Note

Part B of this assignment can be done in a group of two students or individually. Both students need to submit the assignment for both parts and provide both names, email, and student IDs at the top of the assignment.

Submit your complete project, including your Python Notebook with markdown explanations, and a comprehensive PDF document. The PDF should contain clear screenshots of input/output commands with results, images of your deployed Azure portal resources, detailed step-by-step explanations for each process, and final output screenshots. Additionally, include a section in the PDF answering the provided questions (to be specified separately).

Contact your TA for any questions related to this assignment, or post clarification questions to the Piazza platform.

Part B

Data Input: Claim a dataset from Piazza - link. If the dataset is too large, you can take a subset of the data as well. No two groups can have the same dataset.
Your selected dataset should meet the following criteria:

1. **It must contain a minimum of 1,000 instances (rows or data points).**
2. **It should include at least six features (columns or attributes).**

Using this dataset, you are required to address a substantial and meaningful problem. Your analysis should demonstrate:

1. **A clear understanding of the dataset's context and potential applications.**
2. **The ability to formulate relevant questions or hypotheses based on the data.**
3. **Appropriate use of data analysis techniques to extract insights.**
4. **The capacity to draw meaningful conclusions that could inform decision-making or further research.**

Some problems to consider:

1. **Fraud Detection System**
2. **Customer Churn Rate Prediction**
3. **Segmentation using Clustering**
4. **Recommendations with your Dataset**
5. **Sales Forecasting**
6. **Stock Price Predictions**
7. **Human Activity Recognition with Smartphones**
8. **Wine Quality Predictions**
9. **Breast Cancer Prediction**
10. **Sorting of Specific Tweets on Twitter etc.**

Implement this part in Azure Machine learning using Azure Notebook

Dataset:

National Health and Nutrition Health Survey 2013-2014 (NHANES) Age Prediction Subset

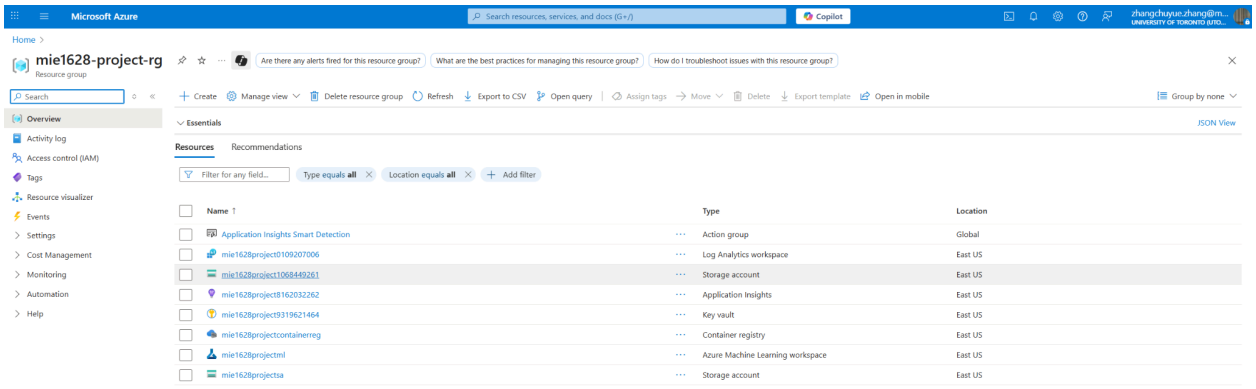
Link:
https://archive.ics.uci.edu/dataset/887/national+health+and+nutrition+health+survey+2013-2014+(nhanes)+age+prediction+subset

#Instance: 6287
#feature: 7

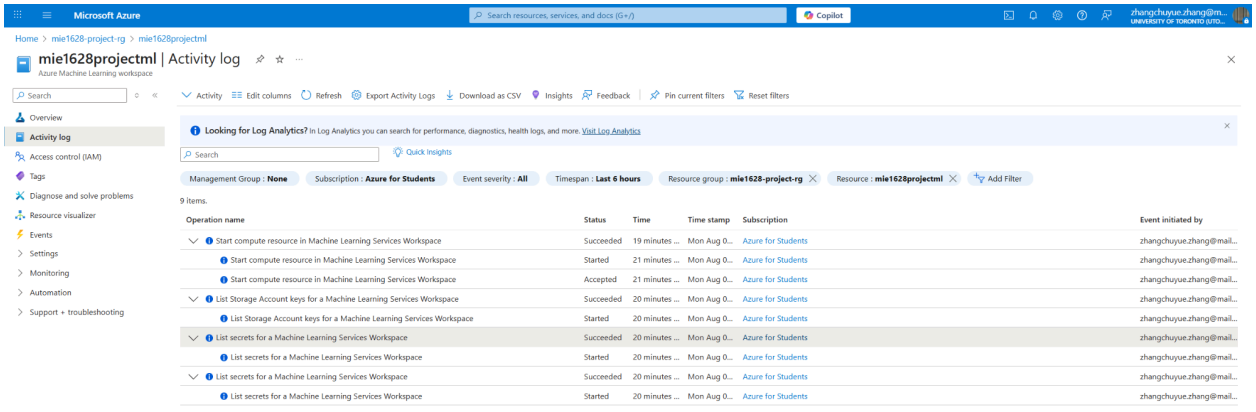
Microsoft Azure Setup:

The setup process is as shown below:

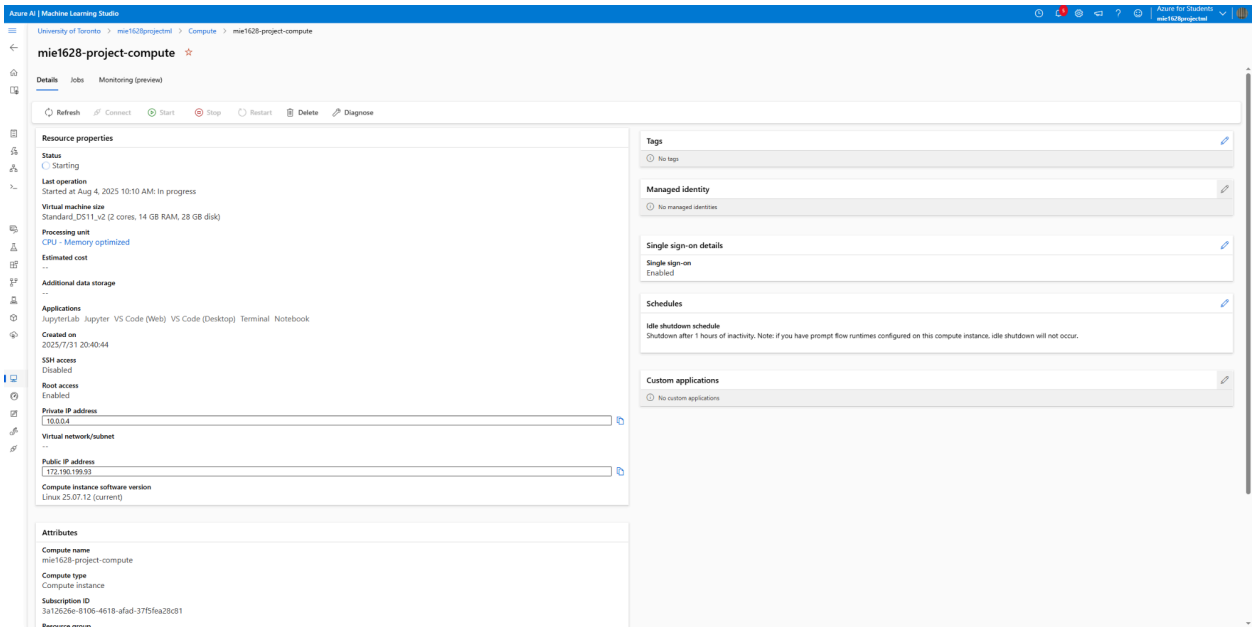
1. Resources Group Setup



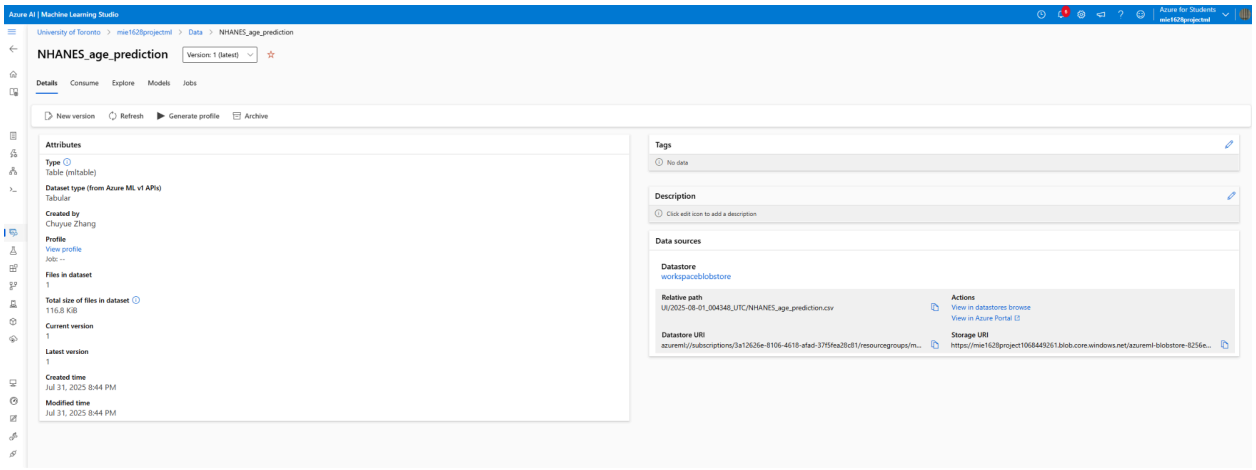
2. ML Resources Setup



3. ML Compute Setup



4. Input Data - Project Dataset



Answers:

1. [Marks: 15] Clearly define the problem you intend to address using this dataset. Present a comprehensive problem statement that includes:
a. A detailed description of the meaningful issue you're tackling
b. An outline of all necessary steps, including:
- **i. Data preprocessing**
- **ii. Data cleaning**
- **iii. Modeling approach**

Your problem statement should be thorough, spanning approximately half to one full page. If you determine that data cleaning is unnecessary, please provide a justification for why this dataset doesn't require cleaning. In such a case, allocate more attention to other crucial aspects such as EDA and the modeling process.
Ensure your problem statement is well-structured, coherent, and provides a clear roadmap for your data analysis project.

Problem Statement:

Nowadays, the increasing rate of chronic diseases such as diabetes and cardiovascular disease (CVD) has posed a significant challenge to public health. Therefore, it is crucial to identify the risk of these diseases at an early stage to facilitate preventive interventions and enhance health outcomes. In the traditional medical domain, those health risk assessments rely on the judgment of experts and predefined medical guidelines. However, with the increase in population and the surge in demand for medical resources, data-driven approaches are emerging as an alternative for personalized health care and risk identification, enabled by the growing availability of large-scale, population-based health datasets.

The project would use a pre-selected subset of the dataset from the National Health and Nutrition Examination Survey (NHANES), administered by the Centers for Disease Control and Prevention (CDC). It collects extensive health and nutritional information from a diverse U.S. population, intending to assess the health and nutritional status of the United States' population. From the statistics of the dataset, data analysis and machine learning techniques can be implemented to extract the underlying patterns and present a predictive insight into the classification of diabetes.

Main Objective:

The primary goal of this project is to develop a machine learning model that can accurately predict the likelihood of diabetes based on physiological measurements, lifestyle choices, and biochemical markers from the subset of the NHANES 2013–2014 dataset. By data-driven analysis of the features like respondents' age, BMI, or glucose levels, the project aims to identify the latent patterns and signal the potential risk of diabetes in order to support preventative care and risk management. Different predictive machine learning classification models would be deployed on Microsoft Notebook using Azure Machine Learning for comparison to the effectiveness of the results

Methodology Outline:

i. Data preprocessing

- **Feature Engineering**
  - A new feature, Glu_Insulin_Ratio, was created to capture the relationship between glucose and insulin levels.
  - BMI was categorized into four groups: Underweight, Normal, Overweight, and Obese.
- **Missing Value Handling**
  - Numerical features with missing values were imputed using the mean.
  - Categorical features were imputed using the mode.
- **Categorical Encoding**
  - Categorical variables (Gender, BMI_Category, Age_Group, Age_Bin) were one-hot encoded, with the first category dropped to avoid redundancy and ensure the encoded features remain interpretable and suitable for machine learning models.
- **Feature Scaling**
  - All numerical features were standardized using **Z-score normalization** to ensure zero mean and unit variance, which is essential for distance-based models like SVM.

ii. Data cleaning

Contrary to the previous assumption that no cleaning was required, data cleaning was performed as follows:

- Missing values were identified and handled systematically (mean/mode imputation).
- No duplicate rows were observed in the dataset.
- No outliers were explicitly removed, but feature engineering and scaling minimized their potential impact.

iii. Modeling approach

- **Target Variable:** Diabetes_Status
- **Train-Test Split:** An 80/20 split was used to separate training and testing data, with a fixed random_state=42 to ensure reproducibility.
- **Feature Set:** All engineered and encoded features excluding the target.
- **Model Training and Evaluation:** The notebook supports flexible model experimentation. Models include: Decision Tree, Support Vector Machine
- **Evaluation Metrics:** Model performance is assessed using metrics such as: Accuracy, Confusion Matrix, RMSE

2. [Marks: 10] Explore your dataset and provide at least 5 meaningful charts/graphs with an explanation.

Graph 1. Heatmap

```python
# Plot heatmap
numeric_features = df.select_dtypes(include='number')
corr_matrix = numeric_features.corr()


plt.figure(figsize=(10, 8))
sns.heatmap(
    corr_matrix,
    annot=False,
    cmap='coolwarm',
    center=0,
    square=True,
    linewidths=0.5,
    cbar_kws={"shrink": 0.8}
)
plt.title('Correlation Heatmap of Numeric Features')
plt.xticks(rotation=45, ha='right')
plt.yticks(rotation=0)
plt.tight_layout()
```

✓ <1 sec

```
Transmission succeeded: Item received: 26. Items accepted: 26
```

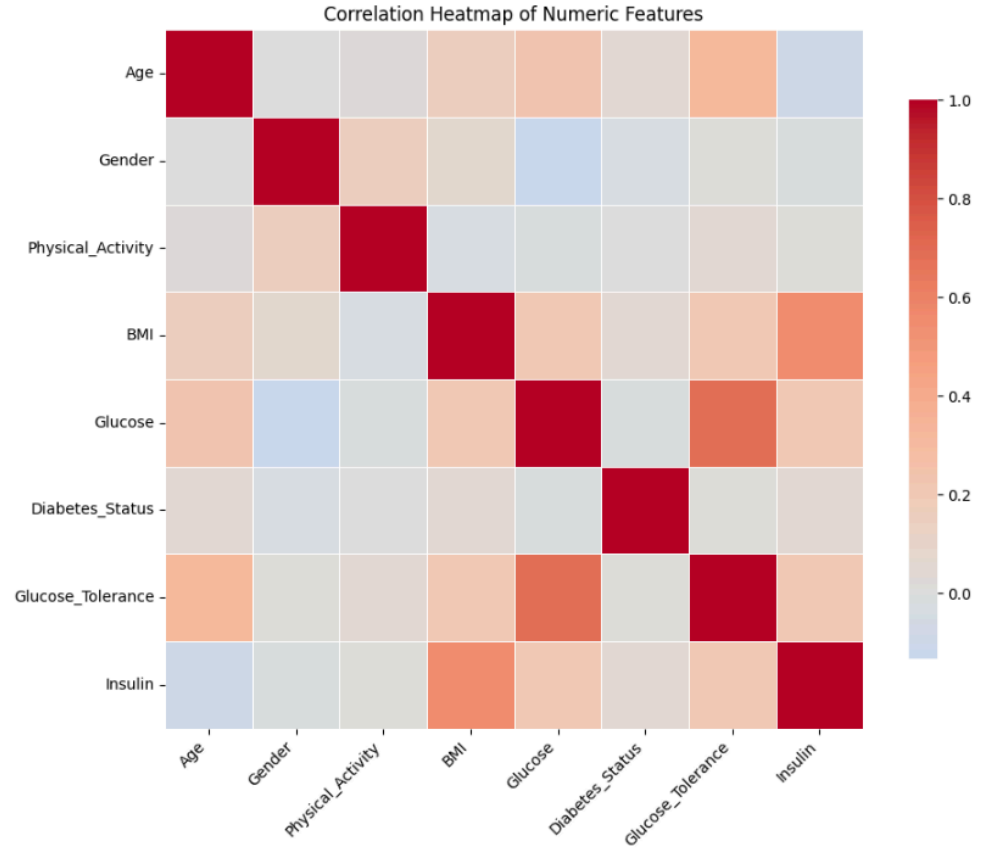


Figure 1. Heatmap

This heatmap visualises the pairwise Pearson correlations among the nine continuous variables.

- Dark red blocks show strong positive relationships. For example, Glucose and Insulin ($r \approx 0.65$), and BMI and Insulin ($r \approx 0.55$) are closely related, which suggests a link between body weight, insulin response, and high blood sugar.
- Light blue and gray areas are common, showing that many variables do not have strong relationships and change independently from each other.

Overall, the chart helps identify redundant predictors and reveal metabolically relevant clusters of features for downstream modeling improvement.

Graph 2. BMI vs Insulin

```python
# Scatter plot
plt.figure(figsize=(10, 6))
scatter = plt.scatter(
    df['BMI'],
    df['Insulin'],
    c=df['Glucose'],
    cmap='coolwarm',
    alpha=0.7
)

plt.colorbar(scatter, label='Glucose (mg/dL)')
plt.title('BMI vs Insulin with Glucose Level Coloring')
plt.xlabel('BMI (BMI)')
plt.ylabel('Insulin (Insulin)')
plt.tight_layout()
plt.show()
```

✓ <1 sec

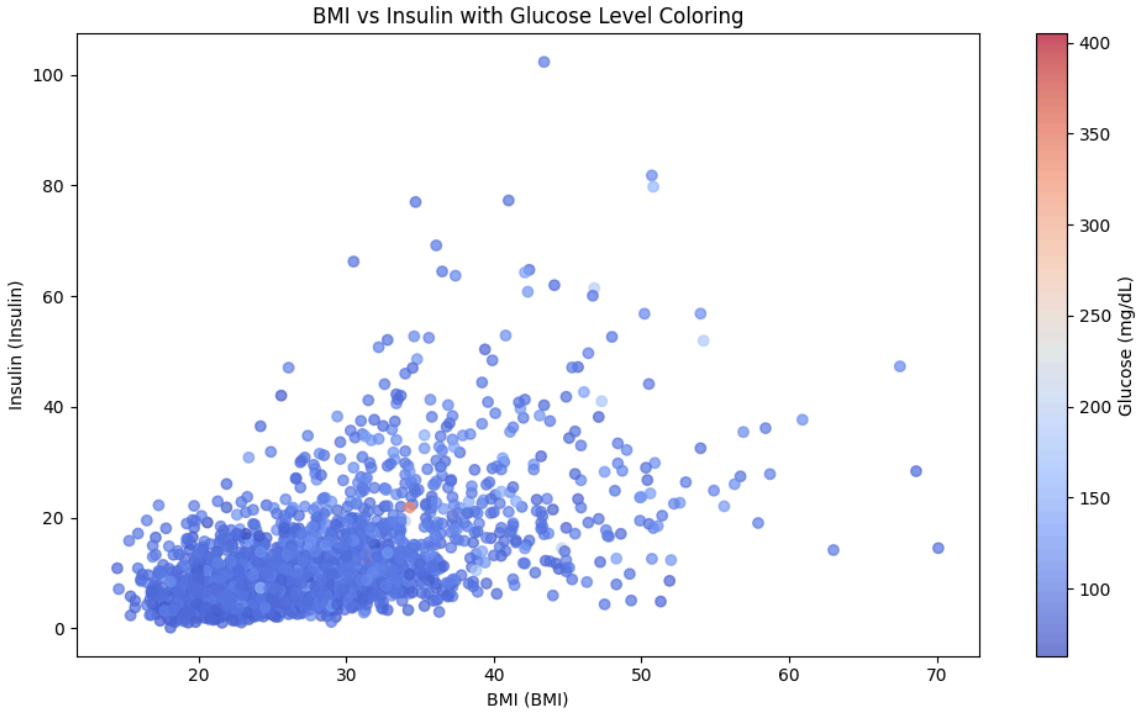


Figure 2. BMI vs Insulin

Each point represents an individual; the x-axis is BMI (kg/m²) and the y-axis is fasting Insulin (µU/mL). Colour intensity encodes Glucose concentration.

- The upward trend shows that insulin levels usually go up as BMI increases, which reflects how the body reacts to extra weight.
- Darker points mostly appear in the top right corner. This means people with both high BMI and high insulin often also have high glucose, showing signs of multiple metabolic problems.
- When BMI is above about 30, the points spread out more, meaning that insulin levels vary a lot among people in the obese group.

This multivariate view demonstrates how three key features interact simultaneously.

Graph 3. Glucose KDE by Diabetes_Status

```
1   plt.figure(figsize=(8, 6))
2   sns.kdeplot(data=df, x='Glucose', hue='Diabetes_Status', common_norm=False)
3   plt.title('Glucose Level by Diabetes Status')
4   plt.xlabel('Glucose (mg/dL)')
5   plt.ylabel('Density')
6   plt.tight_layout()
7   plt.savefig('glucose_by_diabetes_status.png')
```

✓ <1 sec

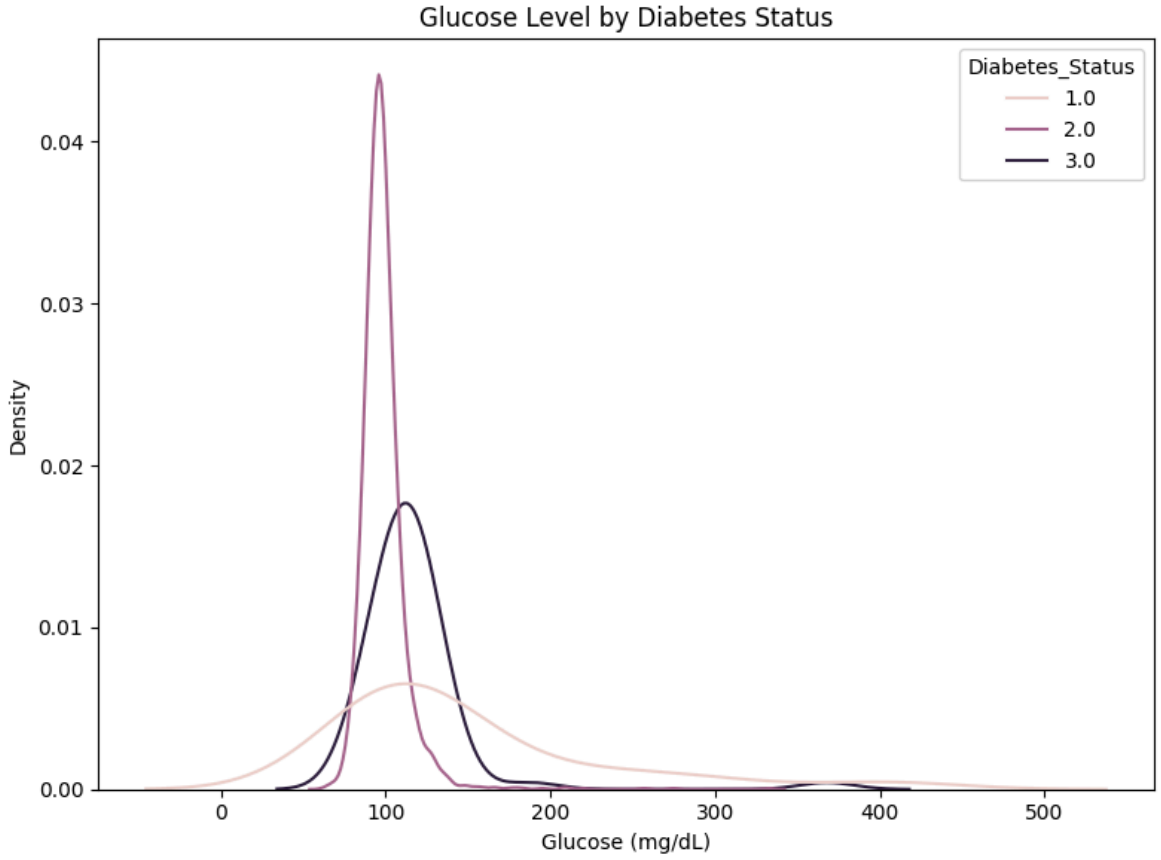


Figure 3. Glucose KDE by Diabetes Status

Density curves compare fasting glucose for three diagnostic categories (1 = No diabetes, 2 = Borderline/Prediabetes, 3 = Diagnosed diabetes).

- People without diabetes have a sharp peak around 95 mg/dL, and the curve drops quickly after.
- The prediabetes curve moves to the right (around 110–125 mg/dL) and becomes wider, showing that glucose levels are rising and more varied during this stage.
- The diabetes curve is the farthest to the right, with a long tail beyond 150 mg/dL, showing that many people in this group have very high glucose levels.

These distinct distributions support the clinical cut-offs and highlight increasing variability in glucose levels with disease advancement.

Graph 4. Glucose vs Age

```
1  plt.figure(figsize=(8, 6))
2  sns.scatterplot(x='Age', y='Glucose', hue='Diabetes_Status', data=df)
3  plt.title('Glucose Level by Age and Diabetes Status')
4  plt.xlabel('Age (Years)')
5  plt.ylabel('Glucose (mg/dL)')
6  plt.tight_layout()
```

✓ <1 sec

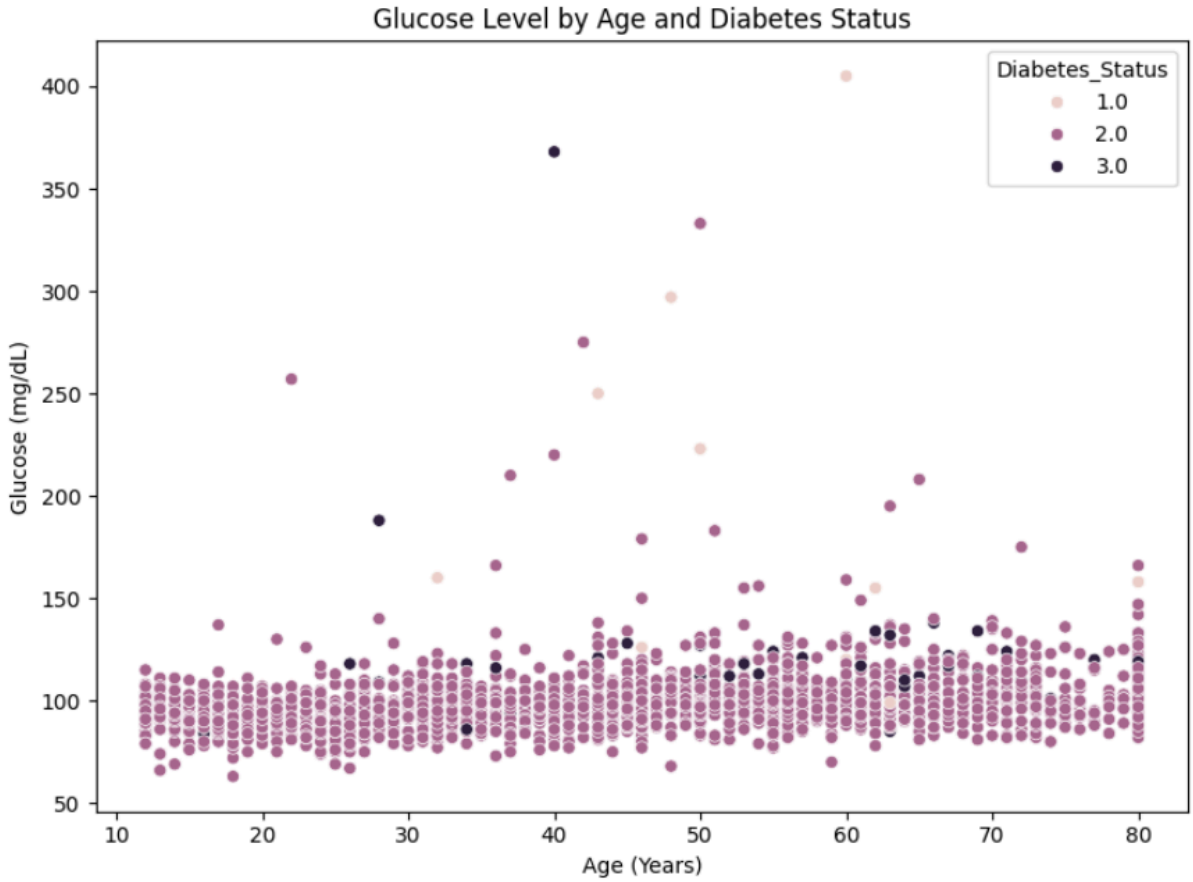


Figure 4. Glucose Level by Age and Diabetes Status

This chart relates age to fasting glucose, coloured by diabetes status.

- In general, glucose levels slowly go up with age. The average level rises from the 20s to older age, which may mean that the body becomes less able to control blood sugar over time.
- Colour clusters:
    - Light-colored dots (no diabetes) are mostly found in younger people.
    - Medium-colored dots (borderline or prediabetes) appear more in middle age.
    - Dark dots (diagnosed diabetes) are mostly seen after around age 45.
- Some dark dots also appear among younger people. These are unusual cases with very high glucose levels, which may need early care and attention.

This chart helps us see how diabetes risk changes with age and shows how health data varies across different life stages.

Graph 5. BMI by Gender

```
plt.figure(figsize=(8, 6))
sns.boxplot(x='Gender', y='BMI', data=df)
plt.title('BMI by Gender')
plt.xlabel('Gender (1=Male, 2=Female)')
plt.ylabel('BMI')
plt.tight_layout()
plt.savefig('bmi_by_gender.png')
```

✓ <1 sec

```
Using categorical units to plot a list of strings that are all parsable as floats or dates. If these strings should be plotted as numbers, cast to the appropriate data type before plotting.
Using categorical units to plot a list of strings that are all parsable as floats or dates. If these strings should be plotted as numbers, cast to the appropriate data type before plotting.
```

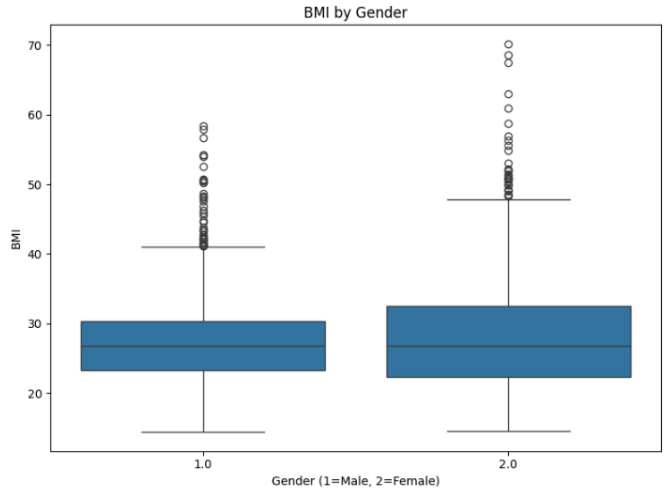


Figure 5. BMI by Gender

Box plots summarise BMI distributions for males (1) vs females (2).

- On average, females have a higher median BMI, about 2 kg/m² more than males. Their BMI values also spread out more in the middle range (wider box).
- Looking at the top of the plots, some females have very high BMI values (over 50 kg/m²), showing more cases of severe obesity in this group.
- There is still a lot of overlap between males and females. Most people from both groups have BMI between 22 and 35 kg/m². However, the shift in the middle values is still important for health analysis.

More graphs/charts are included in source code file

3. [Marks: 10] Do data cleaning/pre-processing as required and explain what you have done for your dataset and why?

Yes, both data cleaning and pre-processing steps are required for the scope of the project.

Data Cleaning

Semantic Relabelling

To remove code-level ambiguity, the original survey variable names were renamed to meaningful labels. For example, RIDAGEYR to Age. This change helps make the data structure clearer for analysis and collaboration.

Type Conversion

Continuous variables were converted to float64 or int64 types, while nominal variables, such as Gender, Diabetes_Status, and the newly added feature Age_Bin were recast as categorical features.

Integrity and Duplication Assessment

Basic data checks using functions like info(), describe() confirmed that there were no missing values in any column. Uniqueness of Respondent_ID confirmed a one-to-one mapping between respondents and records, ensuring confirming data consistency and enabling valid statistical inference.

Construction of 20-Year Age Bins

A new variable was created by dividing the continuous Age variable into five age groups: 0–19, 20–39, 40–59, 60–79, and 80 years or older. This grouped version of age allows for easier comparison between age ranges, while the original age variable was kept for modeling.

Outlier Evaluation and Retention

Boxplots and Z-score histograms were used to check for unusual values in BMI, Glucose, and Insulin. Although some high values were found, they were within reasonable biological limits. Since the project focuses on identifying high-risk metabolic conditions, these values were kept to preserve the real-world variation in the data.

Data Preprocessing

Feature Engineering

Two new features were constructed to enhance the representational capacity of the dataset.

First, a glucose-to-insulin ratio was derived to capture potential nonlinear interactions between blood glucose and insulin levels, while preventing division by zero through a small constant adjustment.

Second, Body Mass Index (BMI) was grouped into four categories: Underweight, Normal, Overweight, and Obese. These groups were based on commonly used medical standards and make it easier to study patterns across weight categories.

Handle Missing Values

Missing values were addressed in a two-step process to make sure the dataset was complete. For numerical columns, missing values were filled in using the average (mean) of each column. Then, for non-numeric columns, any remaining missing values were replaced using the most common value (mode)

in that column. After this process, there were no missing values in the dataset, making it ready for further analysis and encoding.

Categorical Variable Encoding

To facilitate compatibility with machine learning models, categorical variables were transformed using one-hot encoding. This encoding was applied to selected features, including gender, BMI category, and predefined age groupings. To reduce redundancy, the first category of each encoded variable was excluded from the final feature set.

Feature Selection

The target variable identified for prediction was the individual's diabetes status. All remaining variables, excluding the target, were considered as potential features of predictors. The final feature matrix was defined by excluding the target from the set of all available variables.

Feature Scaling

All numerical features in the input matrix were standardized using StandardScaler. This process adjusted each numeric column to have a mean of zero and a standard deviation of one. This ensures that all numeric variables are on the same scale before training a model.

Train-Test Split

The dataset was divided into training and testing sets using an 80/20 split. The target variable was separated from the input features, and the split was performed with a fixed random seed to ensure reproducibility. Stratified sampling was applied based on the target variable, so that class proportions in the training and testing sets are consistent with the original distribution. This approach helps maintain class balance and supports reliable model evaluation on unseen data.

4. [Marks: 15] Implement 2 machine learning models and explain which algorithms you have selected and why. Compare them and show success metrics (Accuracy/ RMSE/ Confusion Matrix) as per your problem. Explain results.

We implemented 2 machine learning models of SVM and Decision Tree using scikit-learn package

Support Vector Machine (SVM)

- Effective in high-dimensional spaces with small to medium size datasets, as the dataset has a size ~6000 instances for the usage.
- It is good for handling the imbalance and overlapping classes as the ratio of non-diabetic and diabetic is imbalanced.
- Captures non-linear patterns in different features and is effective with high-dimensional. medical data, especially after feature engineering and normalization

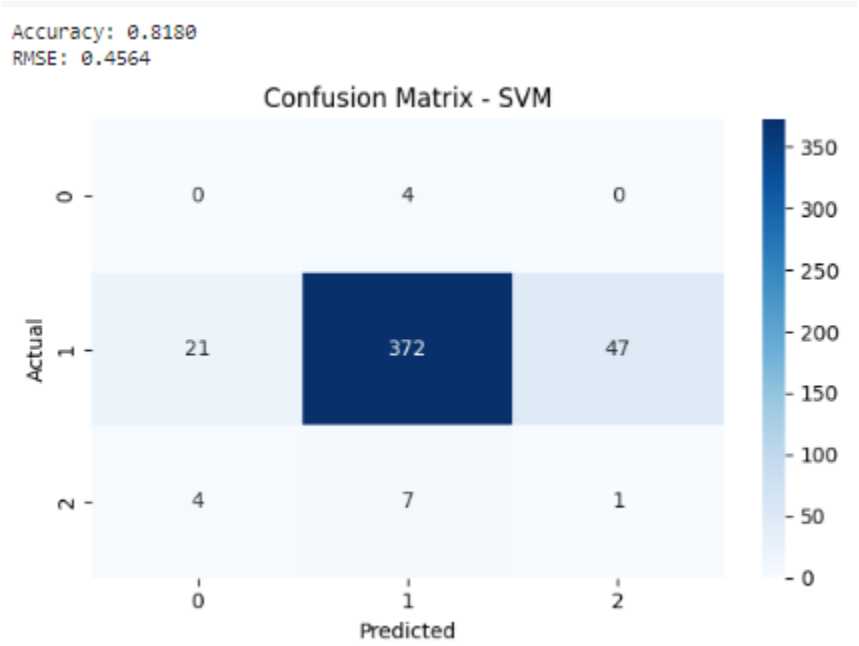


Figure 6. SVM Evaluation

From the result of the SVM model , we could see that the SVM classifier performs well on the dominant class (1: Diabetic) due to more training examples. However, it performs poorly on underrepresented classes (0 and 2), possibly due to class imbalance. From the confusion matrix, 441 samples were correctly classified as class 2 (the dominant class). 3 samples from class 1 and 11 from class 3 all predicted as class 2.The SVM achieves good overall accuracy of 0.83 and lowest RMSE, but it overfits to the majority class, failing to detect minority diabetes statuses.

Decision Tree Evaluation

- It is highly interpretable and intuitive, as it allows users to trace back decisions to specific thresholds.
- Captures non-linear dependencies between multiple continuous features like age, BMI, and insulin level.
- Robust to missing or noisy data.

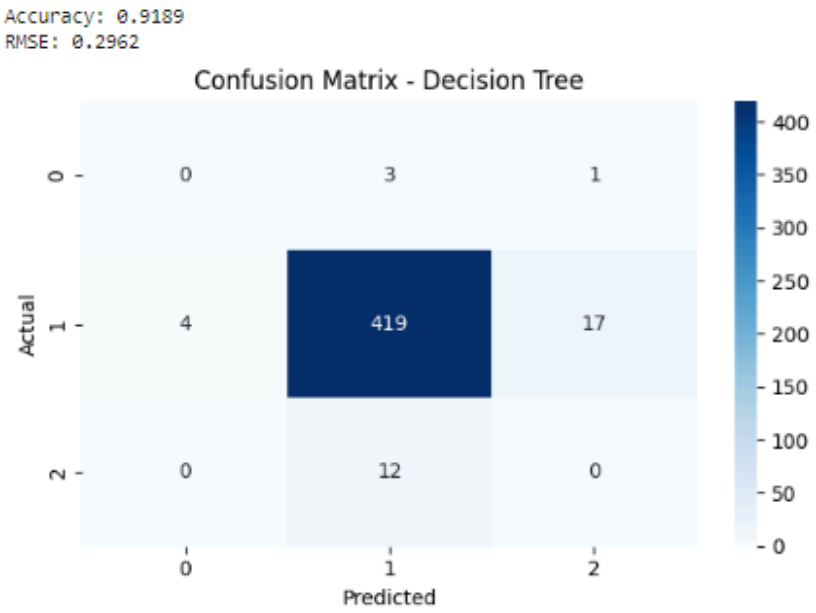


Figure 7. Decision Tree Evaluation

The Decision Tree performs well on the dominant class but fails on minority classes with accuracy of 0.9189. RMSE is 0.2962 which is a relatively low prediction error on average as it can predict good results on average. For the confusion matrix, we can see that the majority of predictions (419 out of 456) are correct yet most errors happen in the minority class(1 and 3 misclassified as class 2) and they are possibly resulted from the imbalance in the dataset.

5. [Marks: 15] Deploy a run-time pipeline for your dataset using Azure Designer Studio. Or do hyperparameter tuning for your algorithms. Explain your results. Or use Automated ML for your data set. Explain the best model results.

We would do hyperparameter tuning for our selected algorithm(SVM and Decision Tree)

We use Grid Search for hyperparameter tuning for both models

SVM Tune

```
# Define parameter grid for SVM
param_grid = {
    'C': [0.1, 1, 10],
    'kernel': ['linear', 'rbf', 'poly'],
    'gamma': ['scale', 'auto']
}

# Set up GridSearchCV
grid_search = GridSearchCV(
    SVC(), param_grid, cv=5, scoring='accuracy', n_jobs=-1, verbose=1
)

# Train
grid_search.fit(X_train, y_train)

# Best model
best_svm = grid_search.best_estimator_
print("\nBest SVM Parameters:", grid_search.best_params_)
```

```
Fitting 5 folds for each of 18 candidates, totalling 90 fits

Best SVM Parameters: {'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}

Accuracy: 0.9649
RMSE: 0.1873
```

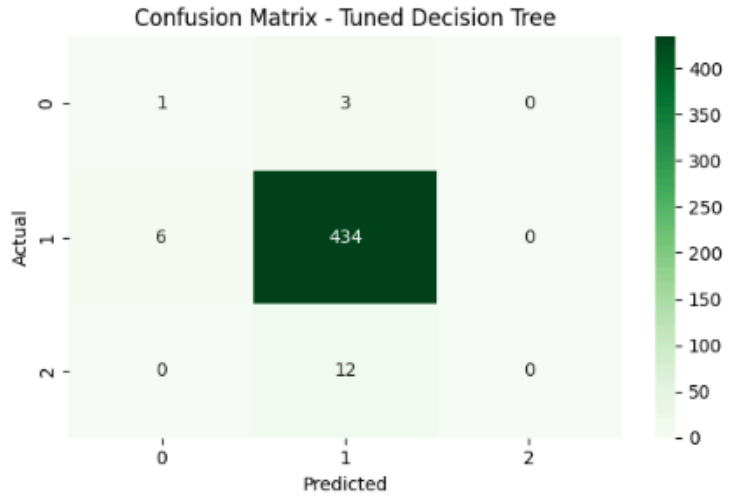
Figure8. Tuned SVM Evaluation

The tuned SVM model has a high accuracy of 0.9649 and low RMSE of 0.187 which is a good result for prediction. However, from the confusion matrix, we can see that all predictions go to class 1 and all other classes are misclassified which is resulted from the class imbalance. The model could detect the majority class(Class 2) correctly yet it could not detect the minority class(Class 0 and 1).

Decision Tree Tune

```
# Define hyperparameter grid
param_grid = {
    'max_depth': [3, 5, 10, None],
    'min_samples_split': [2, 5, 10],
    'criterion': ['gini', 'entropy'],
    'min_samples_leaf': [1, 2, 4]
}

# Grid search
grid_search = GridSearchCV(
    DecisionTreeClassifier(random_state=42),
    param_grid,
    cv=5,
    scoring='accuracy',
    n_jobs=-1,
    verbose=1
)

# Train
grid_search.fit(X_train, y_train)

# Best model
best_tree = grid_search.best_estimator_
print("\nBest Decision Tree Parameters:", grid_search.best_params_)
```

```
Fitting 5 folds for each of 72 candidates, totalling 360 fits

Best Decision Tree Parameters: {'criterion': 'entropy', 'max_depth': 3, 'min_samples_leaf': 4, 'min_samples_split': 2}

Accuracy: 0.9539
RMSE: 0.2146
```




```
Transmission succeeded: Item received: 3. Items accepted: 3
```

Figure 9. Tuned Decision Tree Evaluation

The tuned Decision Tree model achieved a high accuracy of 0.954 and a low RMSE of 0.2146, indicating strong overall predictive performance. The confusion matrix reveals that the model correctly classified 434 out of 442 samples in the majority class (Class 1), and managed to identify 1 out of 4 early-stage diabetes cases (Class 0), while still failing to detect advanced cases (Class 2) We could see that the dataset is an unbalanced dataset in the class of diabetes and resulted in the overall overfitting for majority class(Class 1)