Diabetes Prediction using Machine Learning

About Dataset

The Diabetes Prediction Dataset contains medical and demographic details like age, gender, BMI, hypertension, heart disease, smoking history, HbA1c, and blood glucose levels, along with diabetes status. It's ideal for building machine learning models to predict diabetes risk and supports healthcare analysis and research into contributing factors.

Features:

- gender – Categorical
- age – Numeric (8 months - 80 years)
- hypertension – Binary (0 or 1)
- heart_disease – Binary (0 or 1)
- smoking_history – Categorical (never, No Info, current)
- bmi (Body mass index) – Numeric
- HbA1c_level – Numeric
- blood_glucose_level – Numeric
- diabetes – Target Variable (0 = No, 1 = Yes)

In [2]:
```python
# IMPORT LIBRARIES
import pandas as pd
```

In [35]:
```python
# LOAD THE DATASET
data = pd.read_csv('diabetes_prediction_dataset.csv')
```

In [14]:
```python
data.head()
```

Out[14]:

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blo
0	Female	80.0	0	1	never	25.19	6.6	
1	Female	54.0	0	0	No Info	27.32	6.6	
2	Male	28.0	0	0	never	27.32	5.7	
3	Female	36.0	0	0	current	23.45	5.0	
4	Male	76.0	1	1	current	20.14	4.8	

These codes load the diabetes prediction dataset using the Pandas library. It displays the first few rows (head()), provides dataset structure details (info()), and shows its dimensions (shape). This helps in understanding the dataset's contents, data types, and overall size before starting analysis or modeling.

In [16]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 9 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   gender               100000 non-null  object 
 1   age                  100000 non-null  float64
 2   hypertension         100000 non-null  int64  
 3   heart_disease        100000 non-null  int64  
 4   smoking_history      100000 non-null  object 
 5   bmi                  100000 non-null  float64
 6   HbA1c_level          100000 non-null  float64
 7   blood_glucose_level  100000 non-null  int64  
 8   diabetes             100000 non-null  int64  
dtypes: float64(3), int64(4), object(2)
memory usage: 6.9+ MB
```

In [20]:
```
data.shape
```

Out[20]:
```
(100000, 9)
```

In [22]:
```
data.describe()
```

Out[22]:

	age	hypertension	heart_disease	bmi	HbA1c_level	bloo
count	100000.000000	100000.00000	100000.000000	100000.000000	100000.000000	
mean	41.885856	0.07485	0.039420	27.320767	5.527507	
std	22.516840	0.26315	0.194593	6.636783	1.070672	
min	0.080000	0.00000	0.000000	10.010000	3.500000	
25%	24.000000	0.00000	0.000000	23.630000	4.800000	
50%	43.000000	0.00000	0.000000	27.320000	5.800000	
75%	60.000000	0.00000	0.000000	29.580000	6.200000	
max	80.000000	1.00000	1.000000	95.690000	9.000000	

1. BMI (Body Mass Index):

The average BMI is around 27.3, which falls in the overweight category (normal is 18.5–24.9). Some values go as high as 95.7, which is extremely abnormal and likely outliers or data errors. High BMI is a known risk factor for diabetes

2. HbA1c Level:

The average HbA1c is about 5.5%, which is close to the prediabetic range (5.7%–6.4%). This suggests that many patients might be at risk of developing diabetes, even if they aren't diabetic yet.

3. Blood Glucose Level:

The average blood glucose is 138 mg/dL. Although it's just under the 140 mg/dL cutoff for diabetes diagnosis (post-meal), it still suggests elevated levels, which could point to a large number of prediabetic or undiagnosed diabetic cases.

Basic Descriptive Analysis

In [79]:
```
# IMPORT LIBRARIES
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Distribution of the target variable (diabetes)

- This will calculate the percentage of diabetic and non-diabetic patients in the dataset.
- It will help to check for class imbalance, which is crucial when building machine learning models.

In [54]:
```
diabetes_distribution = data['diabetes'].value_counts(normalize=True) * 100
diabetes_distribution
```

Out[54]:
```
diabetes
0    91.5
1     8.5
Name: proportion, dtype: float64
```

One class (non-diabetic) dominates and so the models may become biased.

Gender distribution

- This will calculate the percentage of each gender in the dataset.
- To understand the gender representation of the data.
- Gender may influence diabetes risk as Gender can influence diabetes risk due to hormonal and biological differences. For example, men store more harmful belly fat, while women may face risks like gestational diabetes or PCOS. Lifestyle and health habits also vary by gender, making it a useful factor in diabetes prediction.

In [63]:
```
gender_distribution = data['gender'].value_counts(normalize=True) * 100
gender_distribution
```

Out[63]:
```
gender
Female    58.552
Male      41.430
Other      0.018
Name: proportion, dtype: float64
```

Average age for diabetic vs non-diabetic

- It will find the average age of diabetic and non-diabetic individuals separately.
- Which will help identify if age is a potential risk factor or not.
- If diabetics tend to be older, age becomes an important feature in predictions.

In [66]:
```
average_age_by_diabetes = data.groupby('diabetes')['age'].mean()
average_age_by_diabetes
```

Out[66]:
```
diabetes
0    40.115187
1    60.946588
Name: age, dtype: float64
```

Insight: Diabetic patients tend to be significantly older.

Most common smoking history categories#-Why:

- Counts how many patients fall into each smoking history category.
- Smoking can be a health risk factor, including for diabetes.

In [70]:
```
smoking_distribution = data['smoking_history'].value_counts()
smoking_distribution
```

Out[70]:
```
smoking_history
No Info        35816
never          35095
former          9352
current         9286
not current     6447
ever            4004
Name: count, dtype: int64
```

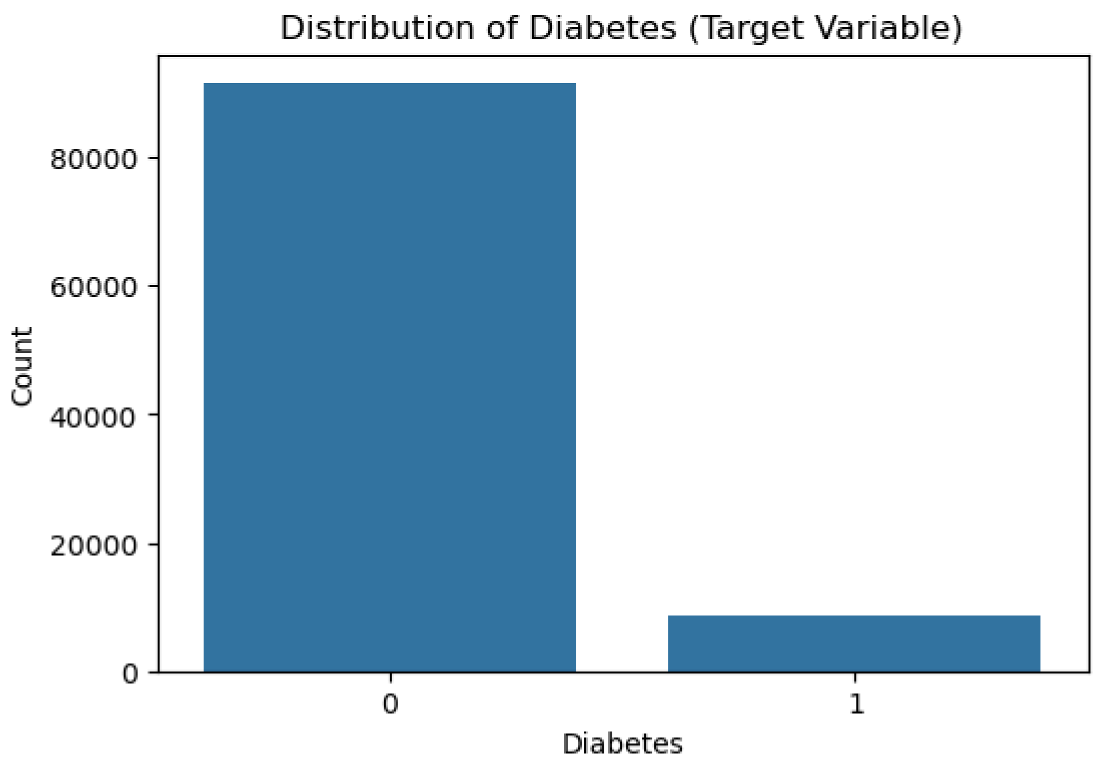
Check for missing values

In [74]:
```
missing_values = data.isnull().sum()
missing_values
```

Out[74]:
```
gender                 0
age                    0
hypertension           0
heart_disease          0
smoking_history        0
bmi                    0
HbA1c_level            0
blood_glucose_level    0
diabetes               0
dtype: int64
```

Plotting target distribution

In [83]:
```
plt.figure(figsize=(6, 4))
sns.countplot(data=data, x='diabetes')
plt.title("Distribution of Diabetes (Target Variable)")
plt.xlabel("Diabetes")
plt.ylabel("Count")
plt.show()
```



This bar chart shows a clear class imbalance in the target variable. Around 90% of individuals are non-diabetic (0), while only 10% are diabetic (1). This imbalance should be addressed during modeling (e.g., using resampling techniques), as it can lead to biased predictions.

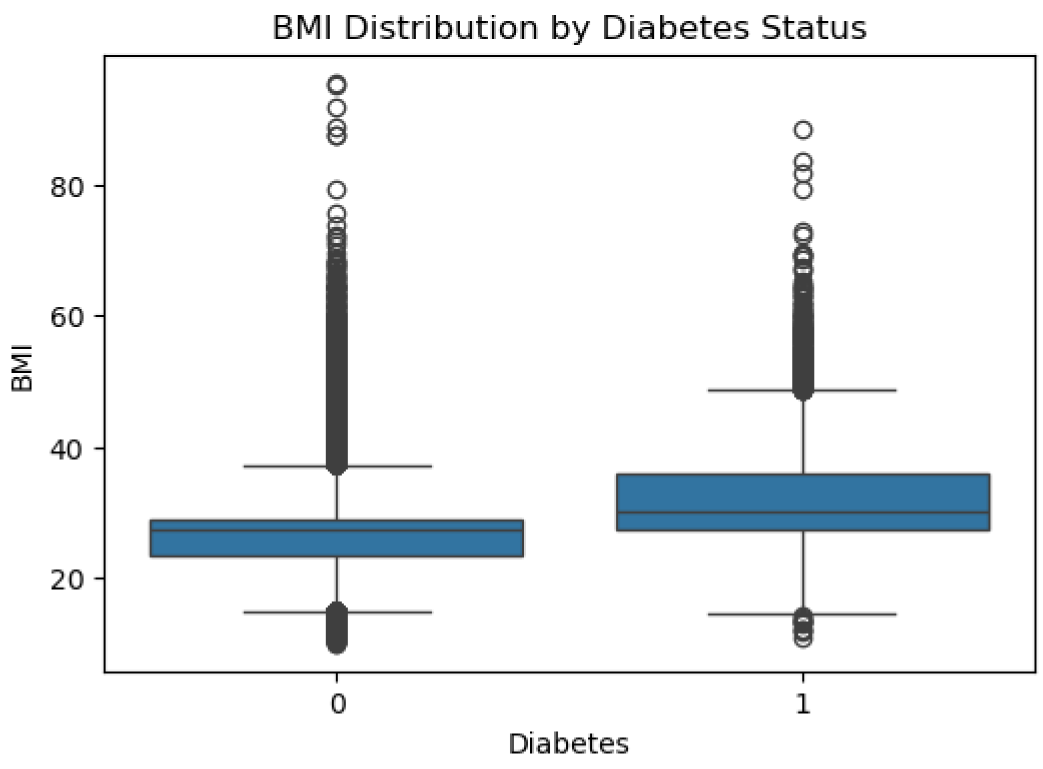
Intermediate Pattern Discovery

BMI differ between diabetic and non-diabetic patients

- Helps analyze if diabetic patients tend to have higher BMI, as obesity is a major risk factor for diabetes

In [92]:
```
bmi_by_diabetes = data.groupby('diabetes')['bmi'].describe()
```

In [94]:
```
plt.figure(figsize=(6, 4))
sns.boxplot(data=data, x='diabetes', y='bmi')
plt.title('BMI Distribution by Diabetes Status')
plt.xlabel('Diabetes')
plt.ylabel('BMI')
plt.show()
```



Insights :

- Diabetic patients (1) generally have a higher BMI than non-diabetic patients (0).
- The median BMI is noticeably higher in diabetics.
- Both groups show outliers with extremely high BMI values.
- This supports the link between higher BMI and diabetes risk.

Correlation between age and glucose levels

- To determine if older age is associated with higher glucose levels, which could indicate increased diabetes risk with age.

In [98]:
```
correlation_age_glucose = data['age'].corr(data['blood_glucose_level'])
correlation_age_glucose
```

Out[98]:
```
0.11067226757038073
```

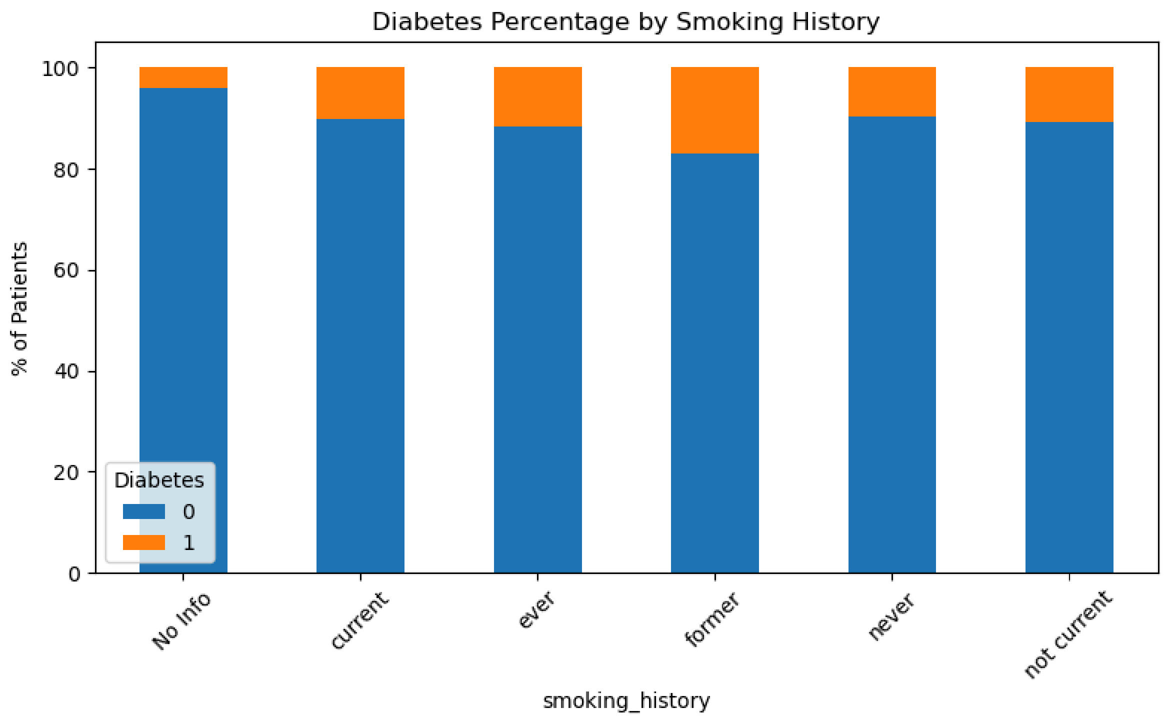
Insight : Slight positive correlation as age increases, glucose level tends to rise mildly.

Smoking habits vs diabetes rate

- To compare the diabetes rate within each smoking group (e.g., current smoker, never smoked, etc.), which helps assess risk factors.

In [105…
```
smoking_diabetes = pd.crosstab(data['smoking_history'], data['diabetes'], normal
```

In [107…
```
smoking_diabetes.plot(kind='bar', stacked=True, figsize=(8, 5))
plt.title('Diabetes Percentage by Smoking History')
plt.ylabel('% of Patients')
plt.legend(title='Diabetes')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Insights :

- Former smokers have the highest percentage of diabetic patients.
- Current and ever smokers also show a relatively higher diabetes rate than never or not current smokers.

Average HbA1c levels for diabetic vs non-diabetic

- HbA1c is a key indicator of long-term blood sugar levels.
- Diabetics are expected to have higher averages, which confirms that.

In [111…
```
hba1c_by_diabetes = data.groupby('diabetes')['HbA1c_level'].mean()
hba1c_by_diabetes
```

Out[111…
```
diabetes
0    5.396761
1    6.934953
Name: HbA1c_level, dtype: float64
```

Insight : Diabetics have significantly higher HbA1c, confirming medical expectations.

Relationship between hypertension/heart disease and diabetes

In [116…
```
hypertension_impact = pd.crosstab(data['hypertension'], data['diabetes'], normal
hypertension_impact
```

Out[116…

diabetes	0	1
hypertension		
0	93.069232	6.930768
1	72.104208	27.895792

Interpretation :

- Among patients without hypertension, only 6.93% are diabetic.
- Among those with hypertension, 27.90% are diabetic.

Conclusion: Patients with hypertension are 4x more likely to have diabetes.

In [118…

```
heart_disease_impact = pd.crosstab(data['heart_disease'], data['diabetes'], norm
heart_disease_impact
```

Out[118…

diabetes	0	1
heart_disease		
0	92.470174	7.529826
1	67.858955	32.141045

Interpretation :

- Among patients without heart disease, only 7.53% are diabetic.
- Among those with heart disease, 32.14% are diabetic.

Conclusion: Diabetes is over 4x more common in patients with heart disease.

Gender influence on diabetes

- To detect if one gender is more likely to be diabetic, which might inform targeted healthcare strategies.

In [124…

```
gender_impact = pd.crosstab(data['gender'], data['diabetes'], normalize='index')
gender_impact
```

Out[124…

diabetes	0	1
gender		
Female	92.381131	7.618869
Male	90.251026	9.748974
Other	100.000000	0.000000

Insight : Males have a slightly higher chance of diabetes.

Machine Learning section

- ### Random Forest Classifier:

Chosen for its ability to handle non-linear relationships, feature interactions, and provide high accuracy. It also offers insights into feature importance, making it ideal for complex medical data like this.

- ### Logistic Regression:

Selected as a baseline model due to its simplicity, interpretability, and strong performance on binary classification problems like predicting diabetes (yes/no). It helps us understand the direct impact of each feature on the target.

In [130…

```
# IMPORT LIBRARIES
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_sc
```

Encoding Categorical Variables

- We make a copy of the original dataframe to keep it intact. Then, we create encoders for the categorical columns gender and smoking_history.
- Machine learning models require numerical input. LabelEncoder converts text labels (like "Male", "Female", "Never", "Former", etc.) into integers so the model can understand them.

In [135…

```
df_encoded = data.copy()
le_gender = LabelEncoder()
le_smoking = LabelEncoder()
```

In [137…

```
df_encoded['gender'] = le_gender.fit_transform(df_encoded['gender'])
df_encoded['smoking_history'] = le_smoking.fit_transform(df_encoded['smoking_his
```

Define Features and Target

- X: All input features (predictors).
- y: The target variable (diabetes: 0 or 1).

In [140…

```
X = df_encoded.drop('diabetes', axis=1)
y = df_encoded['diabetes']
```

Scale Features

- Feature scaling ensures all variables contribute equally to the model, especially when features have different units/ranges (e.g., age vs glucose level).
- It improves model performance and convergence.

In [143…

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

Train-Test Split

- This helps evaluate how well the model generalizes to unseen data.
- 80% for training the model.
- 20% for testing its performance.

In [146…
```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
```

Random Forest Classification

Train a Random Forest Classifier

In [153…
```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

Out[153…
```
▾ RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Predictions and Evaluation

- Use the trained model to predict diabetes status on the test data.

In [156…
```
y_pred = model.predict(X_test)
```

In [174…
```
# Accuracy: Overall correctness.
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

Out[174…
```
0.97065
```

97.1% on the test set = excellent performance

In [210…
```
# Classification Report: Includes precision, recall, F1-score.
report = classification_report(y_test, y_pred)
report
```

Out[210…
```
'              precision    recall  f1-score   support\n\n           0       0.
96      0.99      0.98     18292\n           1       0.86      0.61      0.72
1708\n\n    accuracy                           0.96     20000\n   macro avg
0.91      0.80      0.85     20000\nweighted avg       0.96      0.96      0.96
20000\n'
```

In [216…
```
# Confusion Matrix: Shows true/false positives and negatives for deeper insight.
conf_matrix = confusion_matrix(y_test, y_pred)
conf_matrix
```

Out[216…
```
array([[18127,   165],
       [  661,  1047]], dtype=int64)
```

- 527 diabetic patients were misclassified as non-diabetic → important in a medical context!
- Suggests need for resampling (SMOTE) or threshold tuning if recall on diabetics is critical.

Feature Importance

- This tells us which features were most important in predicting diabetes (e.g., glucose level, age, BMI). It helps with interpretability and potential feature selection.

In [168…

```
feature_importance = pd.Series(model.feature_importances_, index=df_encoded.colu
feature_importance
```

Out[168…

```
HbA1c_level            0.380168
blood_glucose_level    0.333428
bmi                    0.121991
age                    0.103744
smoking_history        0.028206
hypertension           0.014374
heart_disease          0.011138
gender                 0.006951
dtype: float64
```

- Clinical indicators like HbA1c, Glucose, BMI, and Age are the strongest predictors — which aligns with medical science.

Logistic Regression

In [186…

```
# IMPORT LIBRARIES
from sklearn.linear_model import LogisticRegression
```

Train Logistic Regression Model

In [189…

```
log_reg = LogisticRegression(max_iter=1000, random_state=42)
log_reg.fit(X_train, y_train)
```

Out[189…

```
▾ LogisticRegression
LogisticRegression(max_iter=1000, random_state=42)
```

Make Predictions

In [192…

```
y_pred = log_reg.predict(X_test)
```

In [194…

```
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9587
```

In [196…

```
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Classification Report:
               precision    recall  f1-score   support

           0       0.96      0.99      0.98     18292
           1       0.86      0.61      0.72      1708

    accuracy                           0.96     20000
   macro avg       0.91      0.80      0.85     20000
weighted avg       0.96      0.96      0.96     20000
```

In [198…

```
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Confusion Matrix:
 [[18127   165]
 [  661  1047]]
```

- 661 diabetics were missed by the model. In medical cases, false negatives can be dangerous, so improving recall for Class 1 (e.g., using oversampling, adjusting thresholds) is essential.