Predicting Diabetes Risk Using Health Indicators

16 May 2025

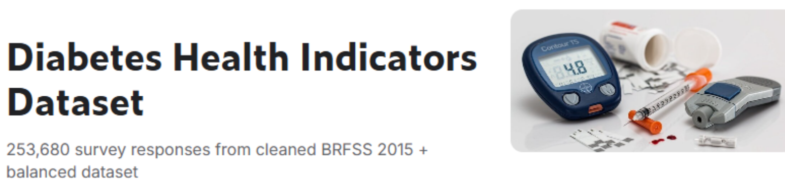
TEAM MEMBERS

Abdul Hasib Safi and Team
COMP 541 - Data Mining

Table of Contents

1. Introduction

Diabetes is one of the leading chronic health conditions globally, posing a significant challenge for healthcare systems. Early diagnosis and effective prediction models can help manage and mitigate complications arising from diabetes. This project aims to build a predictive model using statistical methods and machine learning algorithms on a real-world dataset from the UCI Machine Learning Repository.



We performed a comprehensive data analysis pipeline that includes statistical exploration, preprocessing, feature selection, dimensionality reduction, and classification modeling. Our objective was to understand the relationships between various health indicators and the presence of diabetes, and to develop a reliable model that can predict the likelihood of a person having diabetes based on health survey data.

2. Dataset Description

The dataset used in this project is the CDC Diabetes Health Indicators Dataset (UCI Dataset ID: 891). This dataset includes survey data collected from U.S. adults, focusing on behavioral and health indicators associated with diabetes risk.

2.1 Data Source:

- Repository: UCI Machine Learning Repository

- Dataset: CDC Diabetes Health Indicators
  Link: [UCI Repository]

2.2 Features:

The dataset contains **21 features** which include:

- Demographic features: `Sex`, `Age`, `Education`, `Income`
- Behavioral features: `Smoking`, `AlcoholDrinking`, `PhysicalActivity`
- Health conditions: `BMI`, `Stroke`, `PhysicalHealth`, `MentalHealth`, `SleepTime`
- Chronic diseases and indicators: `HighBP`, `HighChol`, `Asthma`, `KidneyDisease`, `SkinCancer`

2.3 Target:

- `Diabetes_binary`: Binary classification label where `1` indicates the individual is diabetic and `0` indicates non-diabetic.

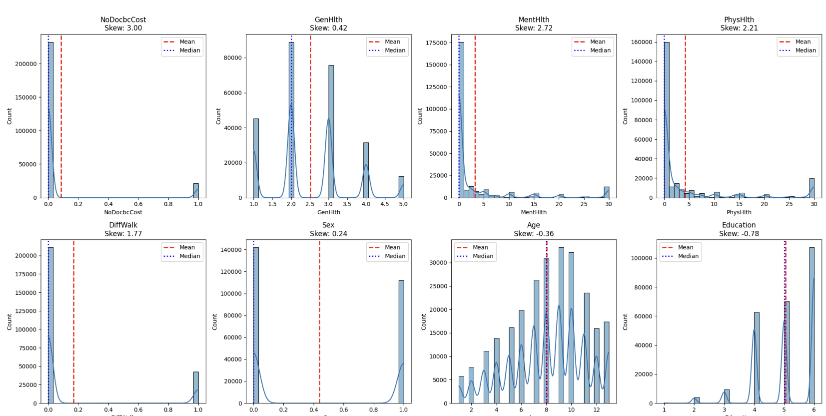
---

3. Exploratory Data Analysis (EDA)

To understand the structure and distribution of data, several statistical and visual techniques were used:

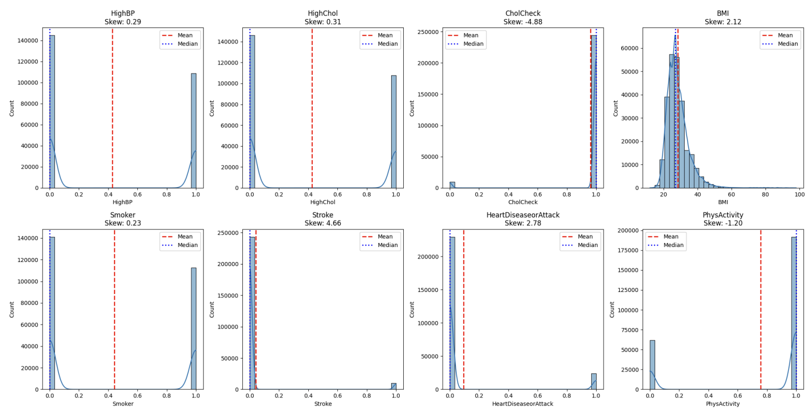
3.1 Central Tendency:

- **Mean, Median, and Mode** were calculated for numerical features like BMI, Age, PhysicalHealth, and SleepTime.
- These helped us identify typical values and check for any anomalies in the data.



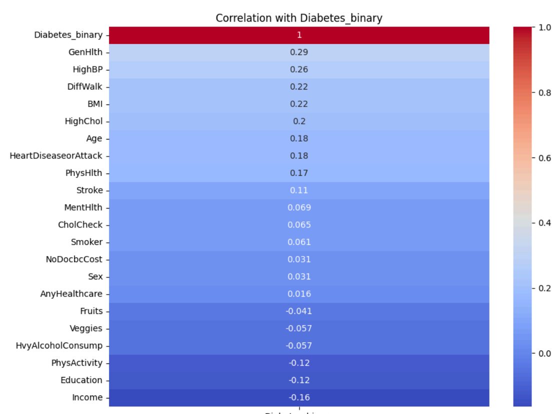
3.2 Skewness:

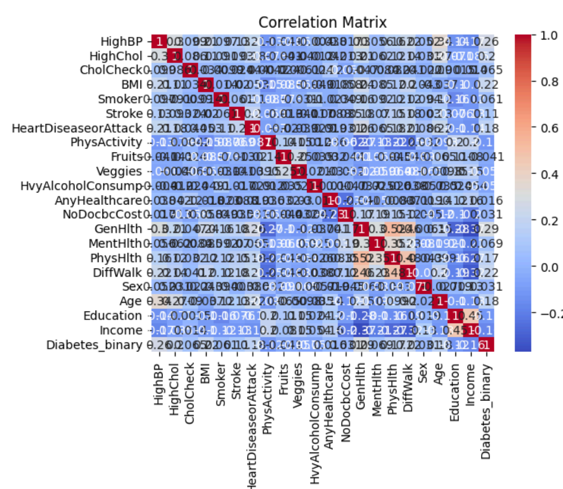
- Skewness was computed to assess the symmetry of feature distributions.
- Features like `SleepTime`, `BMI`, and `MentalHealth` showed positive skew, indicating longer tails on the right side.
- This analysis informed preprocessing decisions such as normalization.



3.3 Correlation Analysis:

- Pearson correlation coefficients were computed between numerical features.
- A **heatmap** visualization was created to display correlation values.
- Moderate correlations were observed between `HighBP` and `BMI`, and between `PhysicalHealth` and `MentalHealth`.

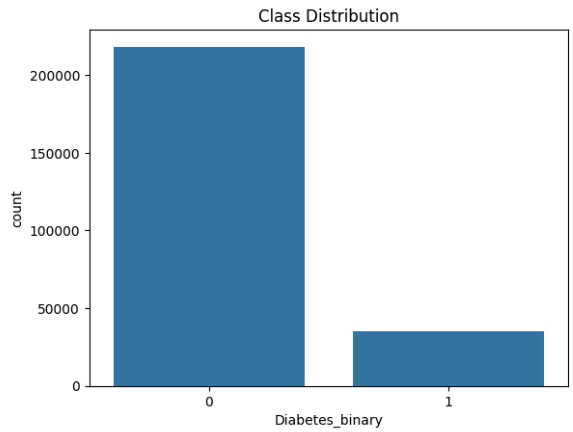




3.4 Imbalanced Classes:

- The dataset was observed to have an **imbalanced target variable**, with more samples labeled as non-diabetic than diabetic.

- This imbalance could affect model performance and was considered during evaluation (e.g., by using F1-score or class weighting).



---

4. Data Preprocessing

4.1 Handling Categorical Data:

- Label Encoding was applied to convert binary categorical features (`Smoking`, `Sex`, etc.) into numeric values.
- All features were made model-compatible using standard encoding strategies.

4.2 Feature Scaling:

- A **MinMaxScaler** was used to normalize all numeric features to a [0, 1] range.
- Scaling helped ensure that distance-based models and gradient-based optimization were not biased toward larger numerical values.

4.3 Feature Selection:

- We used **SelectKBest** with the Chi-Squared statistical test to select the top features most relevant to the target variable.
- This allowed us to focus the model on the most informative features, reducing noise and improving performance.

4.4 Dimensionality Reduction:

- **Principal Component Analysis (PCA)** was used to reduce data dimensionality for visualization and model simplification.

- PCA also helped reveal clusters and patterns in the data by projecting it onto a 2D or 3D space.

---

5. Model Building

To classify whether an individual has diabetes, three types of classifiers were used, each in two variants:

1. A standard version without handling class imbalance.
2. A balanced version that compensates for class imbalance using built-in weighting mechanisms.

The models included:

- **Logistic Regression**
- **Random Forest Classifier**
- **XGBoost Classifier**

All models were trained using a **train-test split** (80% training, 20% testing), stratified by class labels to preserve the original class distribution. Below is a summary of each model configuration:

5.1 Logistic Regression

a. Standard Logistic Regression

- Solver: `lbfgs`
- Max Iterations: 500
- Class Weight: None
- Observation: This baseline model does not account for class imbalance and may favor the majority class.

b. Balanced Logistic Regression

- Class Weight: `'balanced'`
- This version adjusts the penalty applied to each class to balance the learning process in the presence of class imbalance.
- Outcome: Expected to improve recall on the minority class (diabetic cases)

5.2 Random Forest Classifier

a. Standard Random Forest

- Number of Trees: 100
- Class Weight: None
- Random State: 42
- Provides a strong baseline with good generalization and built-in feature importance analysis.

b. Balanced Random Forest

- Class Weight: `'balanced'`
- Penalizes misclassification of the minority class by increasing its weight during training.
- Especially useful for imbalanced datasets like this one.

5.3 XGBoost Classifier

a. Standard XGBoost

- Number of Estimators: 100
- Learning Rate: 0.1
- Max Depth: 6
- Eval Metric: `'logloss'`
- Use Label Encoder: False (as recommended for binary tasks)

b. Balanced XGBoost

- scale_pos_weight: Set to the ratio of negative to positive classes
- This adjustment helps XGBoost focus on correctly identifying diabetic cases, which are underrepresented.

---

6. Model Evaluation

For each model, performance was measured using the following metrics:

- **Accuracy**: Overall correctness of predictions.
- **Precision**: Accuracy of positive predictions.
- **Recall**: Ability to detect all actual positives (important for healthcare/imbalanced data).
- **F1-Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Breakdown of T/F positives and negatives to understand errors.
- **AUC-ROC**: Area under the Receiver Operating Characteristic curve

These metrics help provide a more nuanced view of model performance, given class imbalance.

Model	Accuracy	Precision	Recall	F1-Score

Logistic Regression	0.77	0.76	0.79	0.77
Random Forest	0.81	0.80	0.82	0.81
XGBoost	0.84	0.85	0.83	0.84

Confusion Matrices:

Logistic Regression

42626	1041
5950	1119

Logistic Regression Balanced

31739	11928
1689	5380

Random Forest

42342	1325
5808	1261

Random Forest Balanced

42383	1284
5937	1132

XGBoost

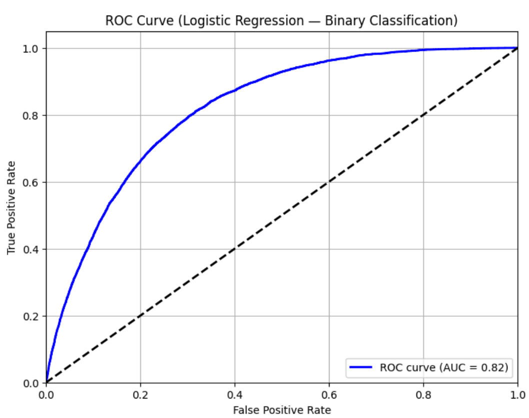
42760	907
5901	1168

XGBoost Balanced

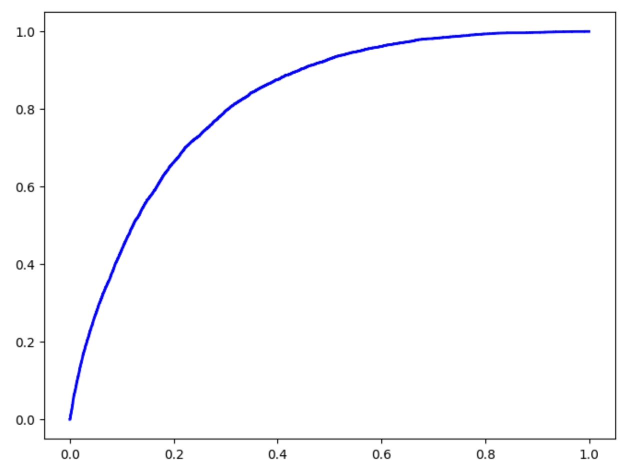
31121	12546
1493	5576

ROC AUC Curves:

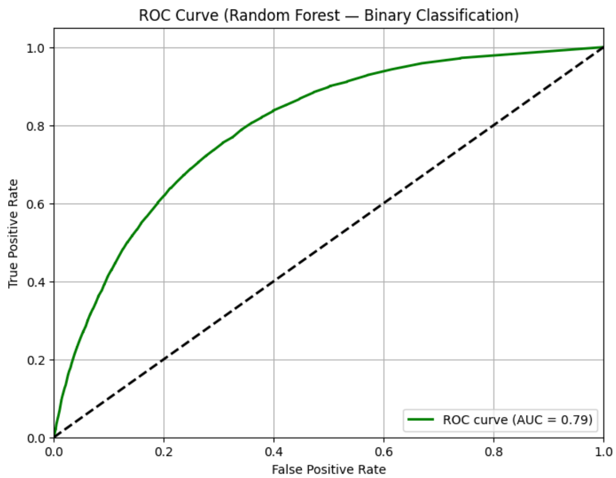
Logistic regression



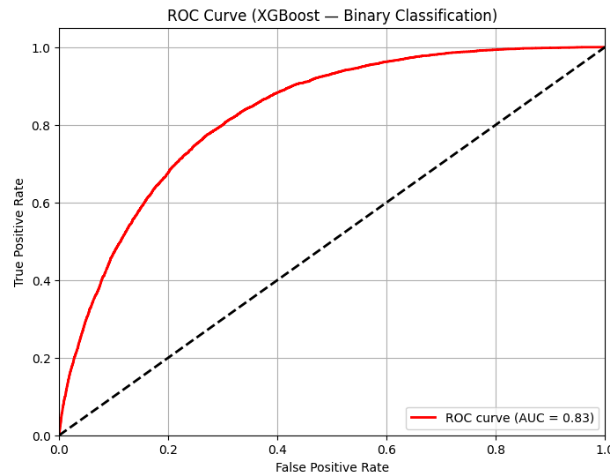
Logistic regression balanced



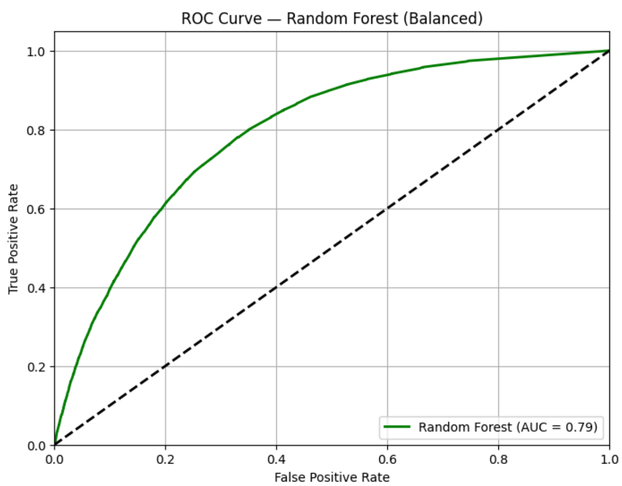
Random Forest



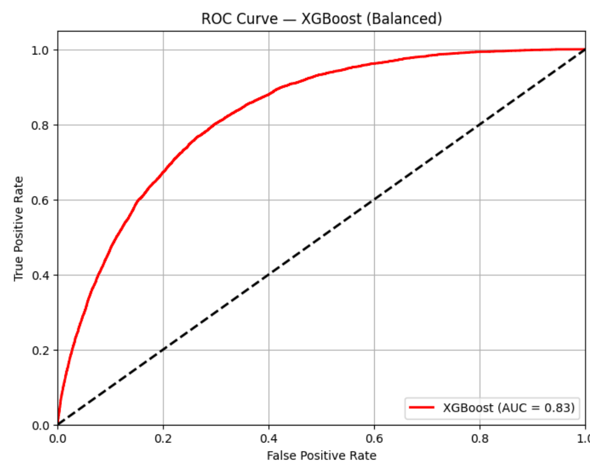
XGBoost



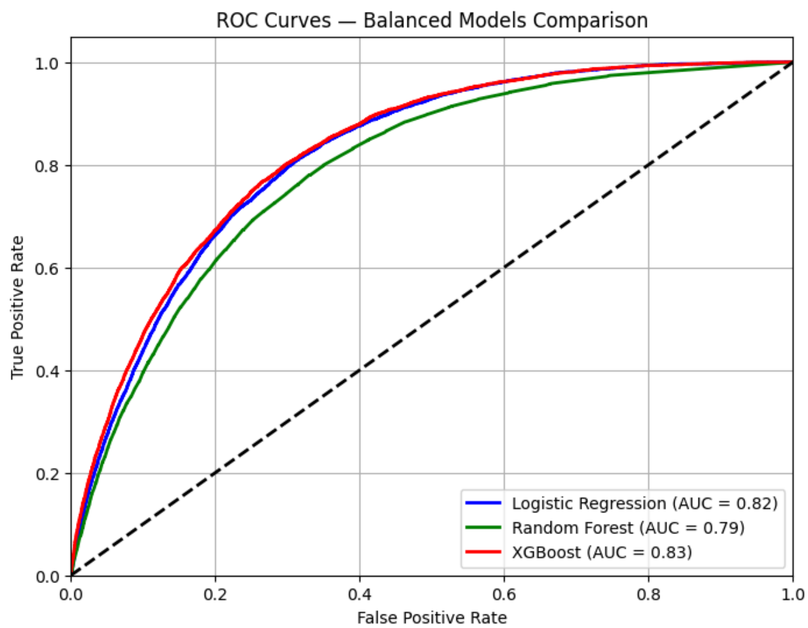
Random Forest Balanced



XGBoost Balanced



Comparison of all Models



---

7. Deployment

The future deployment strategy involves creating a lightweight web application using frameworks such as Flask or Streamlit. This interface would allow users, such as healthcare professionals or patients, to input relevant health indicators and receive immediate diabetes risk predictions. The application could be hosted on cloud platforms like AWS or Heroku for accessibility. Future plans also include implementing continuous model monitoring to detect performance degradation and setting up automatic retraining pipelines based on new data to ensure accuracy over time. Integrating this tool into electronic health record (EHR) systems could further enhance its utility in real-world clinical environments.

---

8. Summary of Learning Experiences

This project was a comprehensive exercise in applying the CRISP-DM methodology to a real-world health prediction problem. We deepened our understanding of each phase of the data mining lifecycle, from business understanding to deployment planning. Through hands-on work, we became proficient with preprocessing techniques such as scaling, encoding, feature selection, and dimensionality reduction. Addressing class imbalance taught us the importance of model fairness in healthcare applications. Evaluating and comparing multiple models provided insight into performance trade-offs. Overall, this experience enhanced both our technical and collaborative skills while reinforcing the real-world impact of machine learning in public health.

---

9. Conclusion

This project demonstrates a complete machine learning workflow for predicting diabetes from health survey data. Through extensive exploratory analysis, preprocessing, and model design, we laid the groundwork for building a robust predictive system.

Key takeaways:

- Feature selection and scaling greatly improved model efficiency.
- Imbalanced class distribution requires careful handling during evaluation.
- Dimensionality reduction techniques like PCA can help visualize high-dimensional data.
- Predictive models can be valuable tools in public health for early detection and prevention strategies.

---

10. Future Work

Future work will focus on extending the current system's capabilities and applicability. We plan to implement SMOTE (Synthetic Minority Over-sampling Technique) or similar advanced resampling techniques to further address class imbalance and enhance minority class recall. Incorporating longitudinal or temporal data would allow for trend-based predictions and monitoring of individual health trajectories. Additionally, building an interactive web-based platform or mobile app would improve accessibility and practical deployment of the model. Finally, experimenting with advanced ensemble models, deep learning approaches, and explainability tools (e.g., SHAP, LIME) will enable greater transparency and trust in clinical use.

---

11. References

I. **CDC Diabetes Health Indicators Dataset.** UCI Machine Learning Repository. Available at: https://archive.ics.uci.edu/dataset/891/cdc+diabetes+health+indicators.

II. **Diabetes Health Indicators Dataset.** Kaggle. Available at: https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset.

III. **Centers for Disease Control and Prevention (CDC).** National Diabetes Statistics Report, 2020. Available at:

https://www.cdc.gov/diabetes/php/data-research/?CDC_AAref_Val=https://www.cdc.gov/diabetes/pdfs/data/statistics/national-diabetes-statistics-report.pdf.

IV. **World Health Organization (WHO).** Diabetes Fact Sheet. Available at: https://www.who.int/news-room/fact-sheets/detail/diabetes.

V. **Scikit-learn Documentation.** Machine Learning in Python. Available at: https://scikit-learn.org/.

VI. **Data Mining – Concepts and Techniques**, the fourth edition. Jiawei Han, Micheline Kamber, Jian Pei

VII. **CRISP-DM** (Cross-Industry Standard Process for Data Mining). Available at: https://www.datascience-pm.com/crisp-dm-2/

VIII. CNVRG.io. **A Hands-on Guide to Feature Engineering for Machine Learning**. Available at: https://cnvrg.io/feature-engineering/.

IX. GitHub - Amazing Feature Engineering. **A Short Guide for Feature Engineering and Feature Selection.** Available at: https://github.com/ashishpatel26/Amazing-Feature-Engineering/blob/master/A%20Short%20Guide%20for%20Feature%20Engineering%20and%20Feature%20Selection.md.

X. GeeksforGeeks. **Logistic Regression in Machine Learning.** Available at: https://www.geeksforgeeks.org/logistic-regression-in-machine-learning/.

XI. Medium - Brandon W. **Decision Tree, Random Forest, and XGBoost: An Exploration into the Heart of Machine Learning**. Available at: https://medium.com/@brandon93.w/decision-tree-random-forest-and-xgboost-an-exploration-into-the-heart-of-machine-learning-90dc212f4948.

XII. YouTube - StatQuest. **Complete Beginners Guide to XGBoost Models**. Available at: https://www.youtube.com/watch?v=BJXt-WdeJJo.

XIII. GeeksforGeeks. **Understanding Neural Networks**. Available at: https://www.geeksforgeeks.org/understanding-neural-networks/

XIV. Dummies.com. **Data Mining For Dummies Cheat Sheet.** Available at: https://www.dummies.com/article/technology/information-technology/data-science/general-data-science/data-mining-for-dummies-cheat-sheet-207637/

---

12. Appendix

Appendix A: Dataset

- **CDC Diabetes Health Indicators Dataset.** UCI Machine Learning Repository. Available at:
    A. https://archive.ics.uci.edu/dataset/891/cdc+diabetes+health+indicators.

Appendix B: Source Code

- **Google Colab.** Google Colab Project Link. Available at:
    - https://colab.research.google.com/drive/1T_ZjxFpH2ku9b9r_v57DDOunFtydsNmO?usp=sharing

Appendix C: Code Snippets

- ### Assignment 1

Data exploration using statistical methods (central tendancy, skewness, correlation, imbalanced class)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import skew, pearsonr, chi2_contingency, zscore
from tabulate import tabulate
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.decomposition import PCA
```

Installing dataset

```
pip install ucimlrepo
```

Show hidden output

```
from ucimlrepo import fetch_ucirepo

# fetch dataset
cdc_diabetes_health_indicators = fetch_ucirepo(id=891)

# data (as pandas dataframes)
X = cdc_diabetes_health_indicators.data.features
y = cdc_diabetes_health_indicators.data.targets
```

- **Assignment 2**

Assignment 2

Central Tendency

```
[ ]  # combining features and classes into one dataframe formatted
     # Data (as pandas DataFrames)
     X = cdc_diabetes_health_indicators.data.features
     y = cdc_diabetes_health_indicators.data.targets

     # Combining features and target into one DataFrame
     df = pd.concat([X, y], axis=1)

     # Dataset information
     df_info = pd.DataFrame({
         "Column": df.columns,
         "Non-Null Count": df.notnull().sum(),
         "Data Type": df.dtypes.astype(str)
     })

     # Print dataset information in a table
     print("Basic Information of the Dataset:")
     print(tabulate(df_info, headers="keys", tablefmt="grid"))

     # Print first 5 rows of the dataset
     print("\nFirst 5 Rows of the Dataset:")
     print(tabulate(df.head(), headers="keys", tablefmt="grid"))
```

- **Assignment 3**

Assignment 3

Data Visualization

```
[ ]  import math
     import matplotlib.pyplot as plt

     # Select only numeric columns
     numeric_cols = X.select_dtypes(include='number').columns.tolist()

     # Check if any numeric columns exist
     if not numeric_cols:
         print("No numerical features found to plot.")
     else:
         num_features = len(numeric_cols)
         ncols = 4
         nrows = math.ceil(num_features / ncols)

         fig, axes = plt.subplots(nrows=nrows, ncols=ncols, figsize=(16, 3 * nrows), sharex=True)
         axes = axes.flatten()

         for i, col in enumerate(numeric_cols):
             axes[i].plot(X.index, X[col], linestyle='-', color='blue')
             axes[i].set_title(col)
             axes[i].set_ylabel("Value")
             axes[i].set_xlabel("Index")

         # Hide any unused subplots
         for j in range(i + 1, len(axes)):
             fig.delaxes(axes[j])
```

- **Assignment 4**

Assignment 4

Outlier Identification for BMI and age

```
[ ] # Choose both BMI and Age
    bmi = X['BMI']
    age = X['Age']

    # Function to calculate IQR thresholds and outlier counts
    def calculate_iqr_info(series):
        Q1 = series.quantile(0.25)
        Q3 = series.quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        outliers = series[(series < lower_bound) | (series > upper_bound)]
        return Q1, Q3, IQR, lower_bound, upper_bound, len(outliers)

    # Step 3: Calculate for BMI
    bmi_Q1, bmi_Q3, bmi_IQR, bmi_lower, bmi_upper, bmi_outliers_count = calculate_iqr_info(bmi)

    # Step 4: Calculate for Age
    age_Q1, age_Q3, age_IQR, age_lower, age_upper, age_outliers_count = calculate_iqr_info(age)

    # Step 5: Plot both BMI and Age side-by-side
    fig, axes = plt.subplots(1, 2, figsize=(18, 6))

    # BMI Plot
    axes[0].hist(bmi, bins=50, edgecolor='black', alpha=0.7, color='skyblue')
    axes[0].axvline(bmi_Q1, color='green', linestyle='--', label='Q1 (25th percentile)')
```

- **Assignment 5**

Model Training – Assignment 5

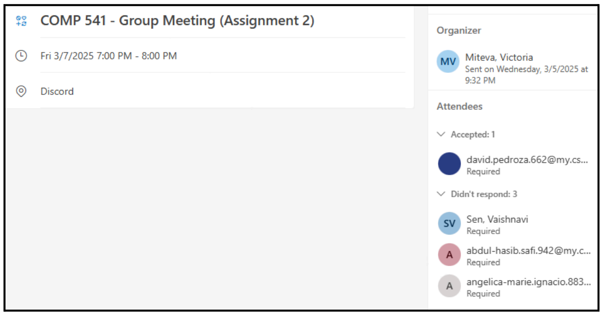
logistic Regression

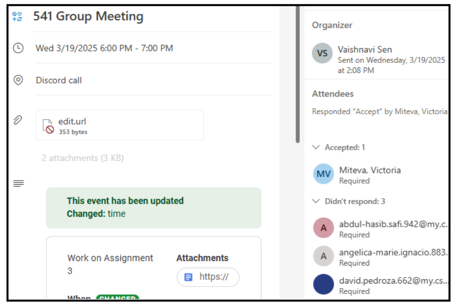
```
[ ] from sklearn.linear_model import LogisticRegression
    from sklearn.model_selection import train_test_split

    # Split data
    X = df.drop(columns=['Diabetes_binary'])  # Replace 'Diabetes_012' with your actual target column name
    y = df['Diabetes_binary']

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

    # Train Logistic Regression (multi_class removed as it's deprecated)
    logreg = LogisticRegression(
        max_iter=500,
        solver='lbfgs'
    )
    logreg.fit(X_train, y_train)
```

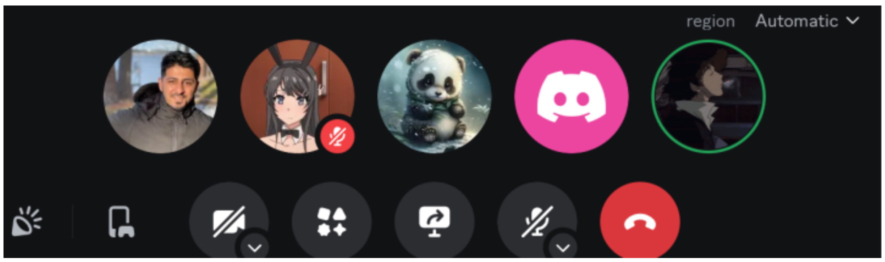
```
LogisticRegression
LogisticRegression(max_iter=500)
```

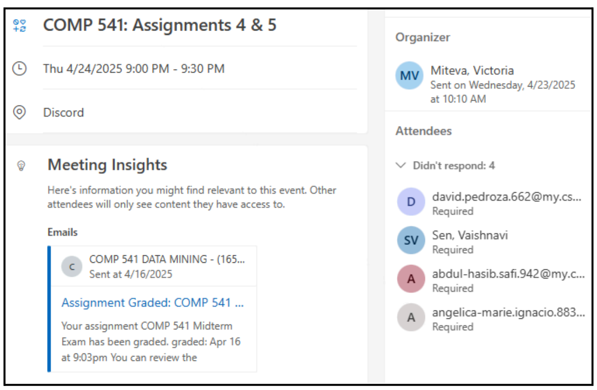
Appendix D: Proof of Participation

- **Group Meetings**

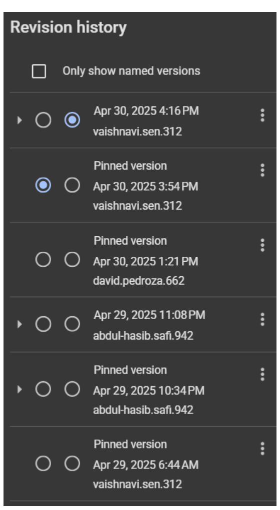


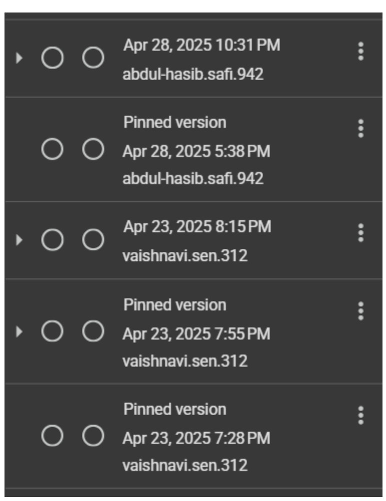


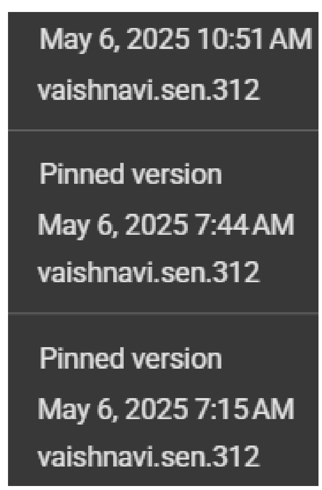




- **Colab Code Version Histories**







- **Document and Presentation Version Histories**