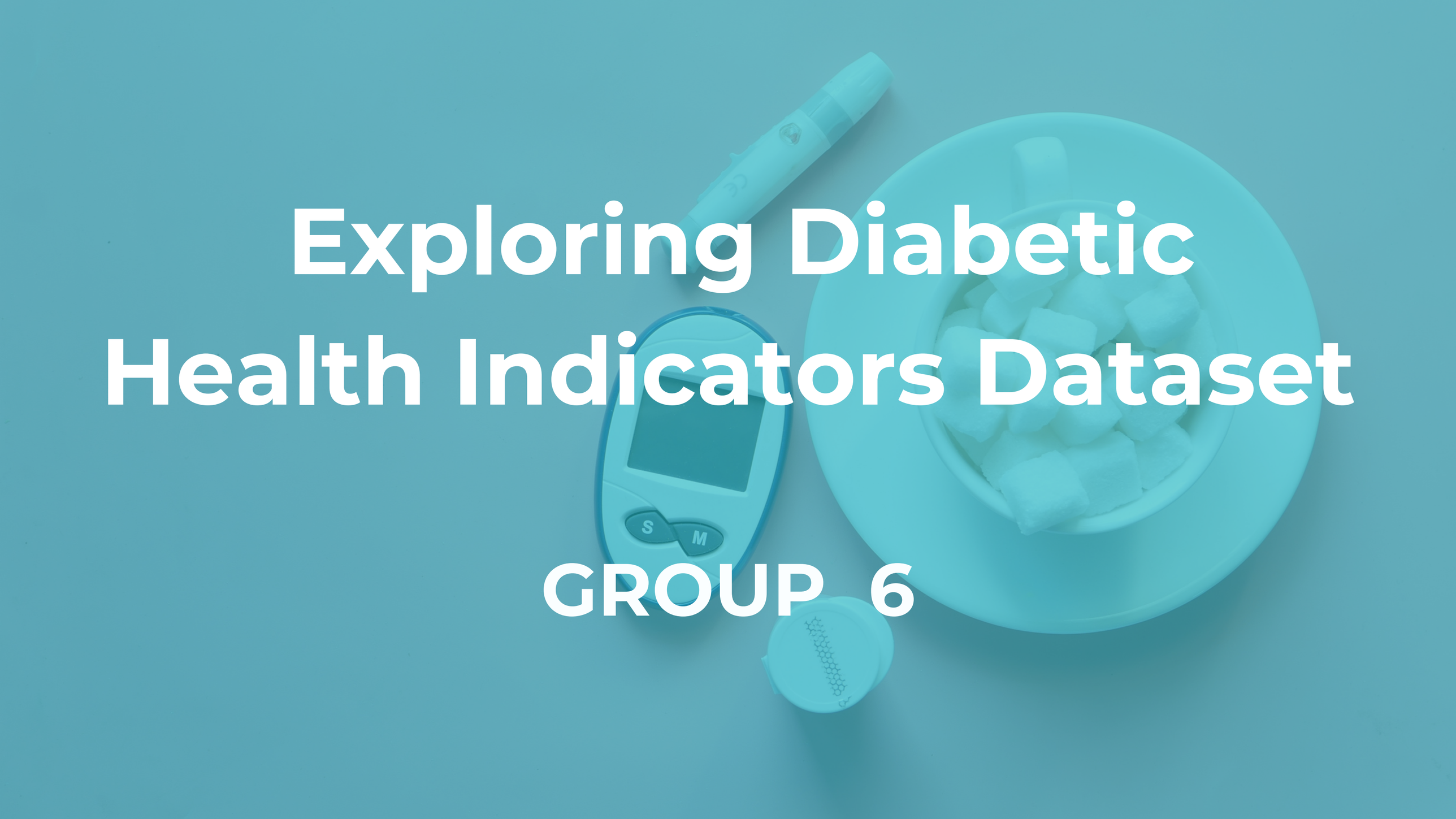


Exploring Diabetic Health Indicators Dataset

GROUP 6

TEAM MEMBERS:

JHARANA ADHIKARI: C0927442

UMA MAHESHWARI : C0928443

YASASWIN PALUKURI: C0928450

Overview

- Why Diabetes Health Indicators Dataset?
- Characteristics of Diabetic Health Indicator
- Lifecycle of Big Data Analytics
- Utilization of Analysis Result
- Conclusion
- Reference

Why Diabetes Health Indicators Dataset?

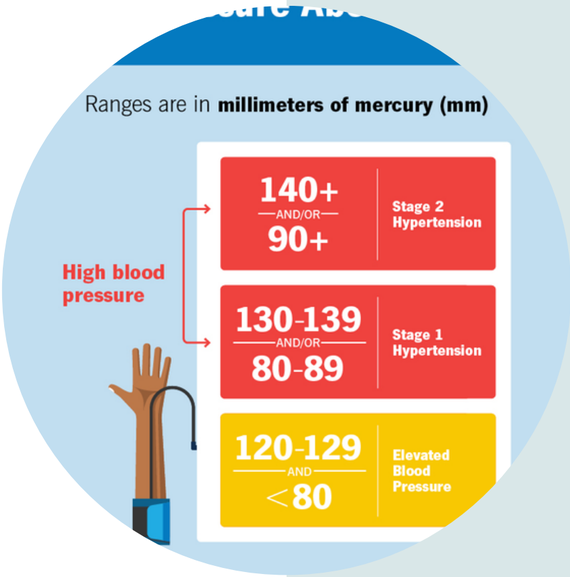
- Research on diabetes is essential to enhancing patient outcomes and lowering complications.
- It lessens the strain on healthcare systems by assisting in the development of cost-effective strategies.
- Research improves diabetics' overall health, early detection, and treatment.

Why Diabetes Health Indicators Dataset?

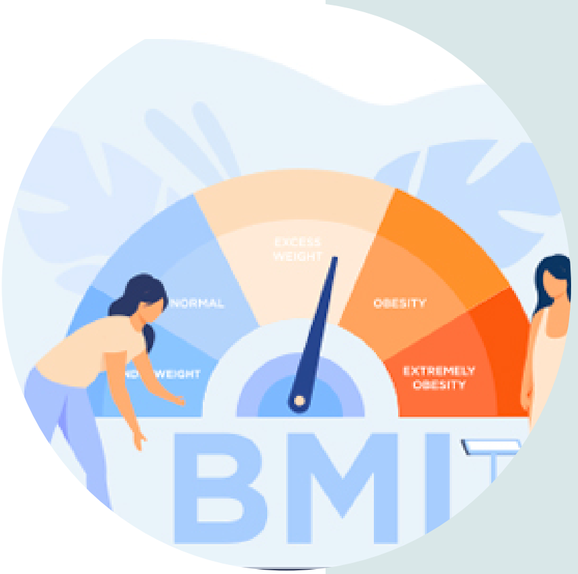
- Rich dataset with diverse health indicators provides ample features for building robust predictive models.

- Machine learning algorithms can leverage the complexity of the dataset to identify patterns and relationships, aiding in diabetes risk prediction and prevention strategies.

Characteristics of Diabetic Health Indicator Datasets:



- **Diabetes_012: Indicates 0 for no diabetes/pregnancy, 1 for prediabetes, and 2 for diabetes; crucial variable for analysis.**
- **HighBP: High blood pressure indicator; relevant for diabetes risk assessment.**
- **HighChol: High cholesterol indicator; associated with diabetes complications.**
- **CholCheck: Cholesterol check frequency; reflects health awareness.**
- **BMI: Body Mass Index; correlates with diabetes risk.**



Characteristics of Diabetic Health Indicator Datasets:

- **Smoker: Smoking status; impacts diabetes susceptibility.**
- **Stroke: Stroke history; potential complication of diabetes.**
- **HeartDiseaseorAttack: Heart disease history; linked to diabetes.**
- **PhysActivity: Physical activity level; influences diabetes prevention.**
- **Fruits: Fruit consumption; dietary factor in diabetes management.**
- **Veggies: Vegetable consumption; dietary factor affecting overall health and potentially diabetes risk.**

Characteristics of Diabetic Health Indicator Datasets:

- HvyAlcoholConsump: Heavy alcohol consumption; lifestyle factor potentially impacting diabetes risk and overall health.
- AnyHealthcare: Healthcare utilization; important for diabetes management.
- NoDocbcCost: Doctor visit affordability; affects healthcare access.
- GenHlth: General health perception; subjective health indicator.
- MentHlth: Mental health status; psychological aspect of diabetes.

Characteristics of Diabetic Health Indicator Datasets:

- **PhysHlth: Physical health status; relevant for diabetes complications.**
- **DiffWalk: Difficulty walking; potential diabetes symptom.**
- **Sex: Gender; demographic variable in diabetes research.**
- **Age: Age of respondents; correlates with diabetes prevalence.**
- **Education: Educational attainment; socio-economic factor in diabetes.**
- **Income: Household income; socio-economic determinant of health.**

BUSINESS CASE EVALUATION

- CDC reports 34.2 million Americans with diabetes and 88 million with prediabetes as of 2018.

- Shockingly, 1 in 5 diabetics and 8 in 10 prediabetics are unaware of their risk, highlighting the urgency for early detection.

- Type II diabetes, most common, costs roughly $327 billion for diagnosed cases, and nearly $400 billion annually including undiagnosed cases, disproportionately affecting lower socioeconomic groups.

Public Health Impact

Diabetes Risk Detection

Diabetes

DATA IDENTIFICATION

- **Dataset Overview: Clean dataset comprising 253,680 survey responses to the CDC's BRFSS2015.**
- **Target Variable: Diabetes_012 with 3 classes: 0 for no diabetes/pregnancy, 1 for prediabetes, and 2 for diabetes**
- **Feature Variables: 21 feature variables available for analysis, providing insights into diabetes risk factors and indicators.**

DATA ACQUISITION AND FILTERING

https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset/data

Diabetes Health Indicators Dataset

253,680 survey responses from cleaned BRFSS 2015 + balanced dataset

k kaggle.com

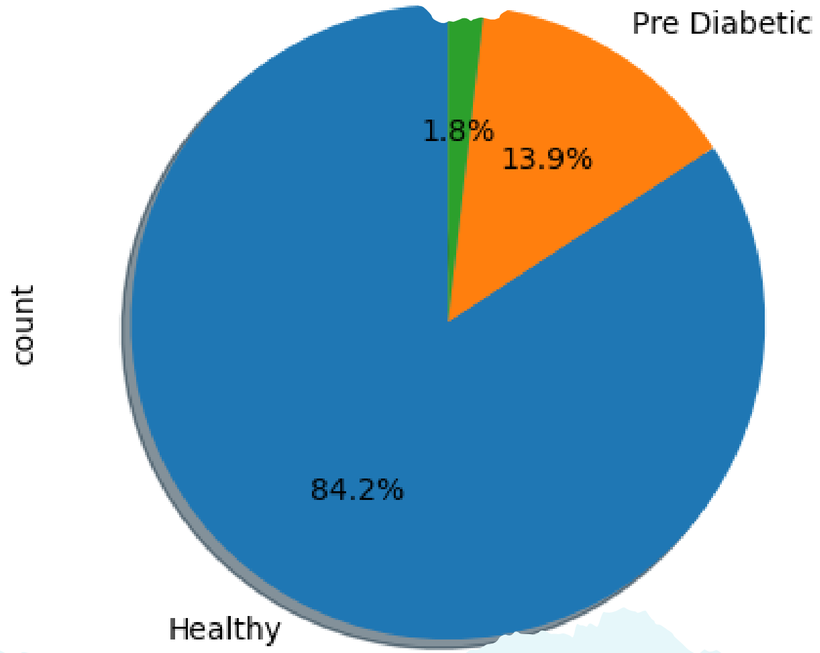
DATA EXTRACTION

- Important characteristics include lifestyle, medical history, demographics, and symptoms.
- Make use of the pandas library in Python to extract specific data.
- Use info, head, and shape to comprehend datasets.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 253680 entries, 0 to 253679
Data columns (total 22 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   Diabetes_012          253680 non-null  float64
 1   HighBP                253680 non-null  float64
 2   HighChol              253680 non-null  float64
 3   CholCheck             253680 non-null  float64
 4   BMI                   253680 non-null  float64
 5   Smoker                253680 non-null  float64
 6   Stroke                253680 non-null  float64
 7   HeartDiseaseorAttack  253680 non-null  float64
 8   PhysActivity          253680 non-null  float64
 9   Fruits                253680 non-null  float64
 10  Veggies               253680 non-null  float64
 11  HvyAlcoholConsump     253680 non-null  float64
 12  AnyHealthcare         253680 non-null  float64
 13  NoDocbcCost           253680 non-null  float64
 14  GenHlth               253680 non-null  float64
 15  MentHlth              253680 non-null  float64
 16  PhysHlth              253680 non-null  float64
 17  DiffWalk              253680 non-null  float64
 18  Sex                   253680 non-null  float64
 19  Age                   253680 non-null  float64
 20  Education             253680 non-null  float64
 21  Income                253680 non-null  float64
dtypes: float64(22)
memory usage: 42.6 MB
```

DATA AGGREGATION & REPRESENTATION



MIN MAX SCALER

Columns

```
df2.head()
```

]:

	Diabetes_012	HighBP	HighChol	BMI	Smoker	Stroke	HeartDiseaseorAttack	PhysActivity	Fruits	Veggies	HvyAlcoholConsump	GenHlth	MentHlth	P
0	0.0	1.0	1.0	40.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	5.0	18.0	
1	0.0	0.0	0.0	25.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	3.0	0.0	
2	0.0	1.0	1.0	28.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	5.0	30.0	
3	0.0	1.0	0.0	27.0	0.0	0.0	0.0	1.0	1.0	1.0	0.0	2.0	0.0	
4	0.0	1.0	1.0	24.0	0.0	0.0	0.0	1.0	1.0	1.0	0.0	2.0	3.0	

```
# copy Dataframeumns)
numerical_columns = ["Diabetes_012","BMI", "GenHlth", "MentHlth", "PhysHlth", "Age"]


df3 = df2.copy(deep=True)

# Selecting numerical columns (excluding binary/boolean col
# Initialize the MinMaxScaler
scaler = MinMaxScaler()

# Fit and transform the numerical features
df3[numerical_columns] = scaler.fit_transform(df3[numerical_columns])
```

DATA ANALYSIS

- EDA (Exploratary Data Analysis)
- Finding the correlation between variables
- Splitting the data into test and train subsets
- Build a machine-learning model
- Perform a cross-validation technique

PREDIABETES

in the United States in 2018

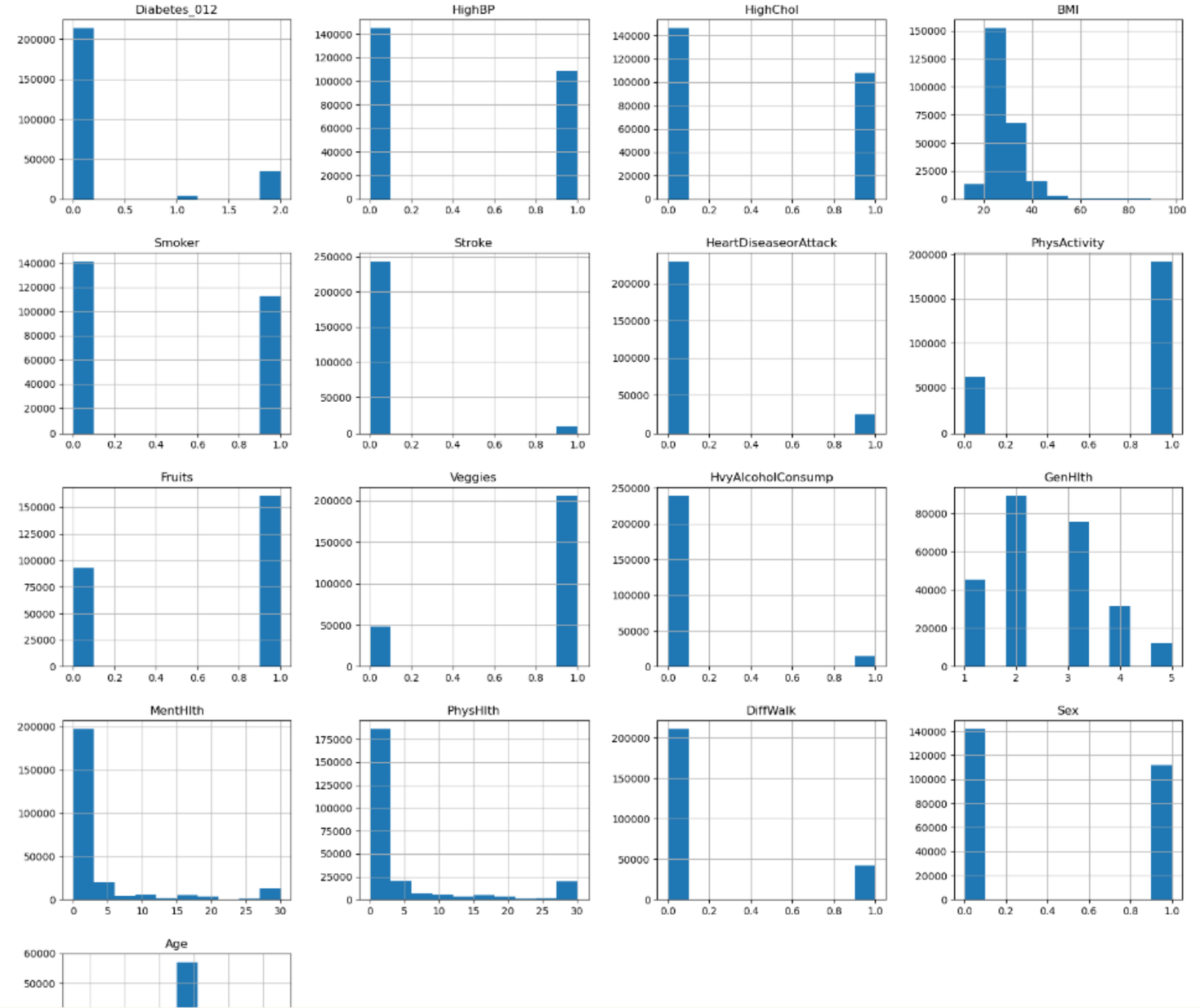
88 million
adults in the U.S.
–more than **1 in 3**–
were living with prediabetes
in 2018

84%
did not know they had it

EDA (EXPLORATARY DATA ANALYSIS)

7.1 EDA (Exploratary Data Analysis)

```
df2.hist(figsize = (20,20))
plt.show()
```



Finding the correlation between variables

```
plt.figure(figsize=(16,9))
sns.heatmap(df2.corr(), annot=True)
```

```
<Axes: >
```

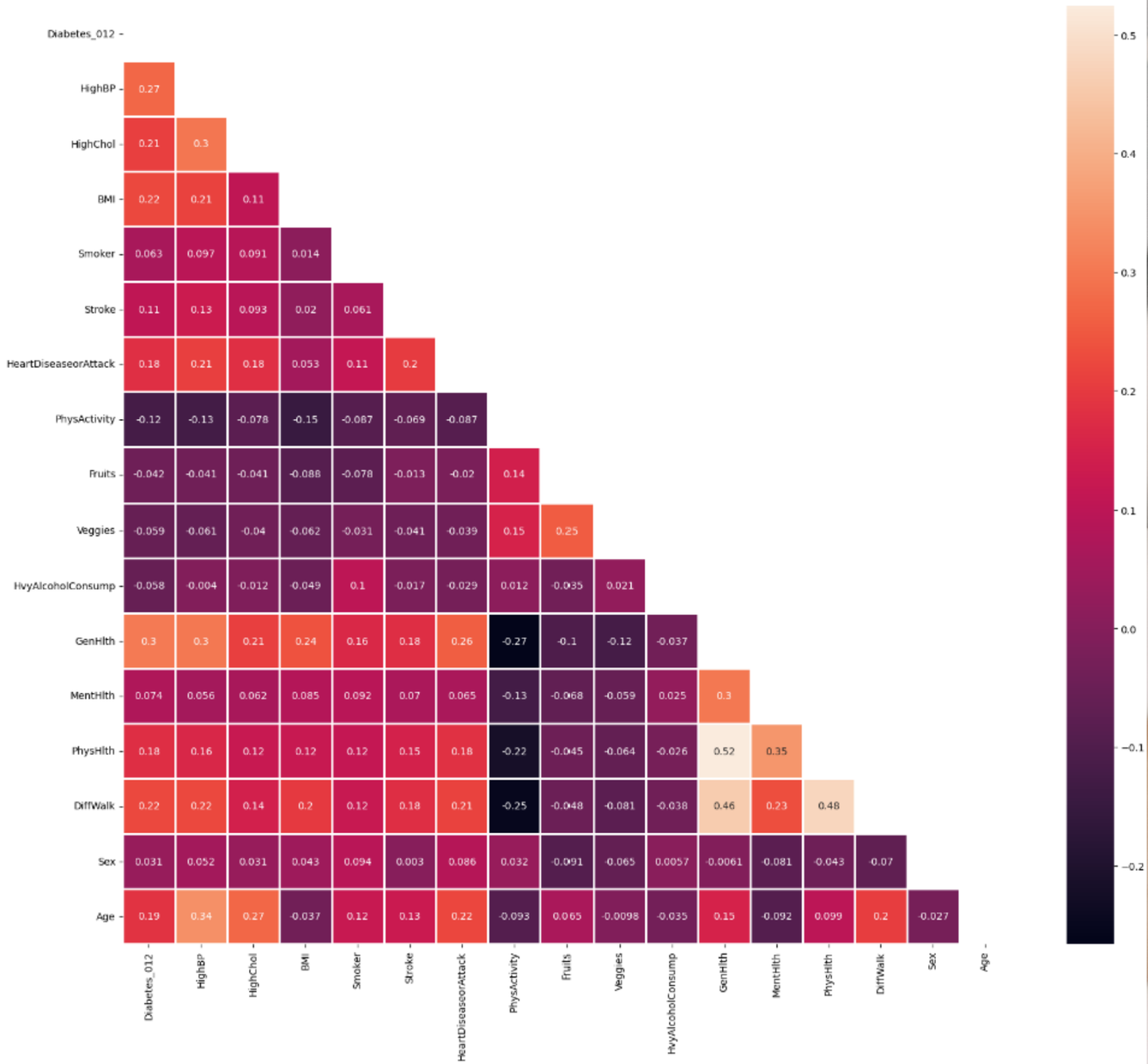
Finding the correlation between variables

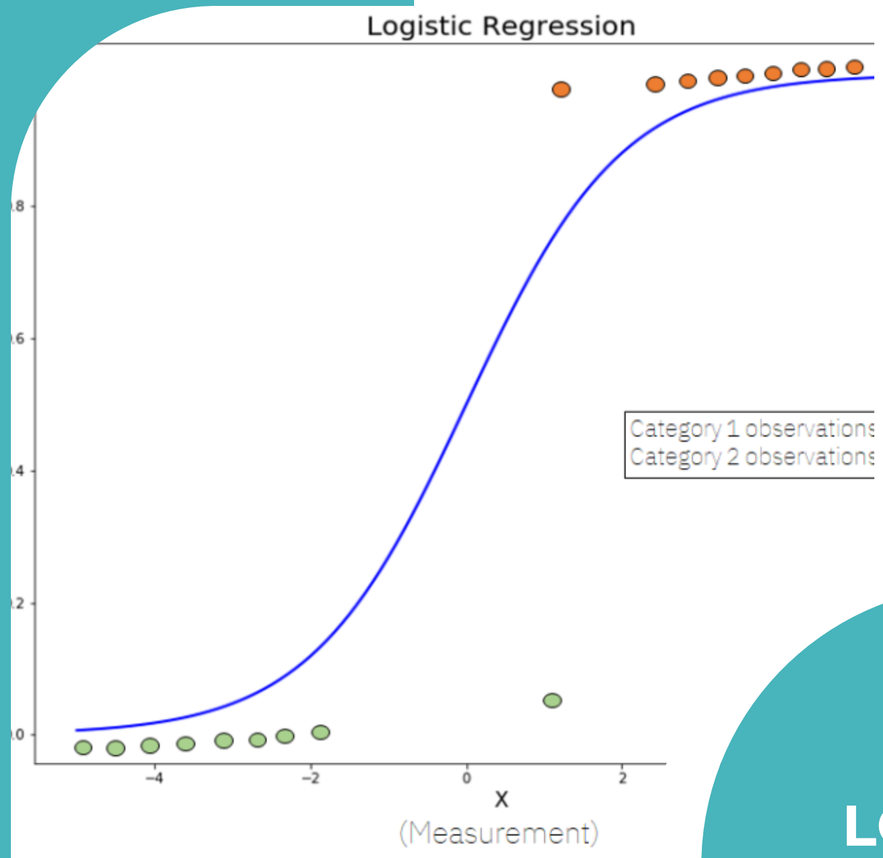
```
hig_corr = df2.corr()
hig_corr_features = hig_corr.index[abs(hig_corr["Diabetes_012"]) >= 0.2]
hig_corr_features
```

```
Index(['Diabetes_012', 'HighBP', 'HighChol', 'BMI', 'GenHlth', 'DiffWalk'],
```

```
plt.figure(figsize=(20, 17))
matrix = np.triu(df2.corr())
sns.heatmap(df2.corr(), annot=True, linewidth=.8, mask=matrix, cmap="rocket")
```

```
<Axes: >
```



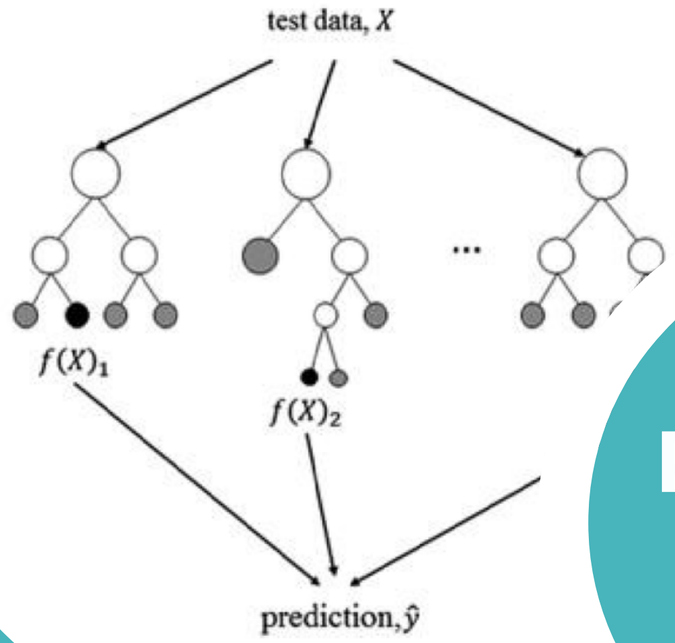


LOGISTIC REGRESSTION

```
#Creating a logistic regression calssifier
log_model = linear_model.LogisticRegression()

# Fitting Random Forest to the Training set
log_model.fit(X_train, y_train)

# Predicting the Test set results
y_pred = log_model.predict(X_test)

# Evaluate the model
accuracy= accuracy_score(y_test, y_pred)
print(f"Accuracy: {round(accuracy*100,2)} %", "\n")
```

Accuracy: 84.86 %

Testing



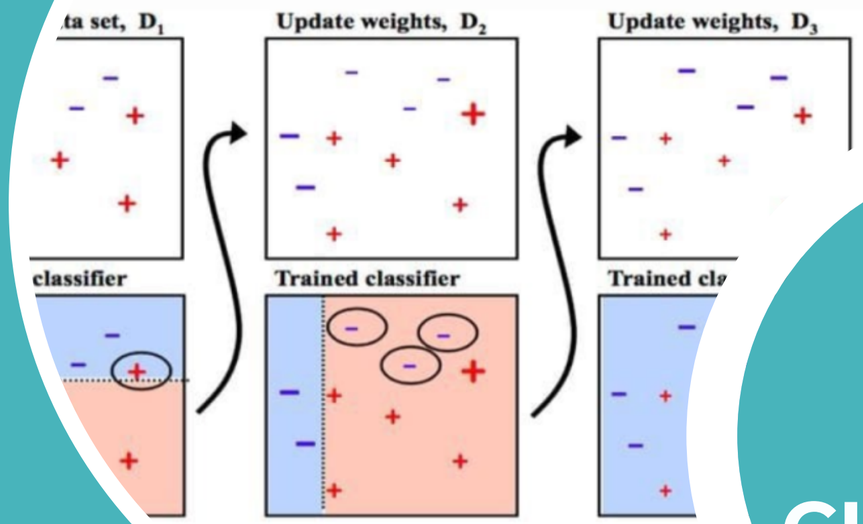
RANDOM FOREST

```python
# Creating a Random Forest Classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_

# Fitting Random Forest to the Training set
rf_classifier.fit(X_rf_train, y_rf_train)

# Predicting the Test set results
y_pred_rf = rf_classifier.predict(X_rf_test)

# Evaluate the model
accuracy_rf = accuracy_score(y_rf_test, y_pred_rf)

print("\nRandom Forest prediction results:", "\n")
print(f"Accuracy: {round(accuracy_rf*100,2)} %", "\n")
```

```
Random Forest prediction results:

Accuracy: 83.48 %
```

Accuracy: 83.48 %

...at is AdaBoost



ADA BOOST CLASSIFIER

```python
ada_model = AdaBoostClassifier(n_estimators=50, random_s

# Fitting Ada Boost to the Training set
ada_model.fit(X_train, y_train)

# Predicting the Test set results
y_pred_adaboost = ada_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred_adaboost)
print(f"Accuracy: {round(accuracy*100,2)} %", "\n")
```

```
Accuracy: 85.11 %
```

Accuracy: 85.11 %

GRADIENT BOOSTING

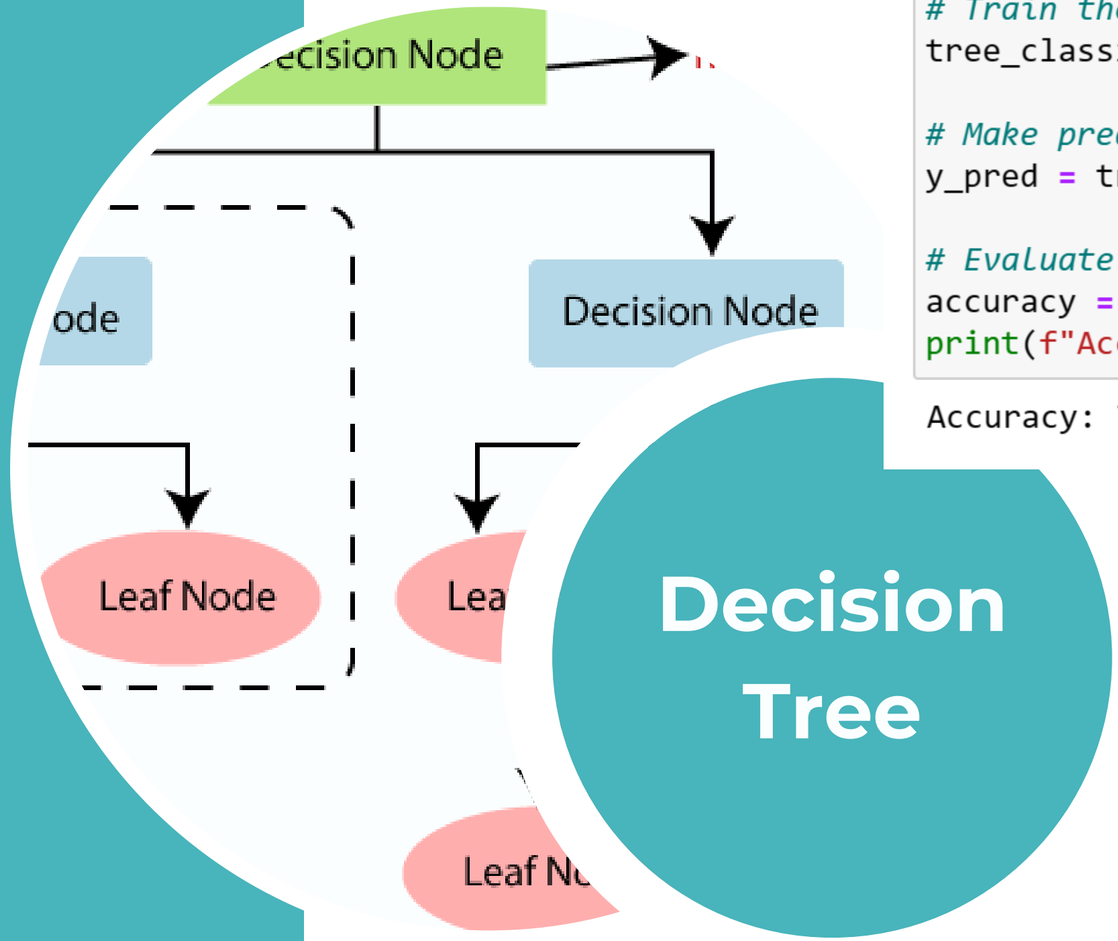
```
gbc_model = GradientBoostingClassifier(n_estimators=100,

# Fit the GBC model to the training data
gbc_model.fit(X_train, y_train)

# Predicting the Test set results
y_pred_gbm = gbc_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred_gbm)
print(f"Accuracy: {round(accuracy*100,2)} %", "\n")
```

Accuracy: 85.12 %

E. Decision Tree

```
# Create a DecisionTreeClassifier object
tree_classifier = DecisionTreeClassifier()

# Train the Decision Tree classifier
tree_classifier.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = tree_classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy*100:.2f}%")
```

```
Accuracy: 79.25%
```



Accuracy: 79.25%

Semi-supervised learning

```
X = df.drop('Diabetes_012', axis=1)
y = df['Diabetes_012']

# Just for demonstration, let's convert it to a semi-supervised setup by 'hiding' som
# Here, we arbitrarily mark 75% of the dataset as 'unlabeled' (-1)
mask = np.random.rand(len(y)) < 0.75
y[mask] = -1

# Splitting the dataset into labeled and unlabeled sets
X_labeled = X[y != -1]
y_labeled = y[y != -1]
X_unlabeled = X[y == -1]

# Confirm the distribution of classes
print(y_labeled.value_counts())

# Scaling the features (assuming X is not yet scaled for this subset)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Reducing the dataset size
X_small = X_scaled[:10000]  # Adjust the number based on available memory
y_small = y[:10000].copy()  # Make a copy to avoid warnings                    ing this array

# Simulate semi-supervised setup by 'hiding' some
mask = np.random.rand(len(y_small)) < 0.75
y_small[mask] = -1  # Marking 75% of the data
```

```
Diabetes_012
0.0    53611
2.0     8905
1.0     1141
Name: count, dtype: int64
```

Label Propagation

```
# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X_small, y_small, test_size=0.2, random_state=42)

# Apply semi-supervised learning: Label Propagation
# Initialize Label Propagation with a fixed random state
label_prop_model = LabelPropagation()

label_prop_model.fit(X_train, y_train)

# Predict on the test set
y_pred = label_prop_model.predict(X_test)

# Since we've used -1 for unlabeled, evaluate only on originally labeled data in y_test
mask_test = y_test != -1
print("Accuracy on test set:", accuracy_score(y_test[mask_test], y_pred[mask_test]))
print("Classification report on test set:\n", classification_report(y_test[mask_test], y_pred[mask_test]))
```

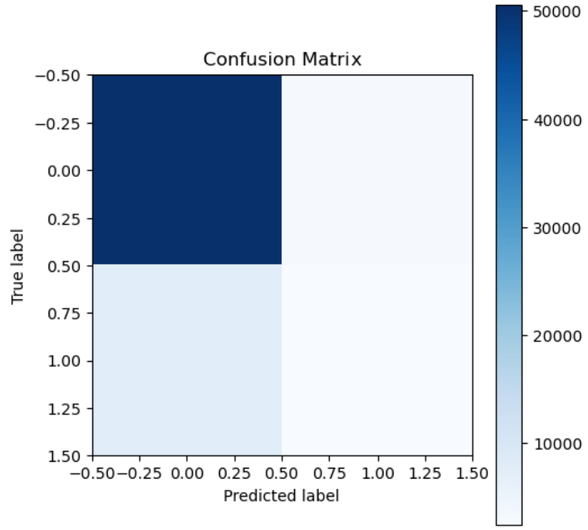
```
Accuracy on test set: 0.7818181818181819
Classification report on test set:
              precision    recall  f1-score   support

         0.0       0.85      0.89      0.87        90
         1.0       0.00      0.00      0.00         2
         2.0       0.43      0.33      0.38        18

    accuracy                           0.78       110
   macro avg       0.43      0.41      0.41       110
weighted avg       0.77      0.78      0.77       110
```

Data Visualization

Confusion matrix

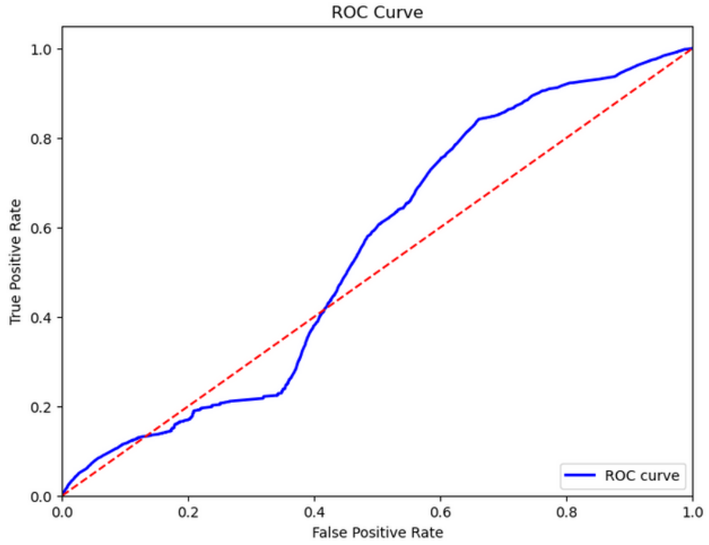
```
#data visualization
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, classification_report, confusion_matrix
from sklearn.metrics import precision_recall_curve

# Plot confusion matrix
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
cm = confusion_matrix(y_rf_test, rf_classifier.predict(X_rf_test))
plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
plt.colorbar()
plt.title('Confusion Matrix')
plt.ylabel('True label')
plt.xlabel('Predicted label')
```

28]: Text(0.5, 0, 'Predicted label')



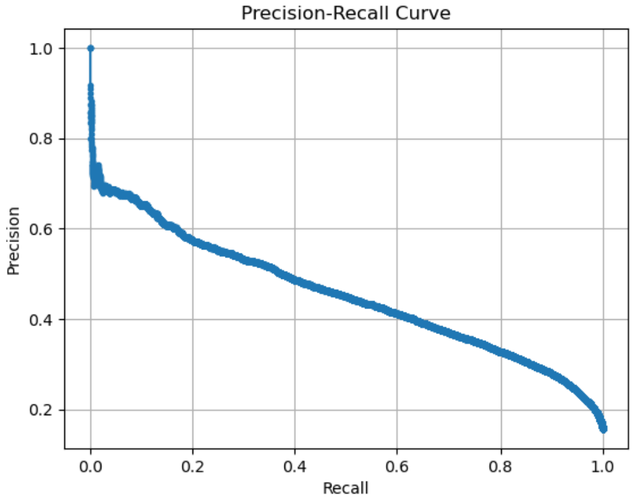
Roc Curve ¶

```
# Compute ROC curve
fpr, tpr, thresholds = roc_curve(y_test, gbc_model.predict_proba(X_rf_test)[:, 1])

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', lw=2, label='ROC curve')
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()
```

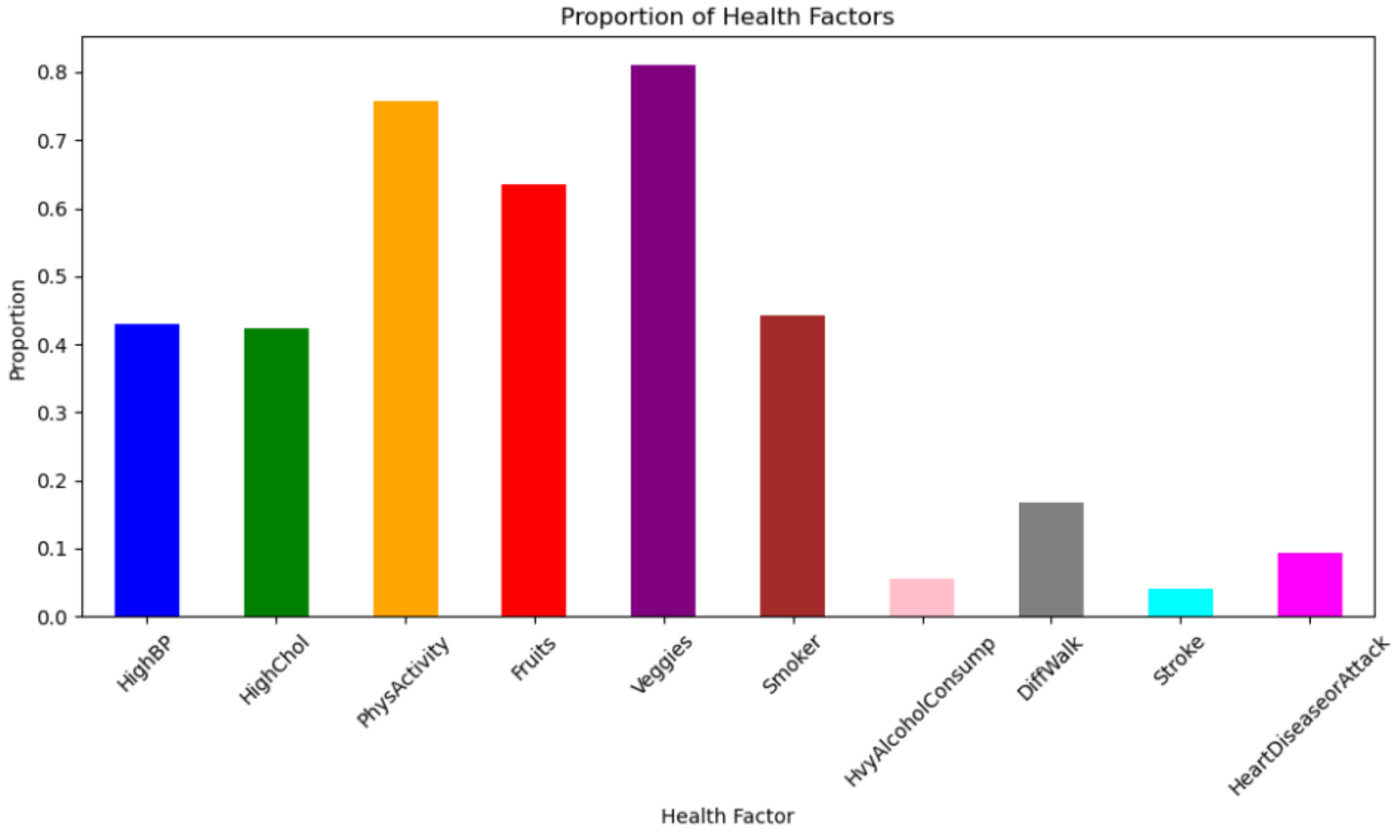


Precision-Recall Curve

```
# Compute precision-recall pairs for different probability thresholds
precisions, recalls, thresholds = precision_recall_curve(y_test, gbc_model.predict_proba(X_test)[:,

# Plot precision-recall curve
plt.plot(recalls, precisions, marker='.')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.title('Precision-Recall Curve')
plt.grid(True)
plt.show()
```



```
combinedBarChart = [
    'HighBP', 'HighChol', 'PhysActivity', 'Fruits', 'Veggies',
    'Smoker', 'HvyAlcoholConsump', 'DiffWalk', 'Stroke', 'HeartDiseaseorAttack'
]
df4 = df[combinedBarChart]

health_factors = df4.sum() / len(df4)

plt.figure(figsize=(10, 6))
health_factors.plot(kind='bar', color=['blue', 'green', 'orange', 'red', 'purple', 'brown', 'pink', 'gray', 'cyan', 'magenta'
plt.title('Proportion of Health Factors')
plt.xlabel('Health Factor')
plt.ylabel('Proportion')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Data Visualization

Diabetes Health Indicators

Utilization of Analysis Result

- Identify high-risk individuals for diabetes:

- Consider demographic factors and health-related features
- High BP, high cholesterol, high BMI, low general health increase risk

- Allocate healthcare resources efficiently:

- Intensive interventions for high-risk individuals
- Standard care for low-risk individuals

- Target interventions towards:

- Weight management
- Blood pressure control
- Smoking cessation

Diabetes Health Indicators

Conclusion

- The datasets with key figures of diabetes health indicators allow to spot high-risk individuals and innovate in the way available resources are used in healthcare management.

- By making use of these datasets the decisions are made in the right way and thus prevention and management of diabetes are enhanced, and public health outcomes become more of a benefit to the community as a whole.

REFERENCES

Diabetes Health Indicators Dataset

253,680 survey responses from cleaned BRFSS 2015 + balanced dataset

k kaggle.com

MAYO CLINIC

Diabetes - Symptoms and causes

Learn more about the different types of this blood sugar disorder, who's at risk and how each type can be treated.

Mayo Clinic

Q A

Thank You