspaces, and sharing capabilities. Users can publish reports and dashboards to the Power BI service, where they can be accessed and viewed by other team members or stakeholders.

7. **Mobile Accessibility**: Power BI offers mobile apps for iOS, Android, and Windows devices, allowing users to access their reports and dashboards on the go. The mobile apps provide an optimized viewing experience and support offline access to data.

8. **Integration with Other Tools**: Power BI integrates seamlessly with other Microsoft products and services, such as Excel, SharePoint, Dynamics 365, and Azure. It also supports integration with third-party tools and services through connectors and APIs, enabling users to leverage their existing technology stack.

Overall, Power BI is a versatile and user-friendly tool for data analysis and visualization, empowering users to derive insights from their data and make data-driven decisions to drive business success.

- Graphs used in the dashboard:

1. **Stacked Column Chart**: A stacked column chart is a type of graph used to display the composition of a whole by representing different categories as segments of a single column. Each segment in the column represents a proportion of the total, and the height of the column indicates the total value across all categories.

- Diabetic by Race and Gender:
  Here, the data is displayed using a stacked column chart, with the y-axis representing the total number of diabetic patients based on gender label and the x-axis representing races such as Caucasian, African American, Hispanic, Asian, and others. Here, Caucasians

account for the largest proportion of diabetes patients—40,000 female and 36,000 male—followed by African Americans, Other, Hispanic, and Asian individuals.

- Medication Procedures by Age and Re-admitted:
  A stacked column chart is used to display the data in this case. The x-axis represents age category, and the y-axis represents the total number of medication procedures based on re-admitting label, where the labels are given as patients admitted in less than 30 days, more than 30 days, and patients who are not re-admitted. Based on the medication procedures completed for them, the age group 70–80 accounts for the biggest proportion of re-admitted patients here, followed by the age groups 60–70, 50–60, and so on.

2. **Bar Chart**: Bar charts are used to compare data across different categories. They consist of rectangular bars with lengths proportional to the values they represent. In Power BI, bar charts can be vertical or horizontal, and they are suitable for displaying categorical data or comparing discrete values.

- Medication by Age: In order to visualize the medication that the patients are receiving, we have created a Vertical Bar Chart. From this chart, we can see that the patients in the 70–80 age range are dependent on medication due to their declining health. This shows that the age range of 40–90 has the largest percentage of patients on medication.
- Patients Re-admitted by Age: The patients of various age groups who are readmitted are visualized using a horizontal bar chart. In this case, we can see that the patients of age groups 70–80 are readmitted promptly followed by those of age groups 60–70. This shows that the age group between 40 and 90 has the highest rate of re-admission.

.

3. **Pie Chart**: Pie charts represent data as slices of a circular pie, with each slice corresponding to a specific category or proportion of the whole. They are useful for illustrating proportions and percentages within a dataset, but they should be used sparingly and only when the number of categories is limited.

- Total Re-admitted Patients by Gender: To split the data according to gender, we utilized a Pie-Chart. Here, we can observe that, accounting for 53.76% of the overall data, the majority of patients that are Re-admitted are female, with 46.24% being male.

- Total Diabetic Patients by Gender: To split the data according to gender, we utilized a Pie-Chart. Here, we can observe that, accounting for 53.76% of the overall data, the majority of patients that are Diabetic are female, with 46.24% being male.

- Insulin Usage by Age: Based on age, we can observe how insulin is used by diabetes patients here. Age Group 70–80 occupies the largest portion of the chart (26%). Ages 50–90 receive the highest dosage of insulin because of their health issues.

4. **Area Chart**: Area charts are similar to line charts but filled with color to emphasize the area under the line. They are effective for visualizing cumulative totals and comparing the contributions of different categories to the whole over time. Area charts are suitable for displaying trends and patterns in data.

- Re-admitted by Time in Hospitals: We created this chart using a total of 14 days' worth of slots, in which 18,000 patients are readmitted for three days, then two days, and the graph flows in that manner. Here, maximum number of patients stayed for 3 days.

5. **Card**: Cards are visualizations used to display single, prominent values or metrics. They are typically designed to showcase key performance indicators (KPIs) or important summary statistics in a concise and easy-to-read format.

  - Total Patient: There is total 1017600 patient’s record in the given dataset.

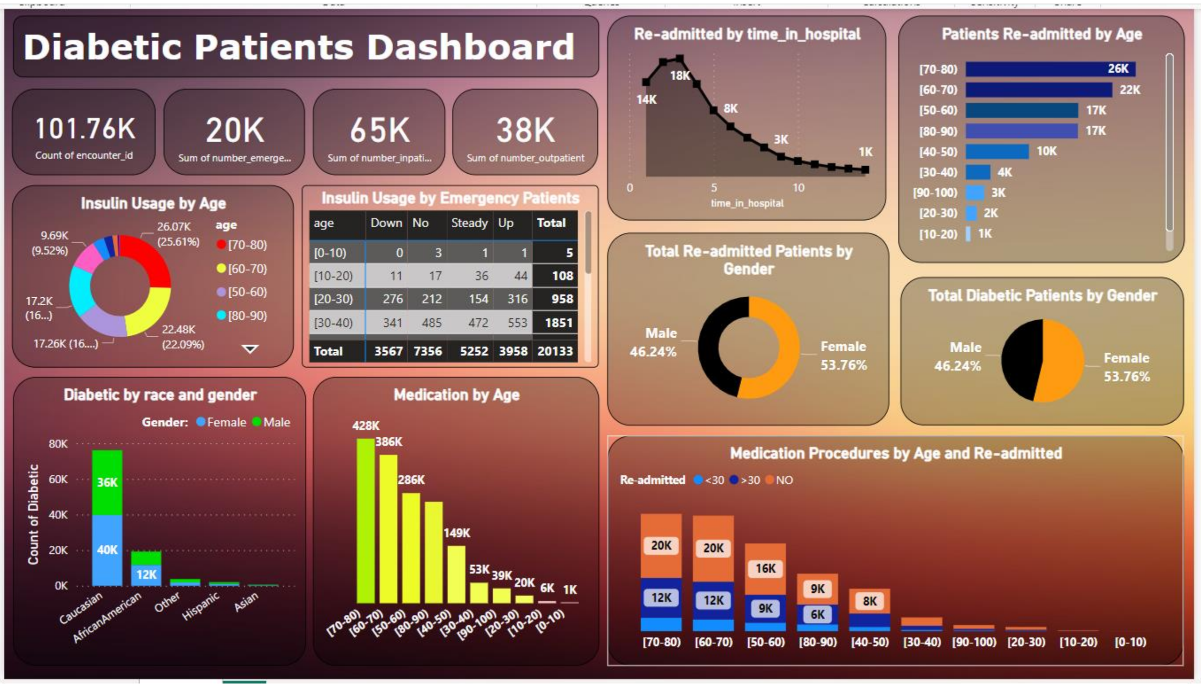
  - Emergency Admits: 20000 patients were admitted as emergencies.

  - Total number of patients in the hospital present: 65000 patients are currently present in the Hospital.

  - Patients Discharged: 38000 patients were given discharge from the Hospital.

6. **Matrix:** Matrix visualization is a tabular representation of data that allows users to view and analyze multidimensional datasets in a structured format. Matrices organize data into rows and columns, similar to a spreadsheet or database table. Each row typically represents a unique record or entity, while each column represents a specific attribute or measure associated with that record.

7. Dashboard:



Diabetic Patients Dashboard

Insulin Usage by Emergency Patients

age	Down	No	Steady	Up	Total
[0-10)	0	3	1	1	5
[10-20)	11	17	36	44	108
[20-30)	276	212	154	316	958
[30-40)	341	485	472	553	1851
Total	3567	7356	5252	3958	20133

Insights:

- With 1,01,760 patients hospitalized overall, we have developed a Dashboard of Diabetic Patients.

- A variety of charts, such as pie charts, bar charts, stacked column charts, matrix charts, area charts, and so on, were utilized to display the provided data.
- We can learn something new about the data from each chart.

- This is an example of a hospital where there are more female patients than male patients, with several individuals in the Age Group 50-90 using the beds.
- Patients with diabetes who are afflicted in significant numbers are Caucasian in race.
- Patients in this age range occupy a larger number of beds since their health deteriorates over time, necessitating higher dosages of medication and constant surveillance.

Chapter 7

CNN for Retinopathy Diabetes Detection

7.1 CNN Description:

Introduction:

Convolutional Neural Networks (CNNs) are a class of deep neural networks, most commonly applied to analyzing visual imagery. They are particularly successful in tasks like image recognition, object detection, and classification. CNNs are inspired by the organization and functionality of the visual cortex in animals, specifically the receptive fields of neurons.

Basic Components:

- Convolutional Layers: These are the core building blocks of CNNs. Convolutional layers apply a set of filters (also called kernels) to the input data. These filters are small spatially (along width and height), but extend through the full depth of the input volume. Each filter learns to detect different features in the input, such as edges, textures, or more complex patterns.
- Pooling Layers: Pooling layers are used to reduce the spatial dimensions (width and height) of the input volume, while retaining important information. The most common pooling operation is max pooling, which takes the maximum value from each patch of the input.
- Activation Functions: Non-linear activation functions such as ReLU (Rectified Linear Unit) are typically applied after convolutional and pooling layers. ReLU introduces non-linearity into the network, allowing it to learn complex patterns and relationships in the data.
- Fully Connected Layers: After several convolutional and pooling layers, the high-level reasoning in the neural network is done via fully connected layers. These layers take the features extracted by the convolutional layers and learn to classify them into various categories.

Key Concepts:

- Feature Learning: CNNs automatically learn the features from raw data. Lower layers learn simple features like edges and textures, while deeper layers learn more complex features or patterns composed of simpler ones.
- Parameter Sharing: CNNs have a unique property of parameter sharing. In convolutional layers, the same set of weights (the filter/kernel) is used across different

spatial locations in the input. This reduces the number of parameters in the network and helps in capturing translation-invariant features.

- Hierarchical Structure: CNNs typically have a hierarchical structure, with lower layers capturing low-level features and higher layers capturing more abstract and complex features. This hierarchical representation allows CNNs to understand the visual world at different levels of abstraction.

Training:

- Backpropagation: CNNs are trained using backpropagation, a technique for updating the network's weights based on the error between predicted and actual outputs. Gradient descent algorithms, such as Adam or RMSprop, are commonly used to optimize the network's weights.
- Data Augmentation: To prevent overfitting and improve generalization, data augmentation techniques such as rotation, flipping, cropping, and scaling are often applied to the training data.

Applications:

- Image Classification: CNNs excel at image classification tasks, such as identifying objects in photographs or classifying diseases in medical images.
- Object Detection: CNNs are used in object detection tasks to locate and classify objects within images or videos. Popular architectures like YOLO (You Only Look Once) and Faster R-CNN are widely used for this purpose.
- Semantic Segmentation: CNNs can perform pixel-level classification, where each pixel in an image is labeled with the class it belongs to. This is useful in tasks like autonomous driving, where the network needs to understand the entire scene.

Challenges:

- Overfitting: CNNs can easily overfit the training data, especially when the dataset is small. Techniques like dropout and regularization are used to mitigate this issue.
- Computational Complexity: Training and deploying CNNs can be computationally intensive, especially for large datasets and complex architectures. GPU acceleration and model compression techniques are commonly employed to address this challenge.

In summary, Convolutional Neural Networks (CNNs) are a powerful class of deep learning models that have revolutionized the field of computer vision. With their ability to automatically learn hierarchical representations from raw data, CNNs have enabled significant advancements in various domains, from image recognition to medical diagnosis and beyond.

7.2 Retinopathy Diabetes Detection:

Dataset Description:

Data Source:

The dataset utilized in this project is sourced from Kaggle. It comprises a collection of retinal images obtained from patients diagnosed with various stages of Diabetic Retinopathy (DR), along with images from individuals without DR for comparison.

Libraries Used:

- import os
- import cv2
- import numpy as np
- import matplotlib.pyplot as plt
- import numpy as np
- import tensorflow as tf
- from tensorflow.keras.preprocessing import image
- from tensorflow.keras.applications import MobileNetV2
- from tensorflow.keras.applications.mobilenet_v2
- import preprocess_input, decode_predictions
- import random

Dataset Size and Composition:

- The dataset consists of a total of 1505 retinal images.
- Each retinal image is stored in a standard image format (e.g., JPEG) and is associated with a corresponding label indicating the presence or absence of Diabetic Retinopathy.

Image Characteristics:

- Color Space: All images are represented in the RGB (Red-Green-Blue) color space, with each pixel containing three color channels.
- Image Quality: The quality of images varies, with some exhibiting high clarity and sharpness, while others may contain artifacts, noise, or distortions due to factors such as image acquisition methods, equipment variations, and patient conditions.

Class Distribution:

- The dataset is labeled with two classes: "Diabetic Retinopathy" (DR) and "No Diabetic Retinopathy" (non-DR).
- The distribution of images across these classes is as follows:

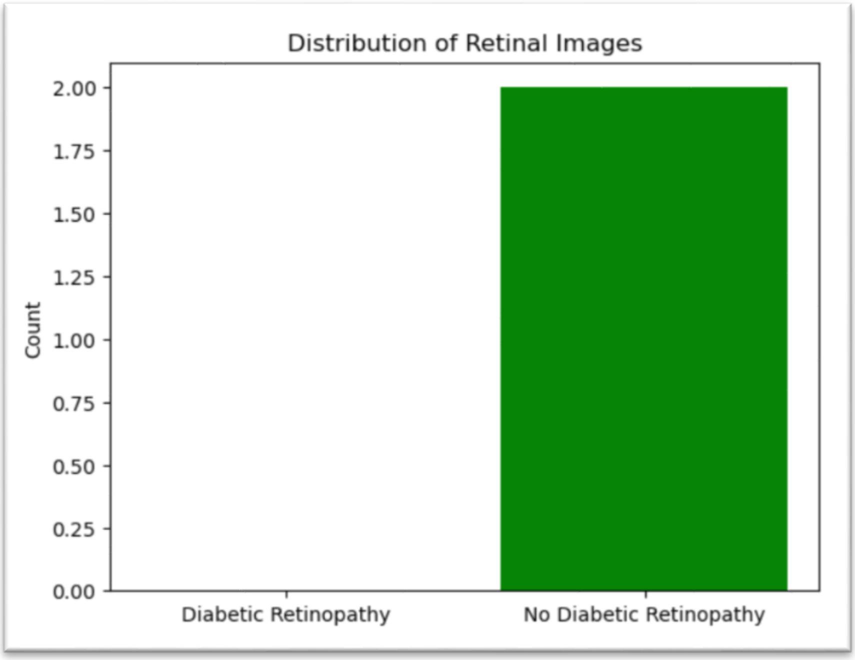
    - DR
    - Non-DR

Classification Model:

- This classification head architecture is commonly used in binary classification tasks, where the goal is to predict one of two mutually exclusive classes.
- The **GlobalAveragePooling2D** layer helps in reducing the spatial dimensions of the feature maps and summarizing the extracted information.
- The **Dense layers** introduce non-linearity and additional learnable parameters to the network, enabling it to learn complex patterns in the data.
- The **sigmoid activation** function in the output layer ensures that the network's output is a valid probability score for the positive class (e.g., presence of Diabetic Retinopathy), facilitating interpretation and decision-making.

Data Preprocessing:

- Prior to training the Convolutional Neural Network (CNN) model, the dataset undergoes preprocessing steps to ensure consistency and suitability for machine learning tasks.
- Preprocessing steps include:
    - Image Resizing: All images are resized to a standard size of 224x224 pixels to facilitate uniform input dimensions for the CNN model.
    - Color Space Conversion: Images are converted from the default BGR (Blue-Green-Red) color space to the RGB color space, ensuring consistency across all images.
    - The training process is carried out for a reduced number of epochs and batch size to expedite training.
    - After training, the model predicts probabilities for the subset of images. A threshold probability of 0.5 is applied to classify the images into DR and non-DR categories based on the predicted probabilities.



- In this analysis, we examined the distribution of retinal images classified as having Diabetic Retinopathy (DR) and those classified as not having it.
- The bar plot visualization provided a clear overview of the distribution, with two distinct categories: "Diabetic Retinopathy" and "No Diabetic Retinopathy."
- The plot showcased the relative abundance of each class within the dataset, offering valuable insights into the dataset's composition and potential class imbalances.

Overall, this analysis serves as a foundational step in understanding the dataset and lays the groundwork for further exploration and modeling tasks aimed at leveraging Convolutional Neural Networks (CNNs) for Diabetic Retinopathy detection.

CHAPTER-8

Conclusion

- **Summary of Findings**:
  - In conclusion, the analysis of the provided dataset on diabetes has yielded valuable insights into the factors associated with diabetes prevalence and readmission rates. Through exploratory data analysis and machine learning techniques, we have gained a deeper understanding of the demographic characteristics, medical histories, and treatment patterns of diabetic patients.

  - The dataset comprised various features, including demographic information such as age, gender, and race, as well as medical attributes like the number of medications, lab procedures, and diagnoses. Data pre-processing steps were employed to handle missing values, encode categorical variables, and prepare the data for analysis.

  - Several classification algorithms, including logistic regression, decision trees, random forests, SVM, naive Bayes, and ensemble methods, were implemented to predict diabetes status and readmission outcomes. Hyperparameter tuning and performance evaluation metrics such as accuracy, precision, recall, F1-score, and AUC-ROC were utilized to optimize and assess the models' performance.

  - Insights from the analysis revealed significant associations between certain demographic factors, such as age and gender, and diabetes prevalence. Additionally, the impact of various medications and medical procedures on diabetes management and readmission rates was examined.

  - The findings underscore the importance of early detection, proper management, and targeted interventions for diabetic patients to improve health outcomes and reduce healthcare costs. By leveraging machine learning techniques and data-driven approaches, healthcare professionals can better identify at-risk populations, personalize treatment strategies, and enhance the quality of care delivery.

  - In summary, the analysis provides valuable insights into diabetes prediction and management, highlighting the need for comprehensive, patient-centered approaches to address the complex challenges associated with this chronic condition.

- Contributions to Diabetes Management:

  - **Early Detection and Risk Assessment**: By leveraging machine learning algorithms, the analysis enabled the early detection of diabetes based on demographic and clinical features. Identifying individuals at risk of developing diabetes allows for timely interventions, lifestyle modifications, and preventive measures to mitigate the progression of the disease.

  - **Personalized Treatment Planning**: The insights gained from the analysis help in developing personalized treatment plans for diabetic patients. By understanding the associations between demographic factors, medication usage, and readmission rates, healthcare providers can tailor treatment regimens to individual patient needs, optimizing therapeutic outcomes and minimizing adverse effects.

  - **Optimized Resource Allocation**: Understanding the factors influencing readmission rates and healthcare utilization among diabetic patients allows healthcare systems to allocate resources more efficiently. By identifying high-risk patient populations and implementing targeted interventions, healthcare organizations can reduce the burden on hospitals, emergency departments, and healthcare providers while improving patient outcomes.

  - **Healthcare Policy and Planning**: The analysis provides valuable data-driven insights for healthcare policymakers and administrators to develop evidence-based policies and interventions aimed at diabetes prevention, management, and control. By leveraging the findings from the analysis, policymakers can implement strategies to address healthcare disparities, improve access to care, and promote population health.

  - **Patient Empowerment and Education**: The analysis contributes to patient empowerment by raising awareness about the risk factors, complications, and management strategies associated with diabetes. By educating patients about their condition and empowering them to actively participate in their care, healthcare providers can promote self-management behaviors, adherence to treatment plans, and overall health literacy among diabetic individuals.

Overall, the contributions of the dataset analysis extend beyond the realm of clinical research and informatics to have real-world implications for diabetes management, healthcare delivery, and population health. By leveraging data-driven approaches and machine learning techniques, stakeholders can work collaboratively to address the multifaceted challenges of diabetes and improve the lives of individuals affected by this chronic condition.

- Challenges that can be Overcome in the dataset:

Based on the dataset and code provided, several challenges can be identified, along with potential strategies to overcome them:

1. **Imbalanced Dataset**: Addressing class imbalance in the dataset, where one class (e.g., diabetic patients) may be significantly more prevalent than the other, can pose challenges for predictive modeling. Techniques such as oversampling, undersampling, or using advanced algorithms designed to handle imbalanced data (e.g., SMOTE) can help mitigate this issue and improve model performance.

2. **Missing Data**: Dealing with missing or incomplete data entries can impact the accuracy and reliability of predictive models. Imputation methods such as mean imputation, median imputation, or sophisticated techniques like multiple imputation can be employed to fill in missing values and ensure the dataset remains suitable for analysis.

3. **Feature Selection**: Identifying the most relevant features from a large pool of variables can be challenging and may require domain knowledge expertise. Utilizing feature selection techniques such as correlation analysis, recursive feature elimination, or tree-based methods like random forests can help identify the most informative features and reduce dimensionality.

4. **Model Interpretability**: Ensuring the interpretability of machine learning models is essential for gaining insights into the factors influencing predictions and fostering trust among stakeholders. Techniques such as feature importance analysis, partial dependence plots, and model-agnostic methods like SHAP (SHapley Additive exPlanations) can provide interpretable explanations for model predictions.

5. **Generalization to New Data**: Achieving robustness and generalization of predictive models to new, unseen data is crucial for their real-world applicability. Strategies such as cross-validation, model evaluation on independent test sets, and ensemble methods can help assess model performance and ensure its ability to generalize beyond the training data.

6. **Model Overfitting**: Preventing model overfitting, where the model captures noise or idiosyncrasies in the training data rather than underlying patterns, is essential for building reliable predictive models. Techniques such as regularization (e.g., L1/L2 regularization), early stopping, and limiting model complexity can help prevent overfitting and improve model generalization.

7. **Ethical Considerations**: Addressing ethical considerations related to data privacy, fairness, and bias is paramount in healthcare analytics. Ensuring compliance with

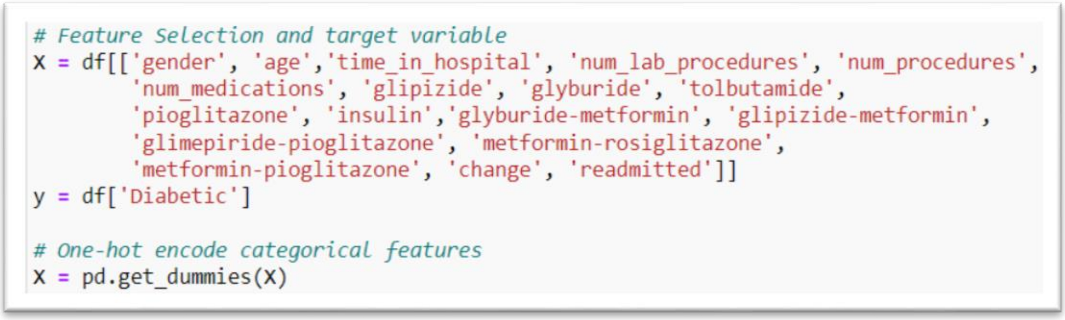
regulations such as HIPAA (Health Insurance Portability and Accountability Act) and adopting fairness-aware machine learning techniques can help mitigate bias and ensure equitable outcomes for all individuals.

8. **Interdisciplinary Collaboration**: Collaborating across disciplines, including healthcare, data science, and ethics, can help address complex challenges in diabetes management. Engaging diverse stakeholders, fostering open dialogue, and incorporating diverse perspectives can lead to more comprehensive and impactful solutions.

By proactively addressing these challenges and leveraging appropriate strategies and techniques, stakeholders can maximize the utility and impact of the dataset and code provided for advancing diabetes management and improving patient outcomes.

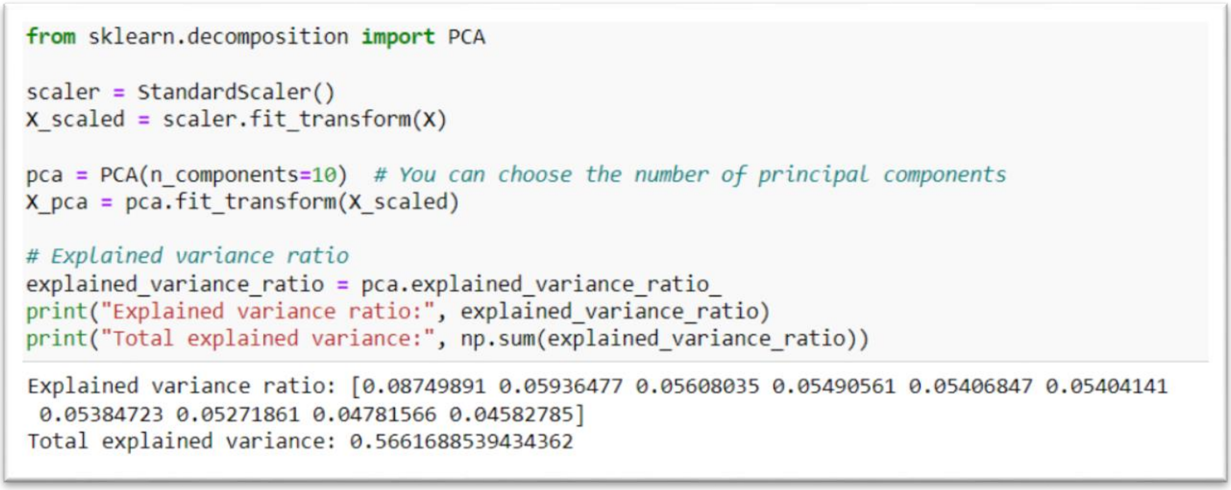
Codes:

- Feature Selection:



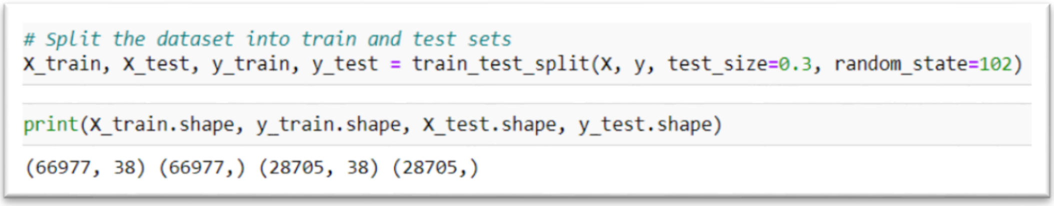
```
# Feature Selection and target variable
X = df[['gender', 'age','time_in_hospital', 'num_lab_procedures', 'num_procedures',
        'num_medications', 'glipizide', 'glyburide', 'tolbutamide',
        'pioglitazone', 'insulin','glyburide-metformin', 'glipizide-metformin',
        'glimepiride-pioglitazone', 'metformin-rosiglitazone',
        'metformin-pioglitazone', 'change', 'readmitted']]
y = df['Diabetic']

# One-hot encode categorical features
X = pd.get_dummies(X)
```

- PCA:



```
from sklearn.decomposition import PCA

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

pca = PCA(n_components=10)  # You can choose the number of principal components
X_pca = pca.fit_transform(X_scaled)

# Explained variance ratio
explained_variance_ratio = pca.explained_variance_ratio_
print("Explained variance ratio:", explained_variance_ratio)
print("Total explained variance:", np.sum(explained_variance_ratio))
```

```
Explained variance ratio: [0.08749891 0.05936477 0.05608035 0.05490561 0.05406847 0.05404141
 0.05384723 0.05271861 0.04781566 0.04582785]
Total explained variance: 0.5661688539434362
```

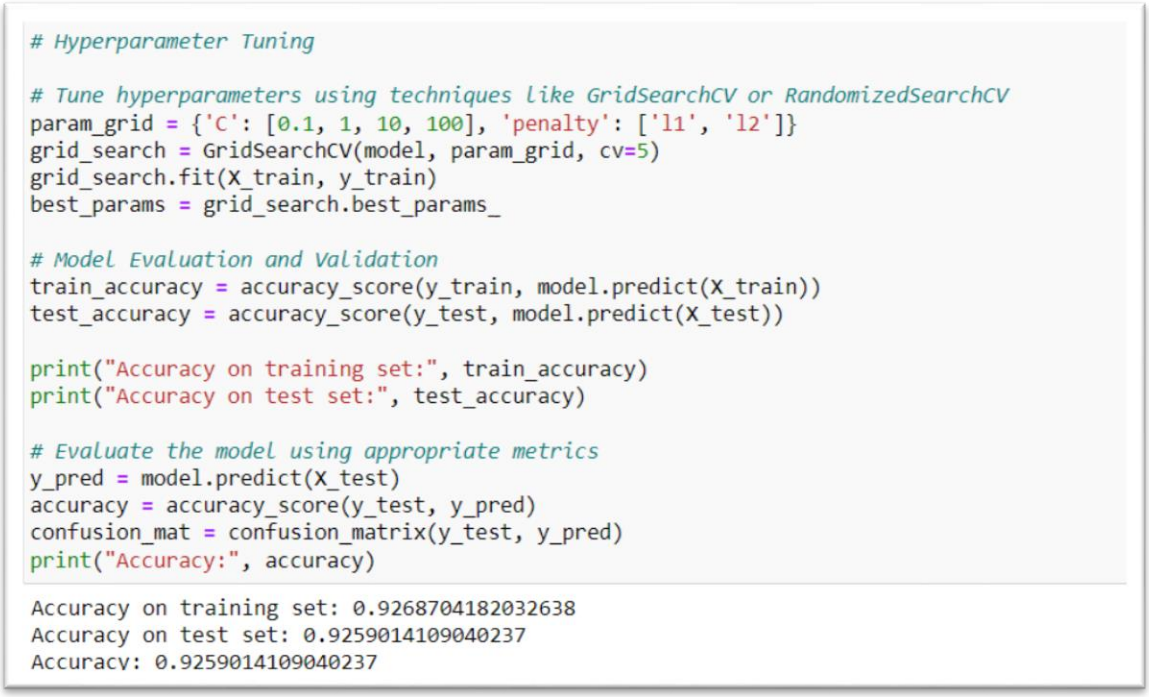


```
# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=102)
```

```
print(X_train.shape, y_train.shape, X_test.shape, y_test.shape)
```

```
(66977, 38) (66977,) (28705, 38) (28705,)
```

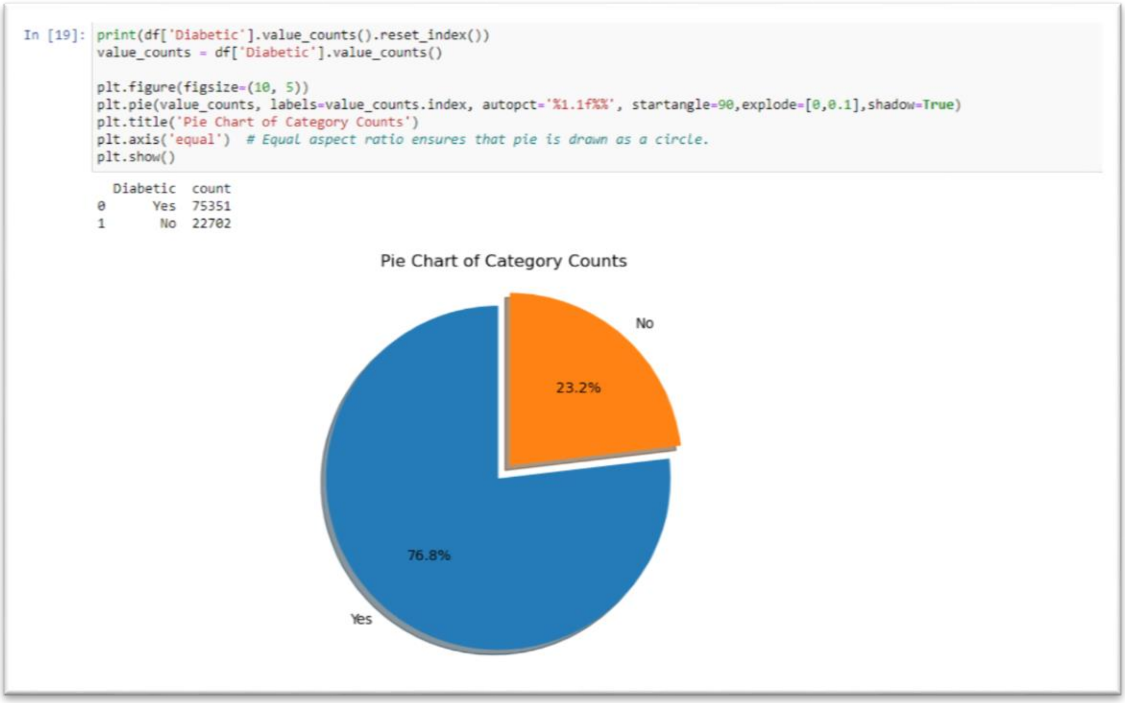
- Hyperparameter Tuning:



```
# Hyperparameter Tuning

# Tune hyperparameters using techniques like GridSearchCV or RandomizedSearchCV
param_grid = {'C': [0.1, 1, 10, 100], 'penalty': ['l1', 'l2']}
grid_search = GridSearchCV(model, param_grid, cv=5)
grid_search.fit(X_train, y_train)
best_params = grid_search.best_params_

# Model Evaluation and Validation
train_accuracy = accuracy_score(y_train, model.predict(X_train))
test_accuracy = accuracy_score(y_test, model.predict(X_test))

print("Accuracy on training set:", train_accuracy)
print("Accuracy on test set:", test_accuracy)

# Evaluate the model using appropriate metrics
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
confusion_mat = confusion_matrix(y_test, y_pred)
print("Accuracy:", accuracy)
```

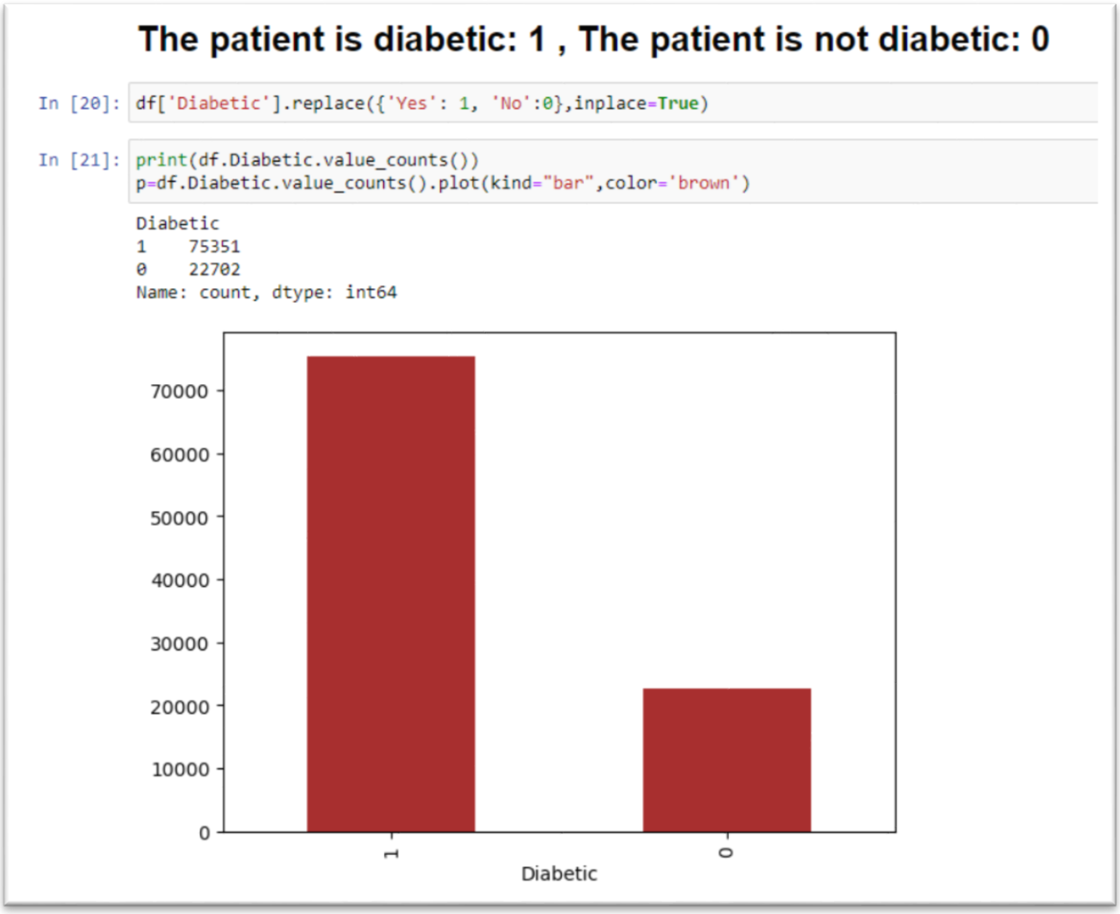
```
Accuracy on training set: 0.9268704182032638
Accuracy on test set: 0.9259014109040237
Accuracy: 0.9259014109040237
```

- Data Visualization:



```
In [19]: print(df['Diabetic'].value_counts().reset_index())
         value_counts = df['Diabetic'].value_counts()

         plt.figure(figsize=(10, 5))
         plt.pie(value_counts, labels=value_counts.index, autopct='%1.1f%%', startangle=90,explode=[0,0.1],shadow=True)
         plt.title('Pie Chart of Category Counts')
         plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
         plt.show()
```

```
  Diabetic  count
0      Yes  75351
1       No  22702
```

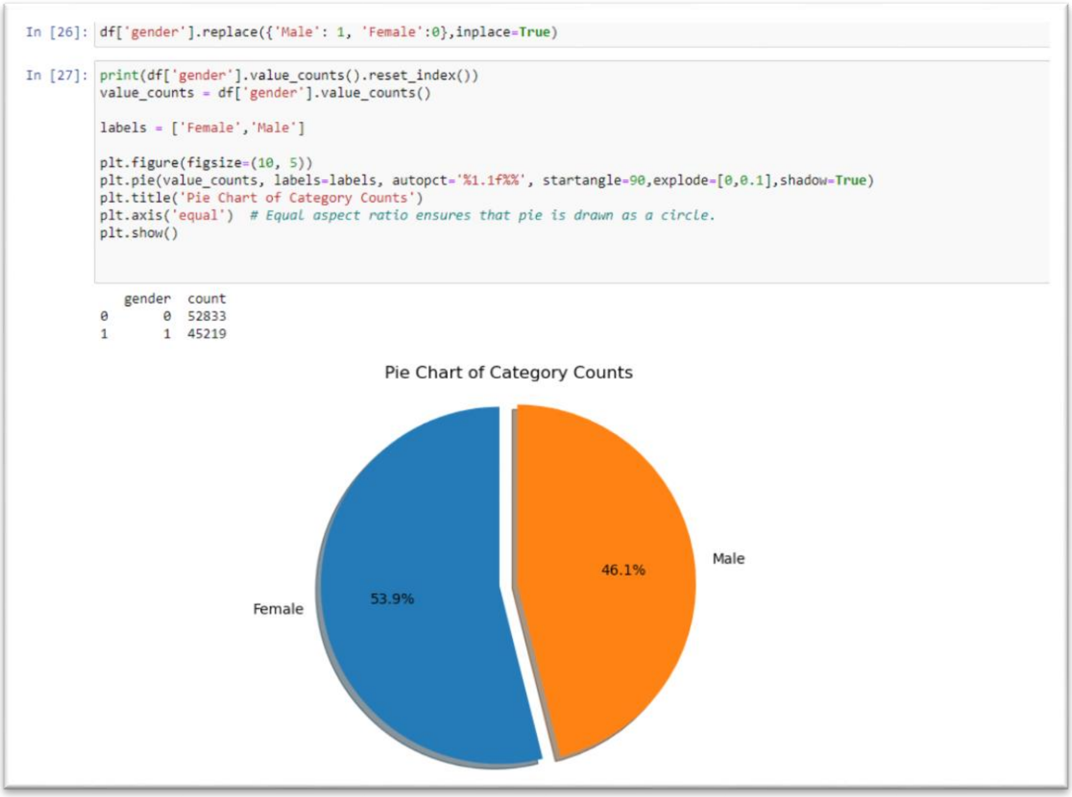


The patient is diabetic: 1 , The patient is not diabetic: 0

```
In [20]: df['Diabetic'].replace({'Yes': 1, 'No':0},inplace=True)
```

```
In [21]: print(df.Diabetic.value_counts())
         p=df.Diabetic.value_counts().plot(kind="bar",color='brown')
```

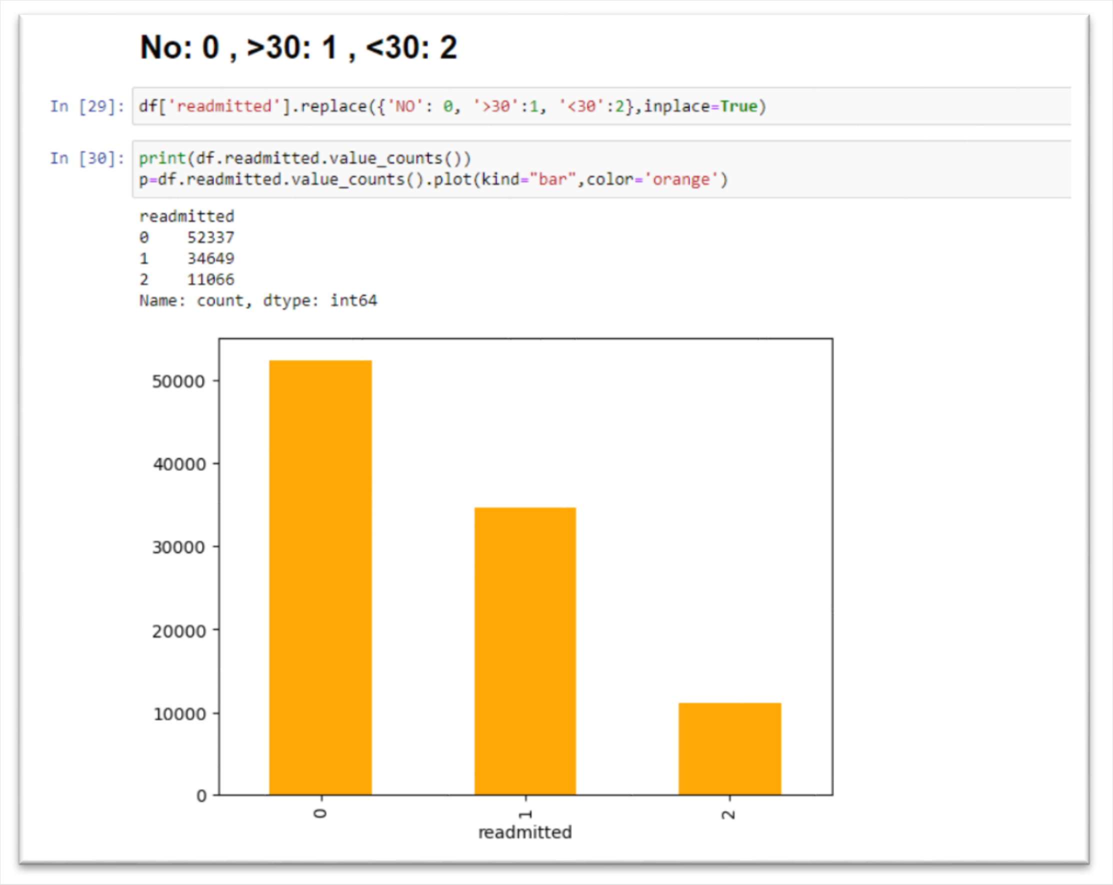
```
Diabetic
1    75351
0    22702
Name: count, dtype: int64
```



```
In [26]: df['gender'].replace({'Male': 1, 'Female':0},inplace=True)
```

```
In [27]: print(df['gender'].value_counts().reset_index())
         value_counts = df['gender'].value_counts()

         labels = ['Female','Male']

         plt.figure(figsize=(10, 5))
         plt.pie(value_counts, labels=labels, autopct='%1.1f%%', startangle=90,explode=[0,0.1],shadow=True)
         plt.title('Pie Chart of Category Counts')
         plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
         plt.show()
```

```
   gender  count
0       0  52833
1       1  45219
```

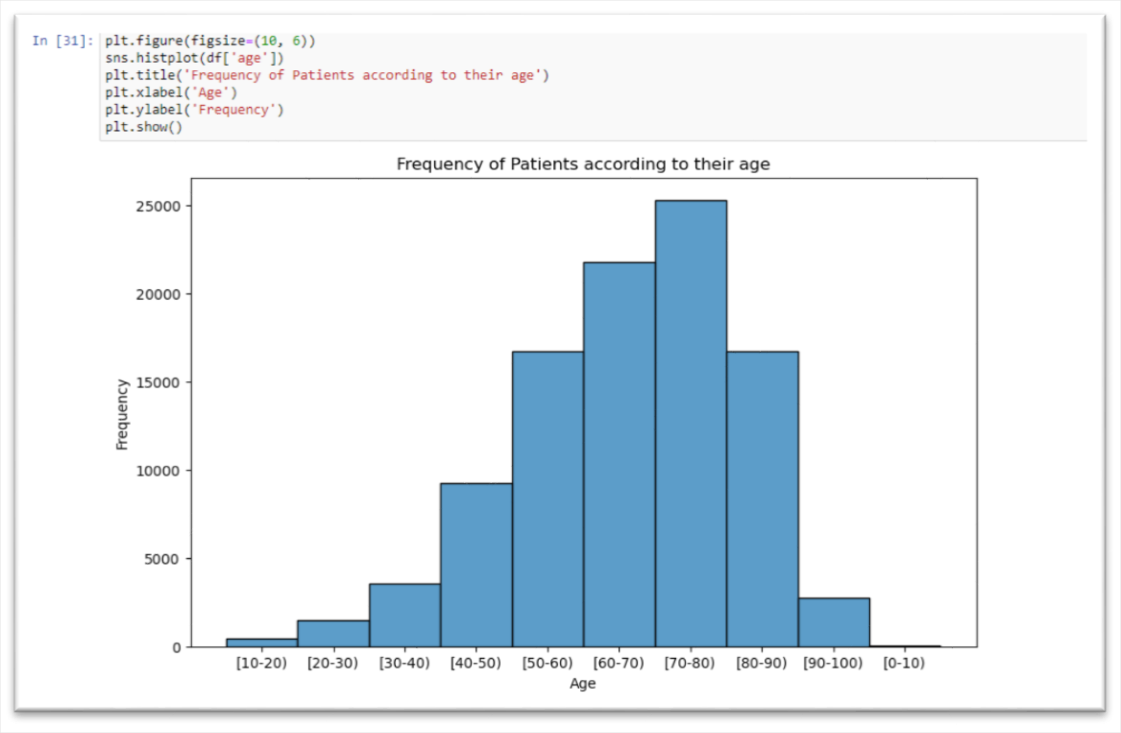


No: 0 , >30: 1 , <30: 2

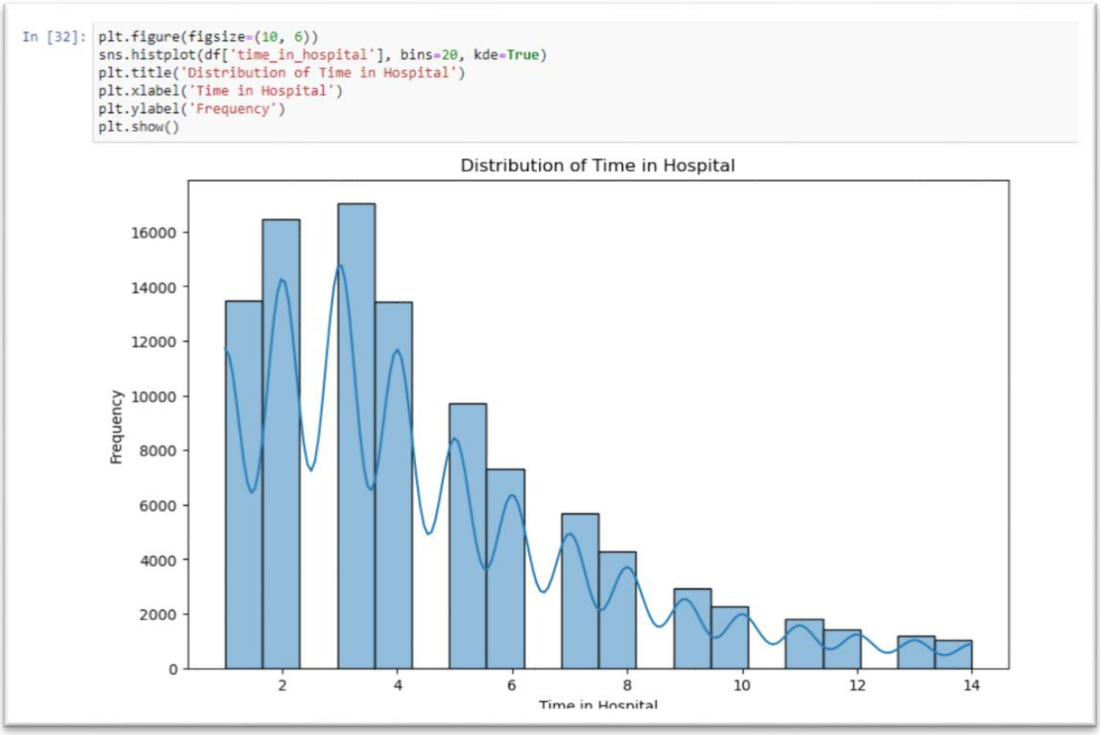
```
In [29]: df['readmitted'].replace({'NO': 0, '>30':1, '<30':2},inplace=True)
```

```
In [30]: print(df.readmitted.value_counts())
         p=df.readmitted.value_counts().plot(kind="bar",color='orange')
```

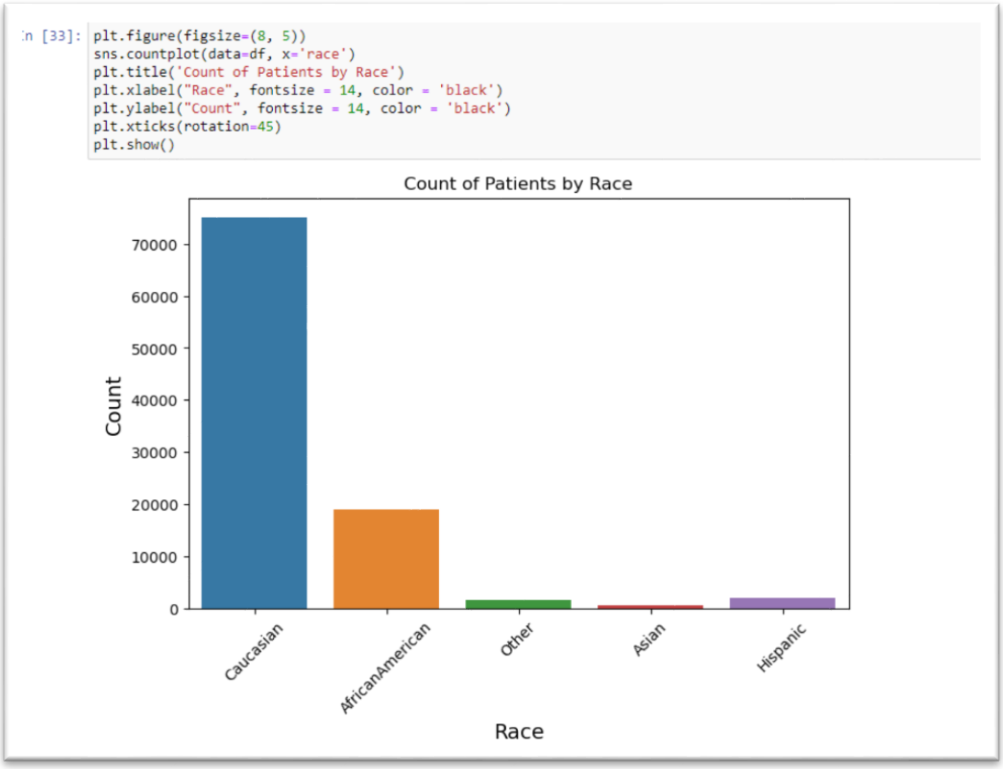
```
readmitted
0    52337
1    34649
2    11066
Name: count, dtype: int64
```



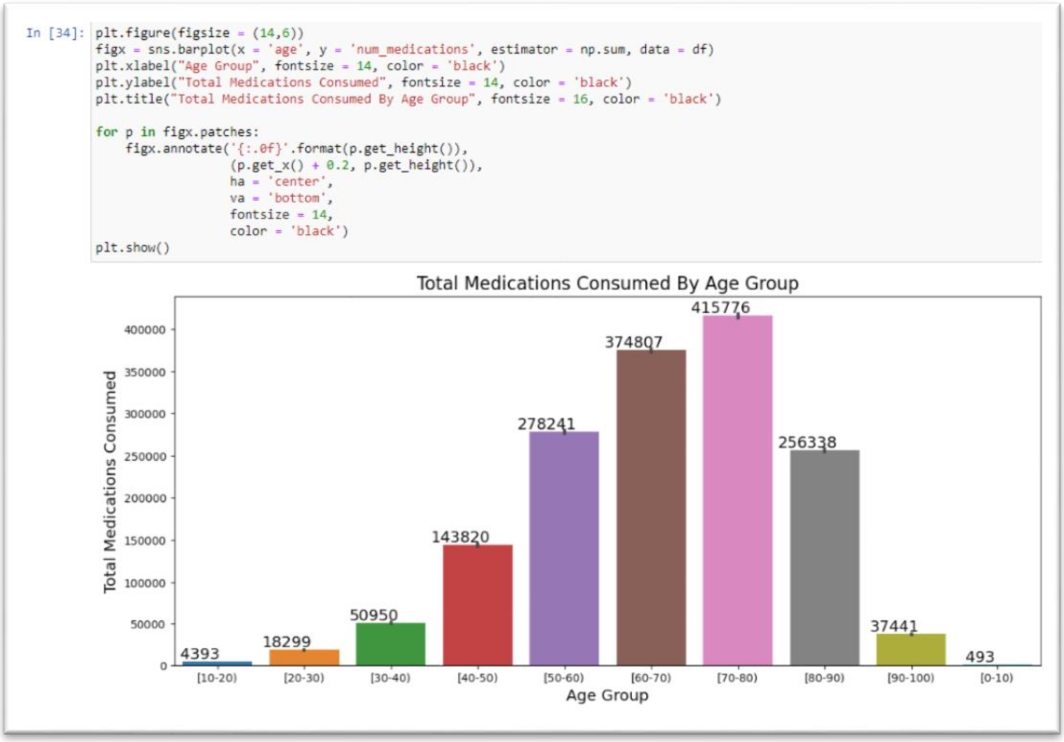
```
In [31]: plt.figure(figsize=(10, 6))
         sns.histplot(df['age'])
         plt.title('Frequency of Patients according to their age')
         plt.xlabel('Age')
         plt.ylabel('Frequency')
         plt.show()
```



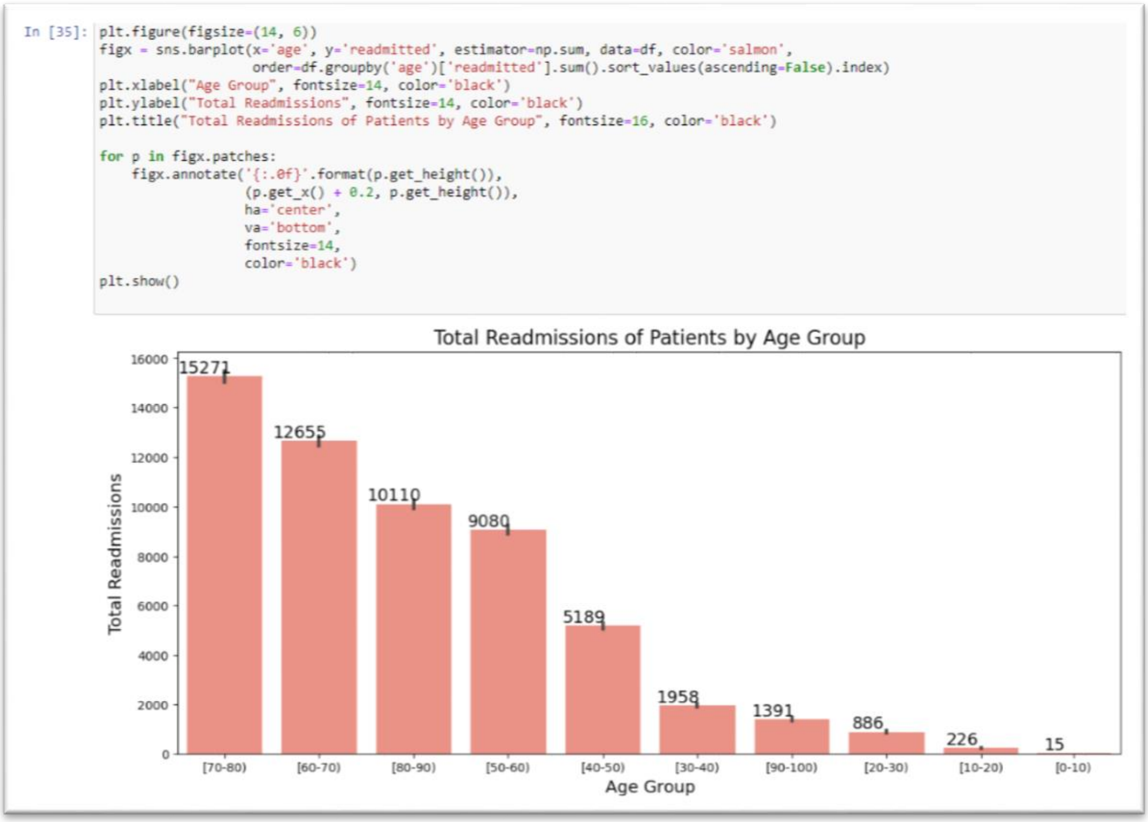
```
In [32]: plt.figure(figsize=(10, 6))
         sns.histplot(df['time_in_hospital'], bins=20, kde=True)
         plt.title('Distribution of Time in Hospital')
         plt.xlabel('Time in Hospital')
         plt.ylabel('Frequency')
         plt.show()
```



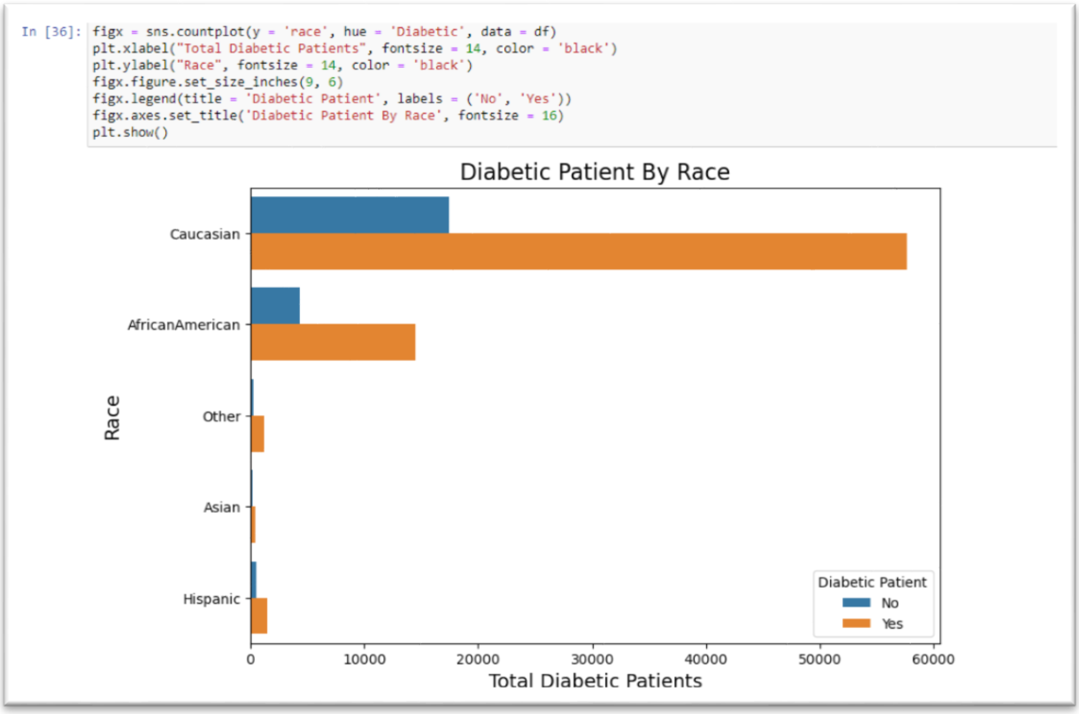
```
In [33]: plt.figure(figsize=(8, 5))
         sns.countplot(data=df, x='race')
         plt.title('Count of Patients by Race')
         plt.xlabel("Race", fontsize = 14, color = 'black')
         plt.ylabel("Count", fontsize = 14, color = 'black')
         plt.xticks(rotation=45)
         plt.show()
```



```
In [34]: plt.figure(figsize = (14,6))
         figx = sns.barplot(x = 'age', y = 'num_medications', estimator = np.sum, data = df)
         plt.xlabel("Age Group", fontsize = 14, color = 'black')
         plt.ylabel("Total Medications Consumed", fontsize = 14, color = 'black')
         plt.title("Total Medications Consumed By Age Group", fontsize = 16, color = 'black')

         for p in figx.patches:
             figx.annotate('{:.0f}'.format(p.get_height()),
                           (p.get_x() + 0.2, p.get_height()),
                           ha = 'center',
                           va = 'bottom',
                           fontsize = 14,
                           color = 'black')
         plt.show()
```

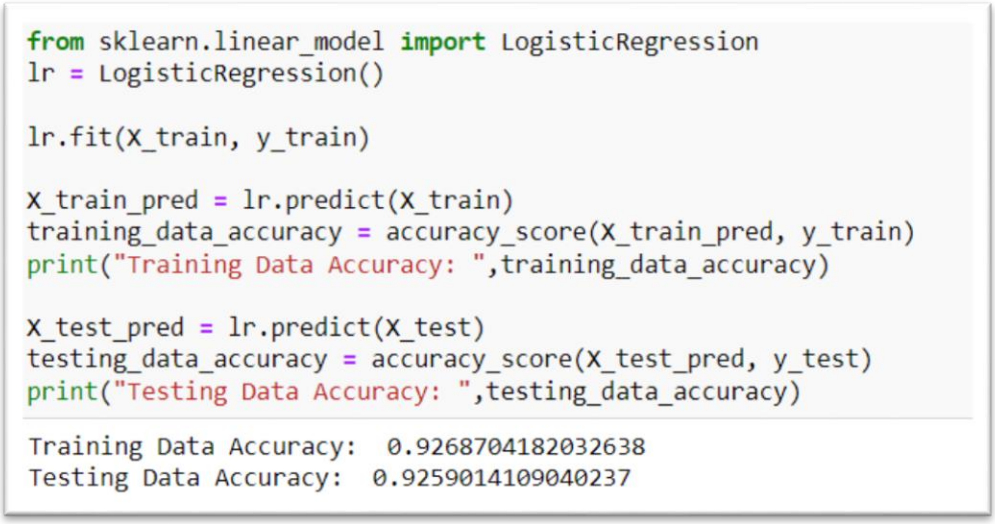


```
In [35]: plt.figure(figsize=(14, 6))
         figx = sns.barplot(x='age', y='readmitted', estimator=np.sum, data=df, color='salmon',
                            order=df.groupby('age')['readmitted'].sum().sort_values(ascending=False).index)
         plt.xlabel("Age Group", fontsize=14, color='black')
         plt.ylabel("Total Readmissions", fontsize=14, color='black')
         plt.title("Total Readmissions of Patients by Age Group", fontsize=16, color='black')

         for p in figx.patches:
             figx.annotate('{:.0f}'.format(p.get_height()),
                           (p.get_x() + 0.2, p.get_height()),
                           ha='center',
                           va='bottom',
                           fontsize=14,
                           color='black')
         plt.show()
```



```
In [36]: figx = sns.countplot(y = 'race', hue = 'Diabetic', data = df)
         plt.xlabel("Total Diabetic Patients", fontsize = 14, color = 'black')
         plt.ylabel("Race", fontsize = 14, color = 'black')
         figx.figure.set_size_inches(9, 6)
         figx.legend(title = 'Diabetic Patient', labels = ('No', 'Yes'))
         figx.axes.set_title('Diabetic Patient By Race', fontsize = 16)
         plt.show()
```

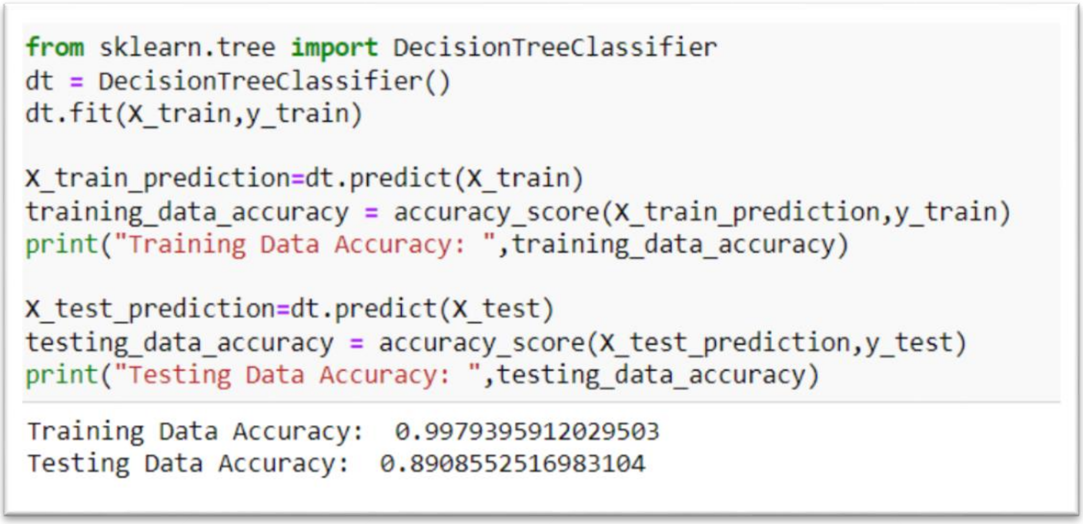
○ Logistic Regression:



```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()

lr.fit(X_train, y_train)

X_train_pred = lr.predict(X_train)
training_data_accuracy = accuracy_score(X_train_pred, y_train)
print("Training Data Accuracy: ",training_data_accuracy)

X_test_pred = lr.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_pred, y_test)
print("Testing Data Accuracy: ",testing_data_accuracy)
```

```
Training Data Accuracy:  0.9268704182032638
Testing Data Accuracy:  0.9259014109040237
```

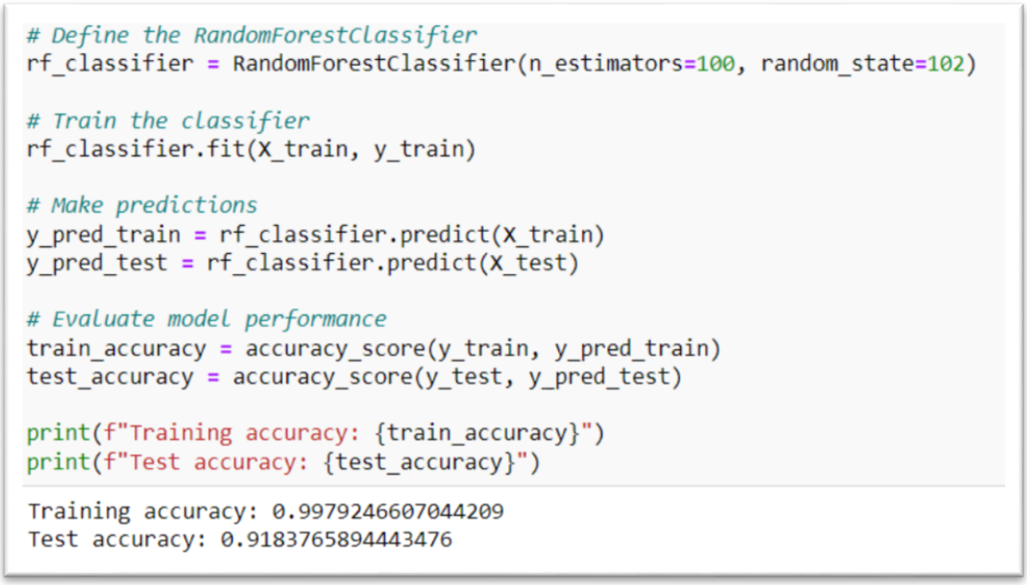
- Decision Tree Classifier:



```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier()
dt.fit(X_train,y_train)

X_train_prediction=dt.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction,y_train)
print("Training Data Accuracy: ",training_data_accuracy)

X_test_prediction=dt.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction,y_test)
print("Testing Data Accuracy: ",testing_data_accuracy)
```

```
Training Data Accuracy:  0.9979395912029503
Testing Data Accuracy:  0.8908552516983104
```

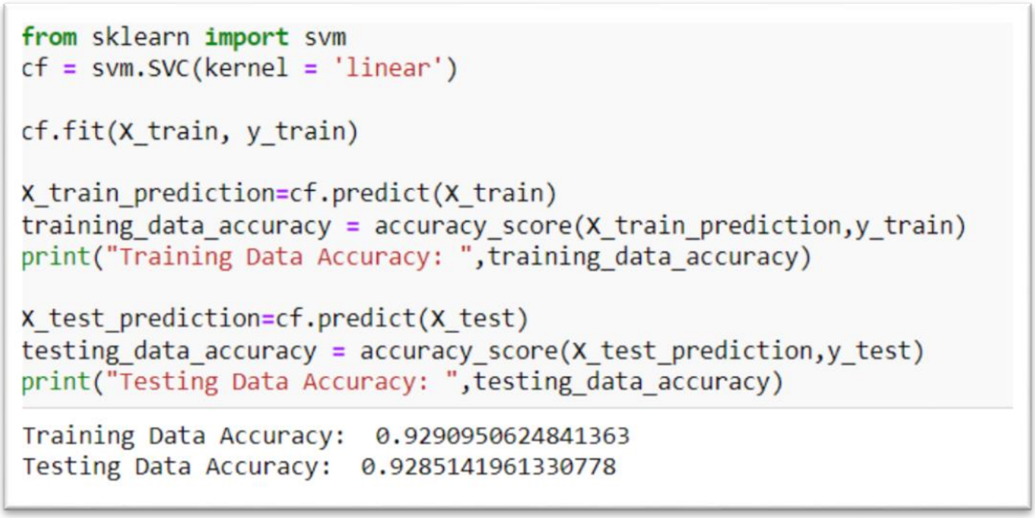
- Random Forest Classifier:



```
# Define the RandomForestClassifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=102)

# Train the classifier
rf_classifier.fit(X_train, y_train)

# Make predictions
y_pred_train = rf_classifier.predict(X_train)
y_pred_test = rf_classifier.predict(X_test)

# Evaluate model performance
train_accuracy = accuracy_score(y_train, y_pred_train)
test_accuracy = accuracy_score(y_test, y_pred_test)

print(f"Training accuracy: {train_accuracy}")
print(f"Test accuracy: {test_accuracy}")
```

```
Training accuracy: 0.9979246607044209
Test accuracy: 0.9183765894443476
```

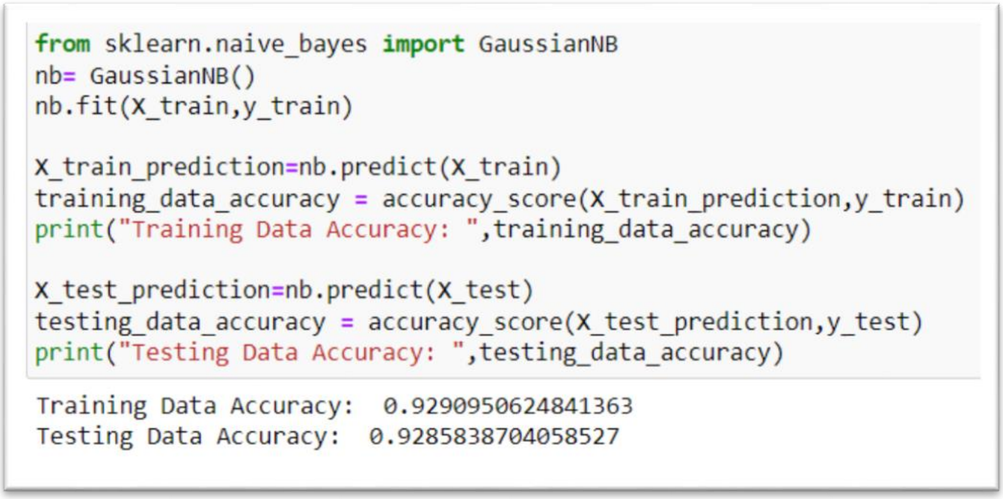
○ Support Vector Machine:



```
from sklearn import svm
cf = svm.SVC(kernel = 'linear')

cf.fit(X_train, y_train)

X_train_prediction=cf.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction,y_train)
print("Training Data Accuracy: ",training_data_accuracy)

X_test_prediction=cf.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction,y_test)
print("Testing Data Accuracy: ",testing_data_accuracy)
```

```
Training Data Accuracy:  0.9290950624841363
Testing Data Accuracy:  0.9285141961330778
```

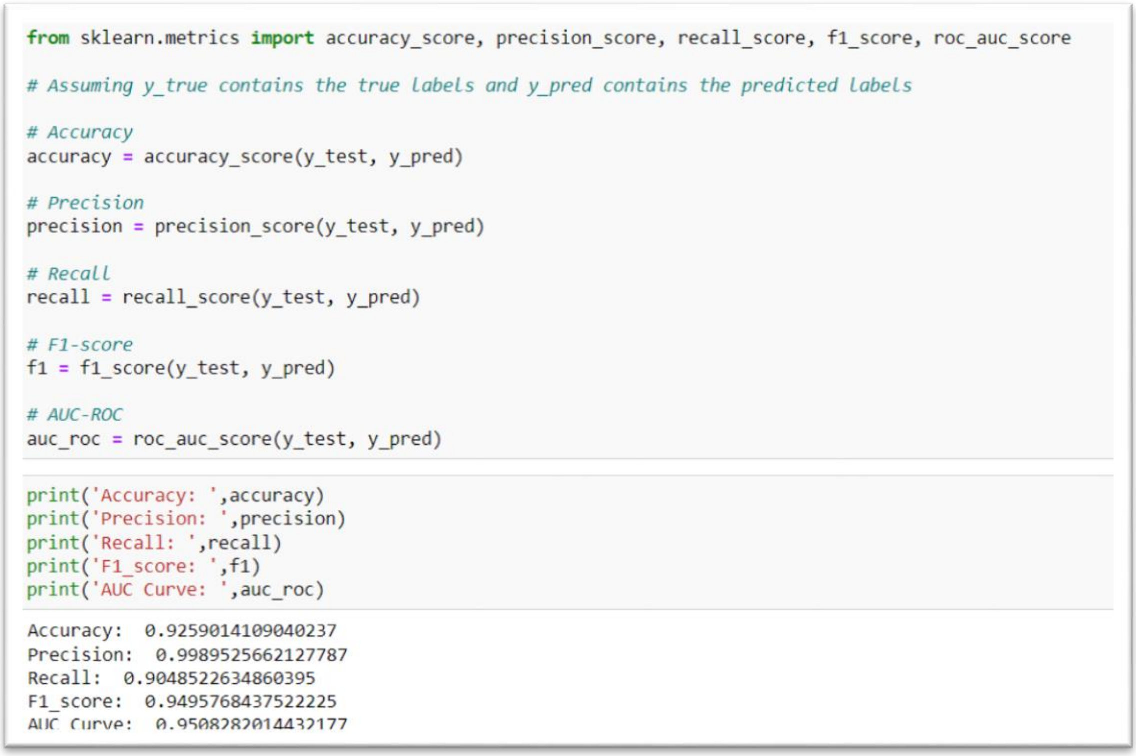
○ Naïve Bayes:



```
from sklearn.naive_bayes import GaussianNB
nb= GaussianNB()
nb.fit(X_train,y_train)

X_train_prediction=nb.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction,y_train)
print("Training Data Accuracy: ",training_data_accuracy)

X_test_prediction=nb.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction,y_test)
print("Testing Data Accuracy: ",testing_data_accuracy)
```

```
Training Data Accuracy:  0.9290950624841363
Testing Data Accuracy:  0.9285838704058527
```

○ Accuracy Scores:

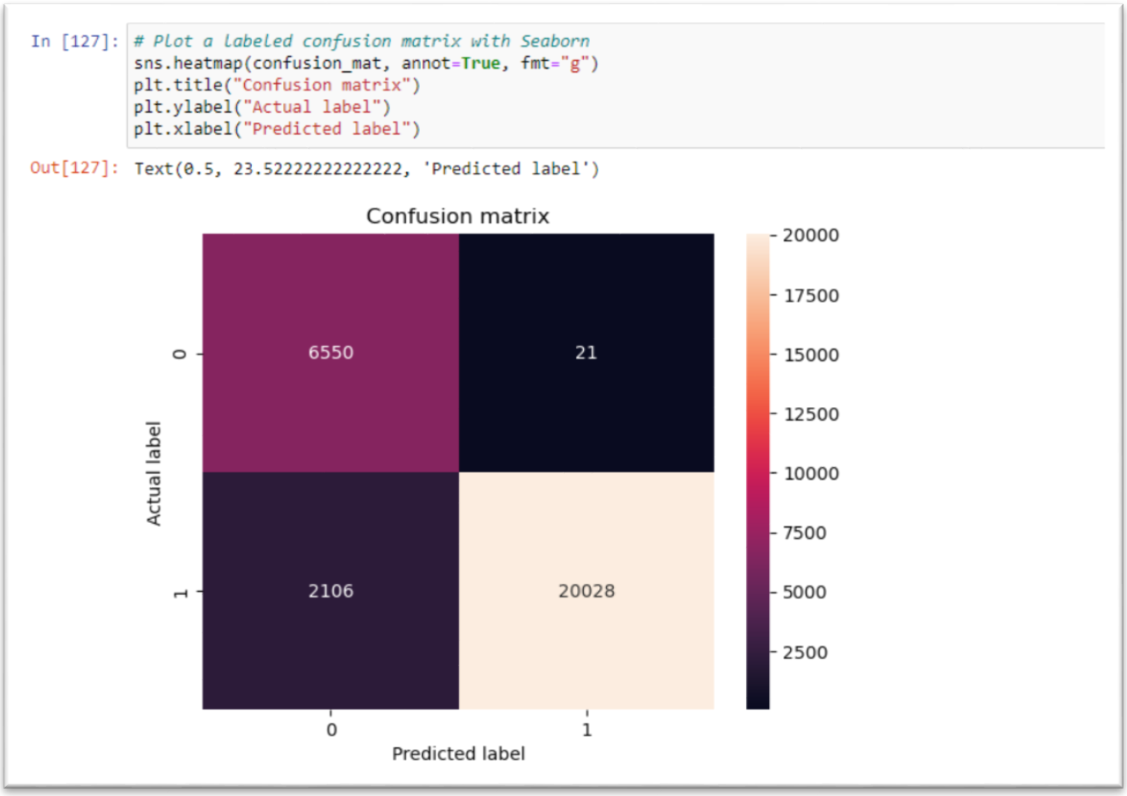


```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score

# Assuming y_true contains the true labels and y_pred contains the predicted labels

# Accuracy
accuracy = accuracy_score(y_test, y_pred)

# Precision
precision = precision_score(y_test, y_pred)

# Recall
recall = recall_score(y_test, y_pred)

# F1-score
f1 = f1_score(y_test, y_pred)

# AUC-ROC
auc_roc = roc_auc_score(y_test, y_pred)
```

```
print('Accuracy: ',accuracy)
print('Precision: ',precision)
print('Recall: ',recall)
print('F1_score: ',f1)
print('AUC Curve: ',auc_roc)
```

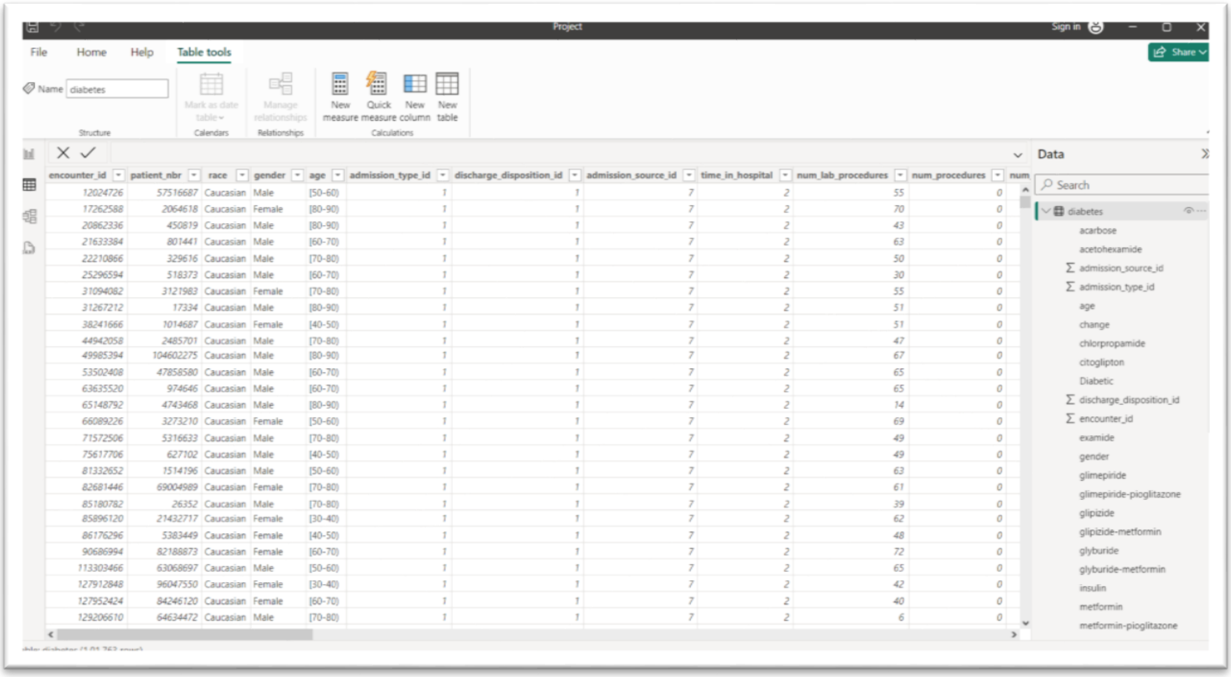
```
Accuracy:  0.9259014109040237
Precision:  0.9989525662127787
Recall:  0.9048522634860395
F1_score:  0.9495768437522225
AUC Curve:  0.9508282014432177
```

○ Confusion Matrix:



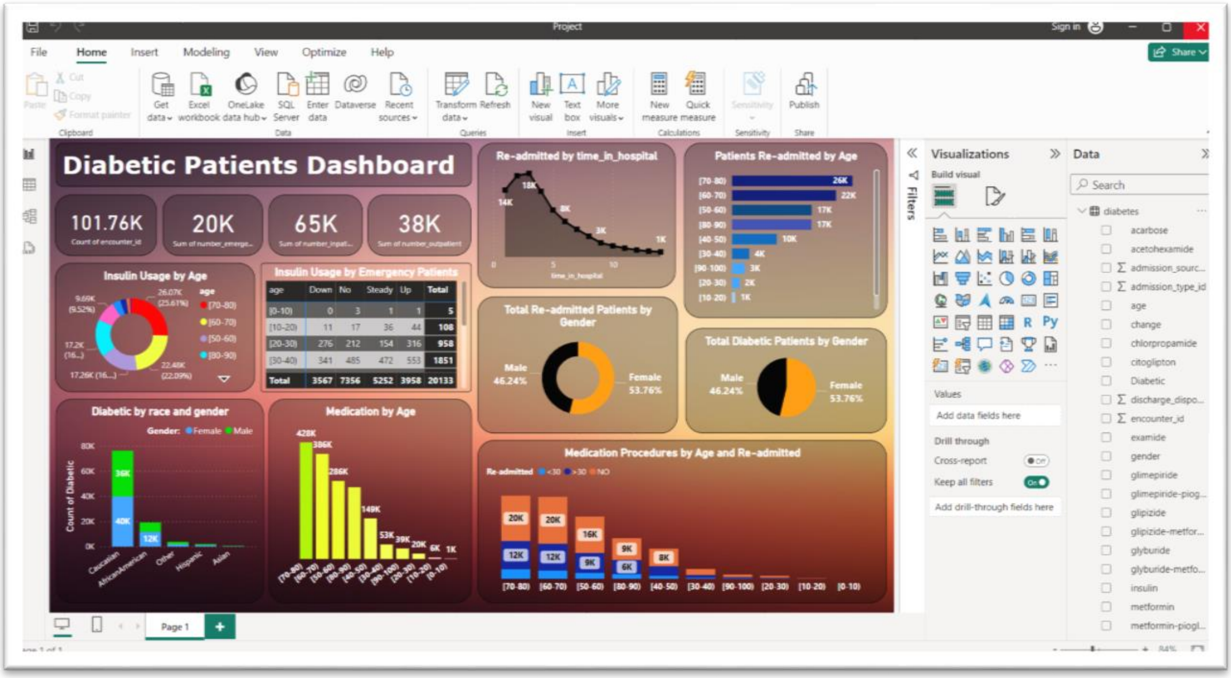
```
In [127]: # Plot a labeled confusion matrix with Seaborn
          sns.heatmap(confusion_mat, annot=True, fmt="g")
          plt.title("Confusion matrix")
          plt.ylabel("Actual label")
          plt.xlabel("Predicted label")

Out[127]: Text(0.5, 23.52222222222222, 'Predicted label')
```

- Power BI Dashboard:



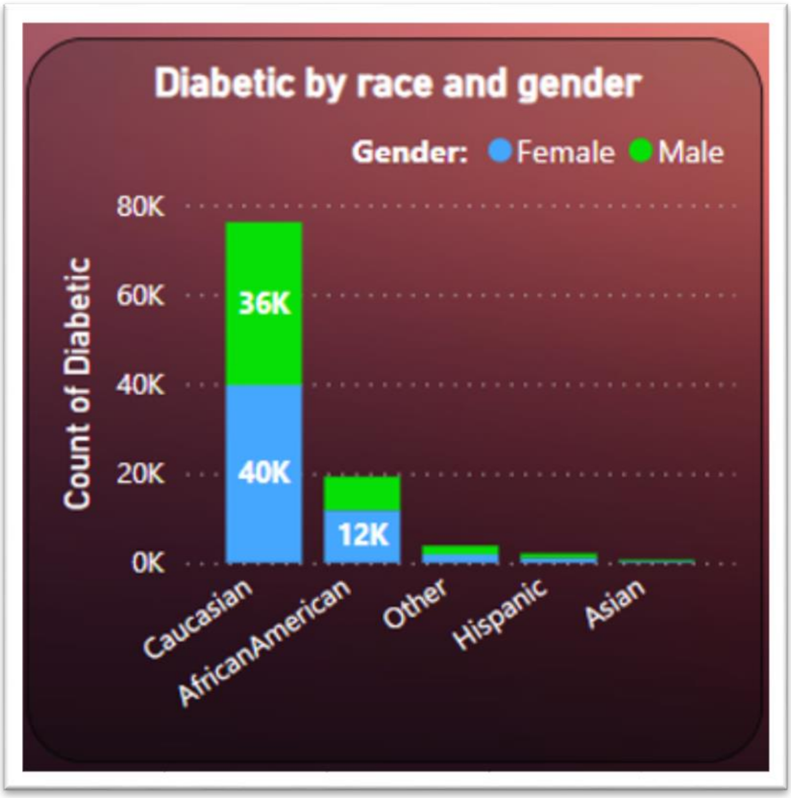
encounter_id	patient_nbr	race	gender	age	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	num_lab_procedures	num_procedures	num
12024726	57516687	Caucasian	Male	[50-60)	1	1	7	2	55	0	
17262588	2064618	Caucasian	Female	[80-90)	1	1	7	2	70	0	
20862336	450819	Caucasian	Male	[80-90)	1	1	7	2	43	0	
21633384	801441	Caucasian	Male	[60-70)	1	1	7	2	63	0	
22210866	329616	Caucasian	Male	[70-80)	1	1	7	2	50	0	
25296594	518373	Caucasian	Male	[60-70)	1	1	7	2	30	0	
31094082	3121983	Caucasian	Female	[70-80)	1	1	7	2	55	0	
31267212	17334	Caucasian	Male	[80-90)	1	1	7	2	51	0	
38241666	1014687	Caucasian	Female	[40-50)	1	1	7	2	51	0	
44942058	2485701	Caucasian	Male	[70-80)	1	1	7	2	47	0	
49985394	104602275	Caucasian	Male	[80-90)	1	1	7	2	67	0	
53502408	47858580	Caucasian	Male	[60-70)	1	1	7	2	65	0	
63635520	974646	Caucasian	Male	[60-70)	1	1	7	2	65	0	
65148792	4743468	Caucasian	Male	[80-90)	1	1	7	2	14	0	
66089226	3273210	Caucasian	Female	[50-60)	1	1	7	2	69	0	
71572506	5316633	Caucasian	Male	[70-80)	1	1	7	2	49	0	
75617706	627102	Caucasian	Male	[40-50)	1	1	7	2	49	0	
81332652	1514196	Caucasian	Male	[50-60)	1	1	7	2	63	0	
82681446	69004989	Caucasian	Female	[70-80)	1	1	7	2	61	0	
85180782	26352	Caucasian	Male	[70-80)	1	1	7	2	39	0	
85896120	21432717	Caucasian	Female	[30-40)	1	1	7	2	62	0	
86176296	5383449	Caucasian	Female	[40-50)	1	1	7	2	48	0	
90686994	82188873	Caucasian	Female	[60-70)	1	1	7	2	72	0	
113303466	63068697	Caucasian	Male	[50-60)	1	1	7	2	65	0	
127912848	96047550	Caucasian	Female	[30-40)	1	1	7	2	42	0	
127952424	84246120	Caucasian	Female	[60-70)	1	1	7	2	40	0	
129206610	64634472	Caucasian	Male	[70-80)	1	1	7	2	6	0	

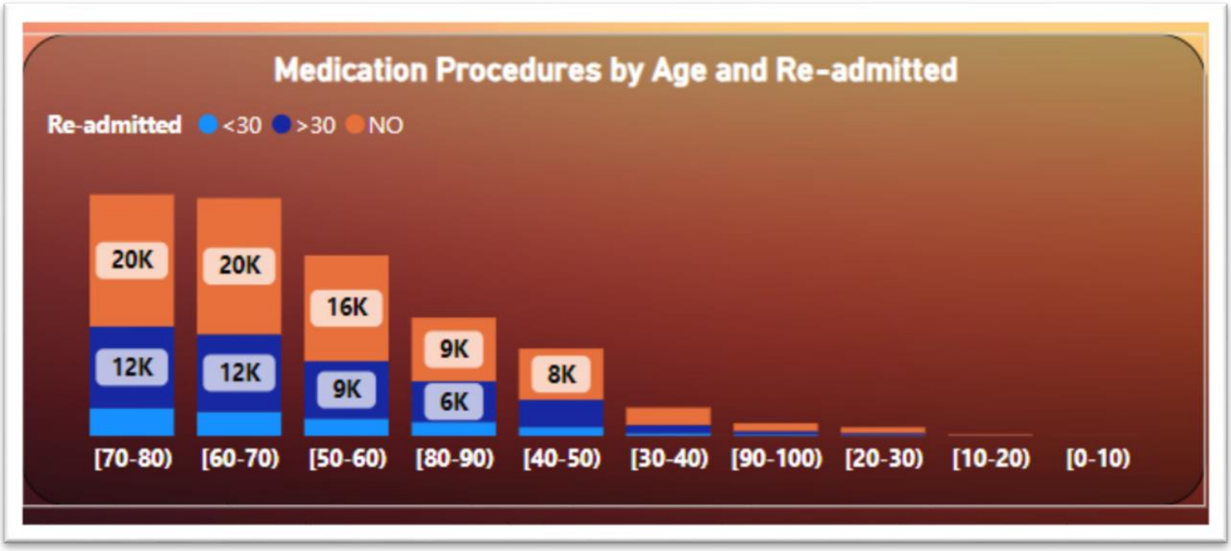
Table: diabetes (1,01,763 rows)

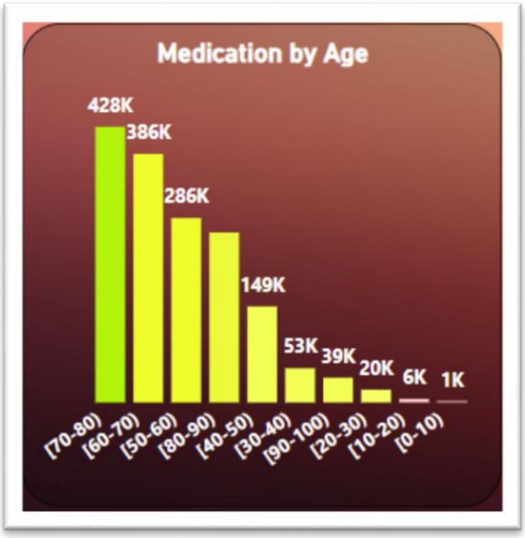


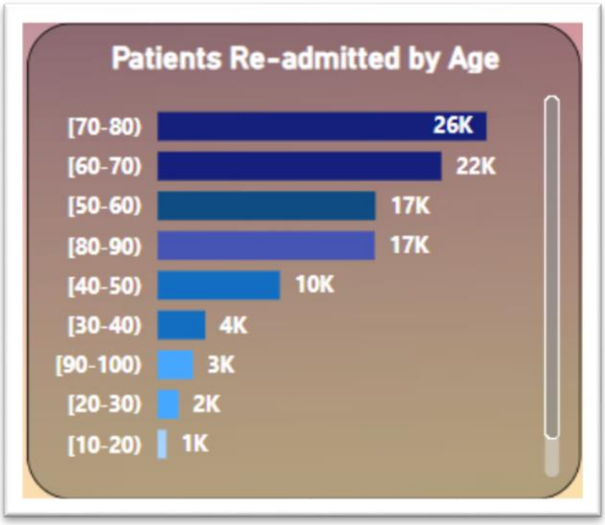
Insulin Usage by Emergency Patients

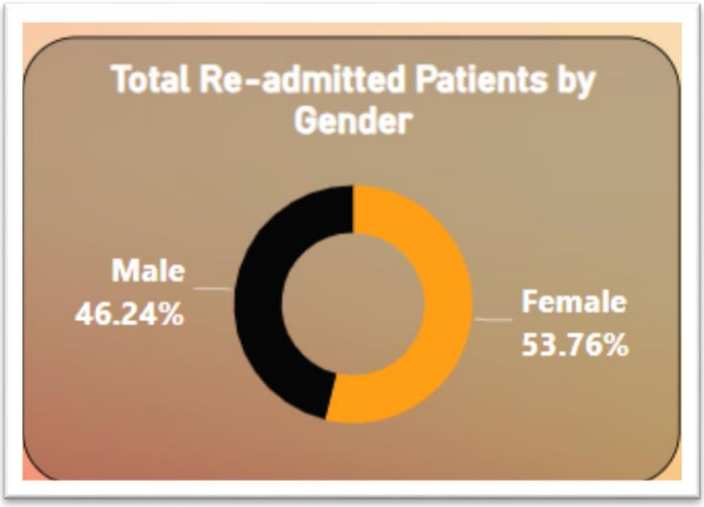
age	Down	No	Steady	Up	Total
[0-10)	0	3	1	1	5
[10-20)	11	17	36	44	108
[20-30)	276	212	154	316	958
[30-40)	341	485	472	553	1851
Total	3567	7356	5252	3958	20133

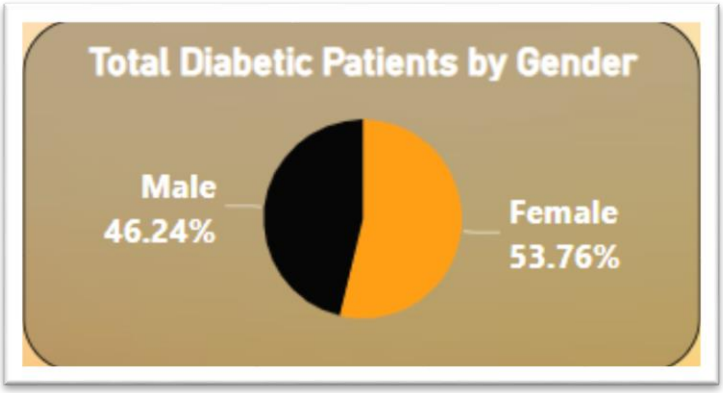

Diabetic by race and gender
Gender: ● Female ● Male
Count of Diabetic
80K
60K
40K
20K
0K
36K
40K
12K
Caucasian
AfricanAmerican
Other
Hispanic
Asian

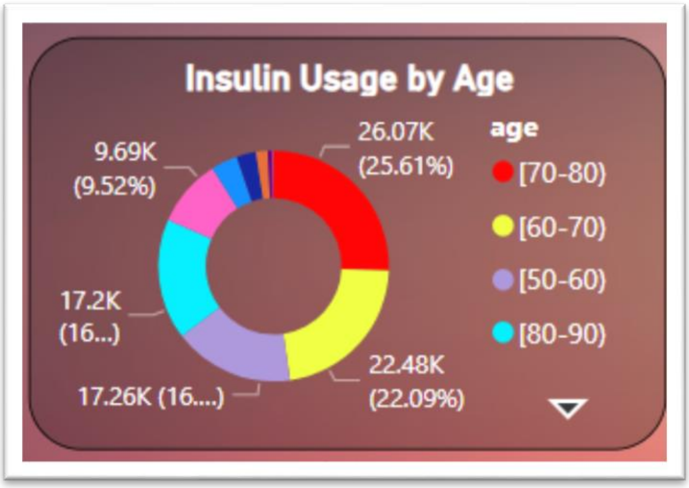

Medication Procedures by Age and Re-admitted
Re-admitted ● <30 ● >30 ● NO
20K
12K
20K
12K
16K
9K
9K
6K
8K
[70-80)
[60-70)
[50-60)
[80-90)
[40-50)
[30-40)
[90-100)
[20-30)
[10-20)
[0-10)

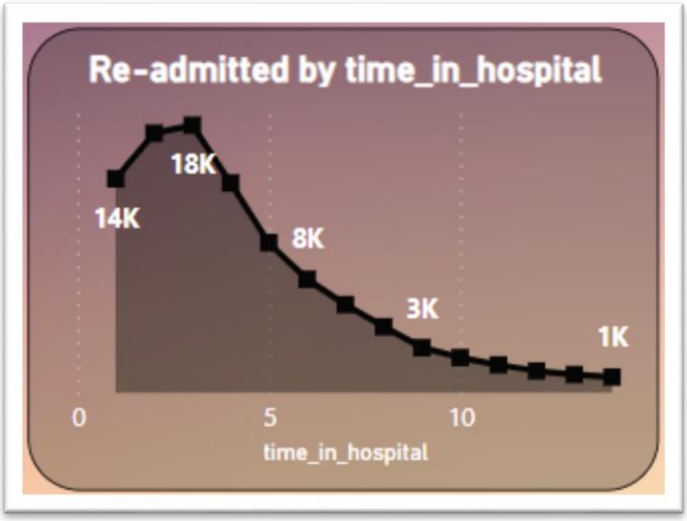

Medication by Age
428K
386K
286K
149K
53K
39K
20K
6K
1K
[70-80)
[60-70)
[50-60)
[80-90)
[40-50)
[30-40)
[90-100)
[20-30)
[10-20)
[0-10)

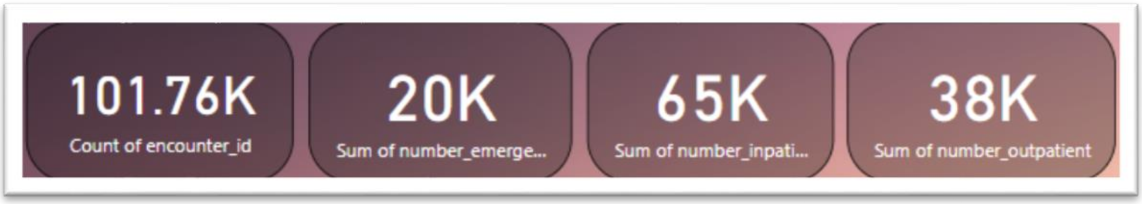

Patients Re-admitted by Age
[70-80) 26K
[60-70) 22K
[50-60) 17K
[80-90) 17K
[40-50) 10K
[30-40) 4K
[90-100) 3K
[20-30) 2K
[10-20) 1K


Total Re-admitted Patients by Gender
Male
46.24%
Female
53.76%


Total Diabetic Patients by Gender
Male
46.24%
Female
53.76%


Insulin Usage by Age
26.07K
(25.61%)
9.69K
(9.52%)
17.2K
(16...)
17.26K (16....)
22.48K
(22.09%)
age
● [70-80)
● [60-70)
● [50-60)
● [80-90)


Re-admitted by time_in_hospital
14K
18K
8K
3K
1K
0
5
10
time_in_hospital



101.76K

Count of encounter_id

20K

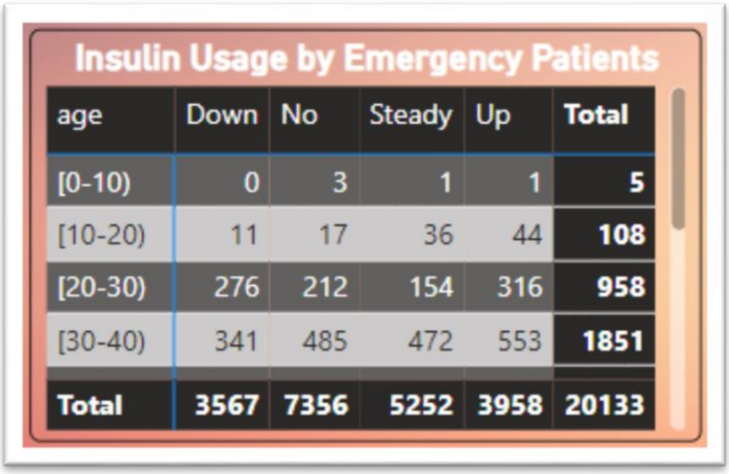
Sum of number_emerge...

65K

Sum of number_inpati...

38K

Sum of number_outpatient



Insulin Usage by Emergency Patients

age	Down	No	Steady	Up	Total
[0-10)	0	3	1	1	**5**
[10-20)	11	17	36	44	**108**
[20-30)	276	212	154	316	**958**
[30-40)	341	485	472	553	**1851**
Total	**3567**	**7356**	**5252**	**3958**	**20133**

CONCLUSION

To sum up, I thought the degree programs, workshops, and practical training in Data Science were really beneficial. I now have a thorough understanding of many statistical tools and techniques used in data analytics, along with hands-on practice in using these techniques on real-world datasets.

Particularly helpful was the training, which gave me the opportunity to deal with actual datasets and acquire knowledge of every step of the data analysis process, from pre-processing and data cleaning to modeling and visualization. The certificate programs and workshops, which provide a comprehensive review of the fundamental ideas and concepts in data analytics, were also quite educational.

All things considered, I believe that these experiences have given me the abilities and information I need to pursue a career in data science. I have faith in my capacity to apply critical thinking to complicated datasets and to employ statistical techniques and instruments to get insightful conclusions and make informed judgments.

BIBLIOGRAPHY

UCI Machine Learning Repository. (1999-2008). Diabetes 130-US hospitals for years 1999-2008 Data Set. Retrieved from:

https://archive.ics.uci.edu/dataset/296/diabetes+130-us+hospitals+for+years+1999-2008