**Introduction:**

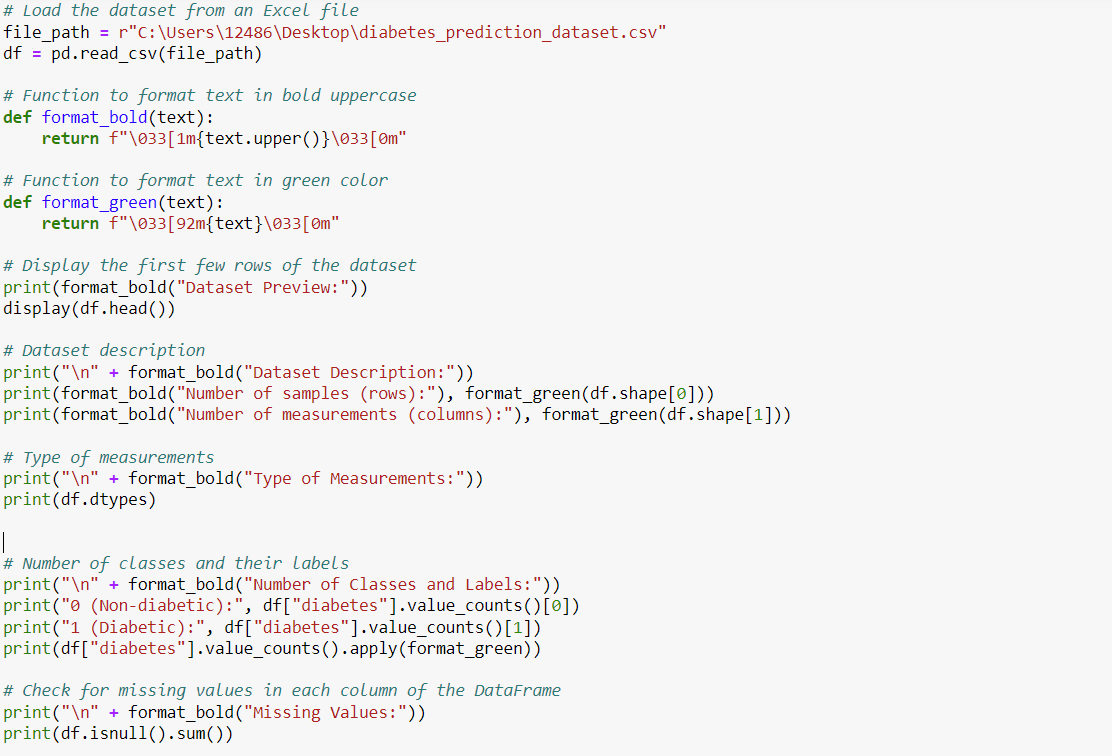
Start with an introduction of your project. This introduction should  
introduce (1) the problem you want to solve.

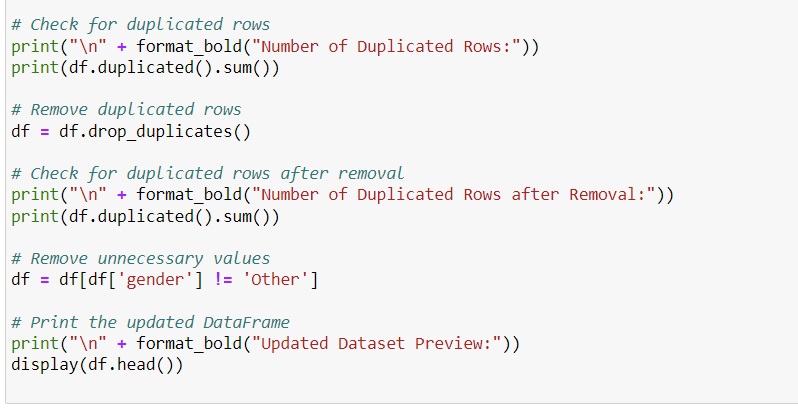
**Dataset descriptions**:

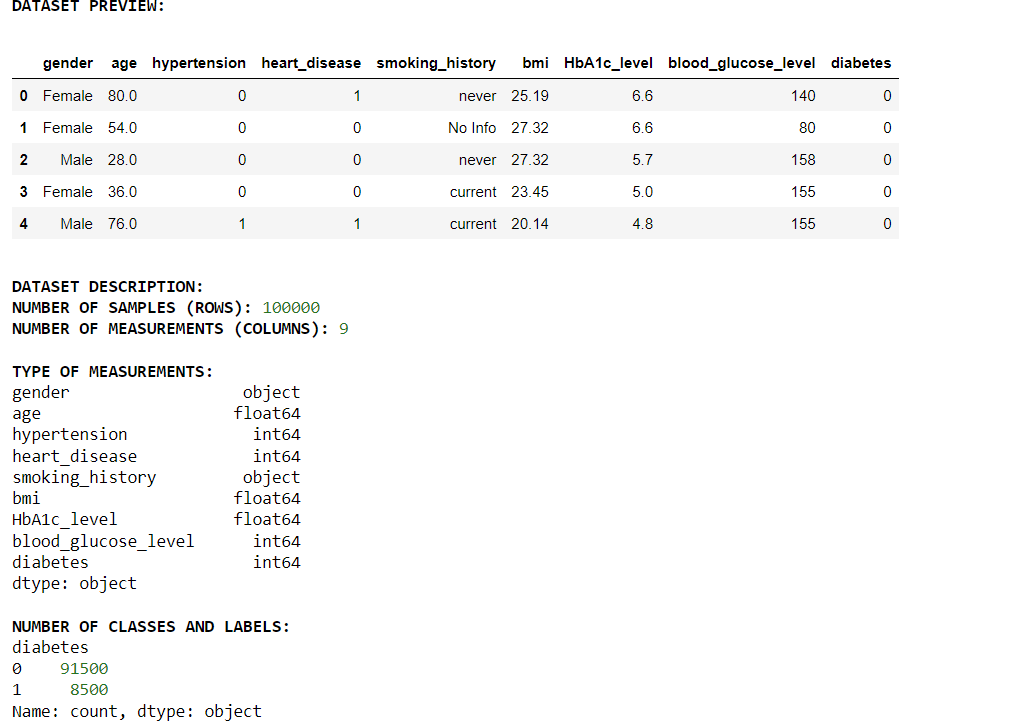
Size:

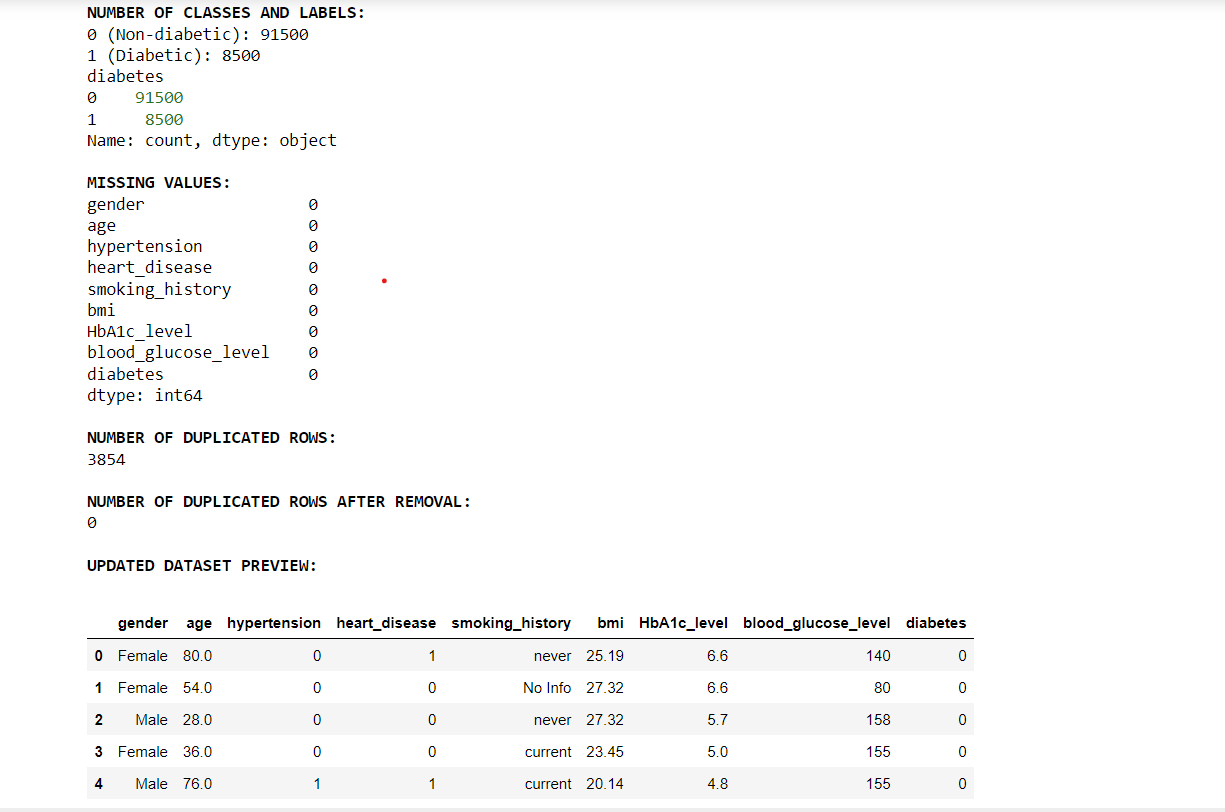
The number of measurements:

The type of the measurements:

The number of classes and their labels: 



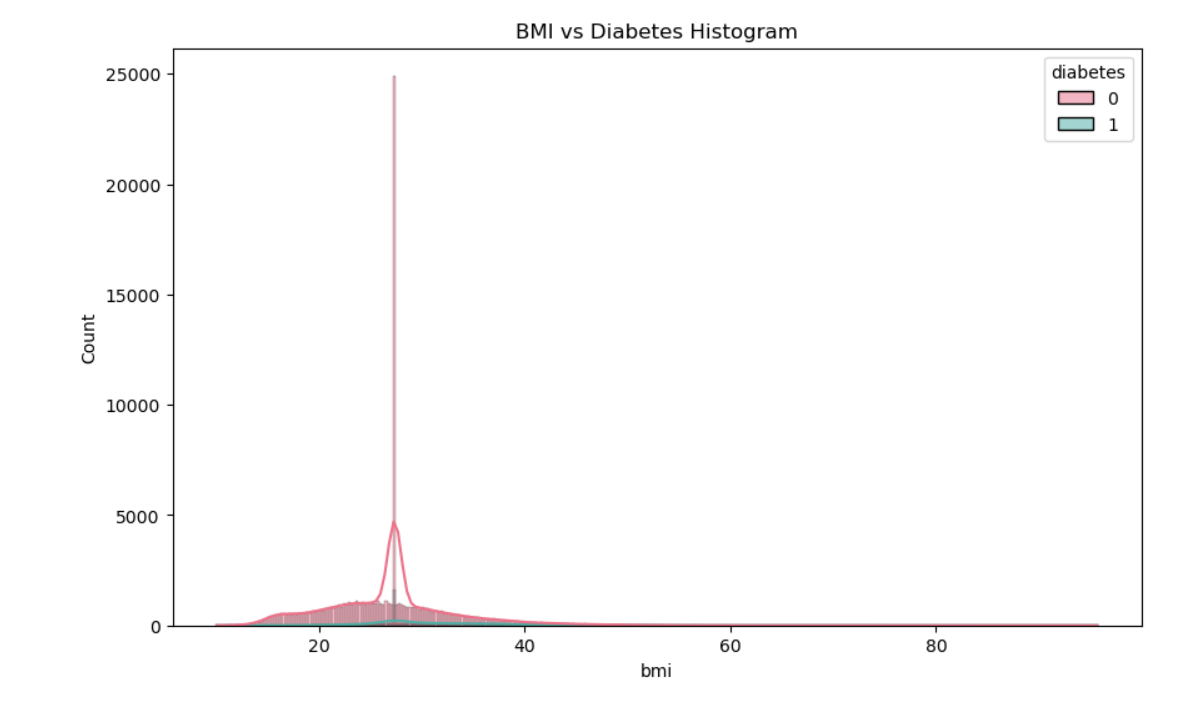


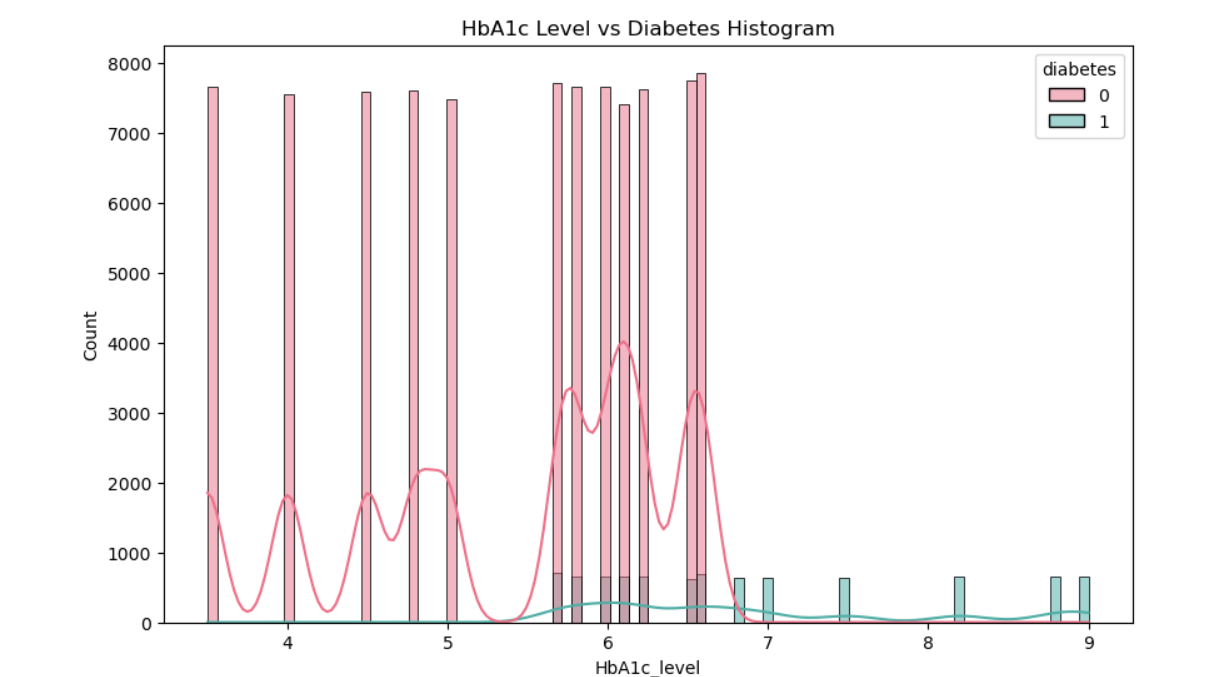


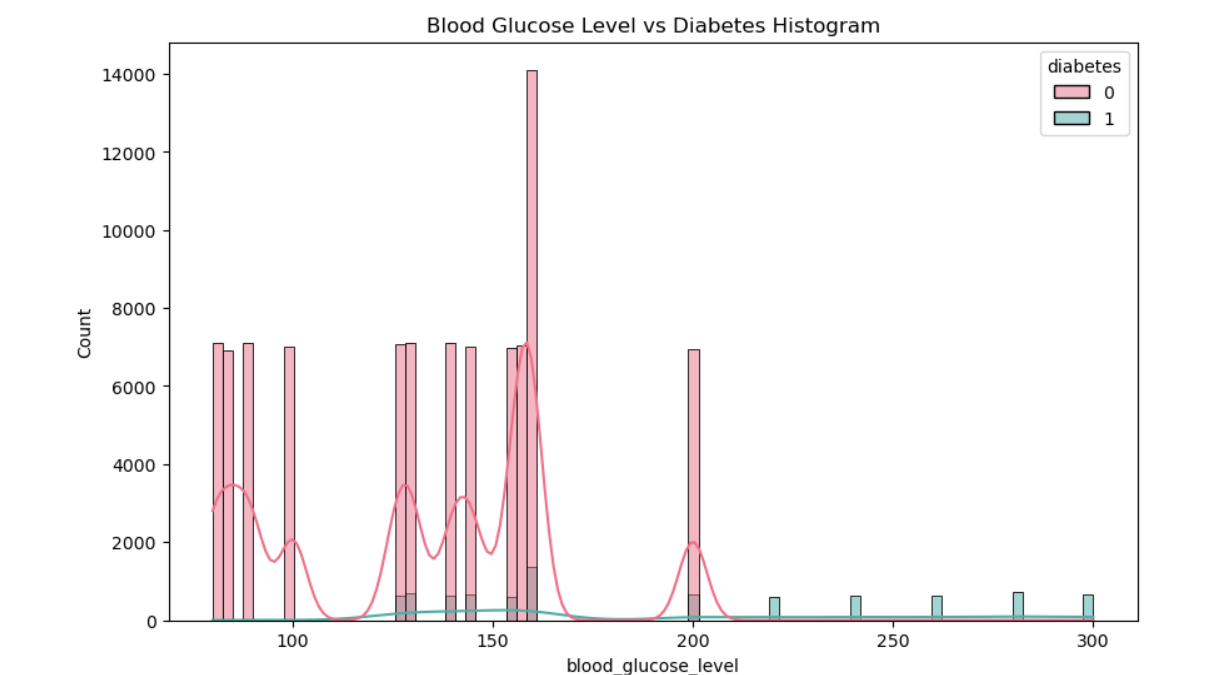
**Load the data and discover & visualize it to get insights:**

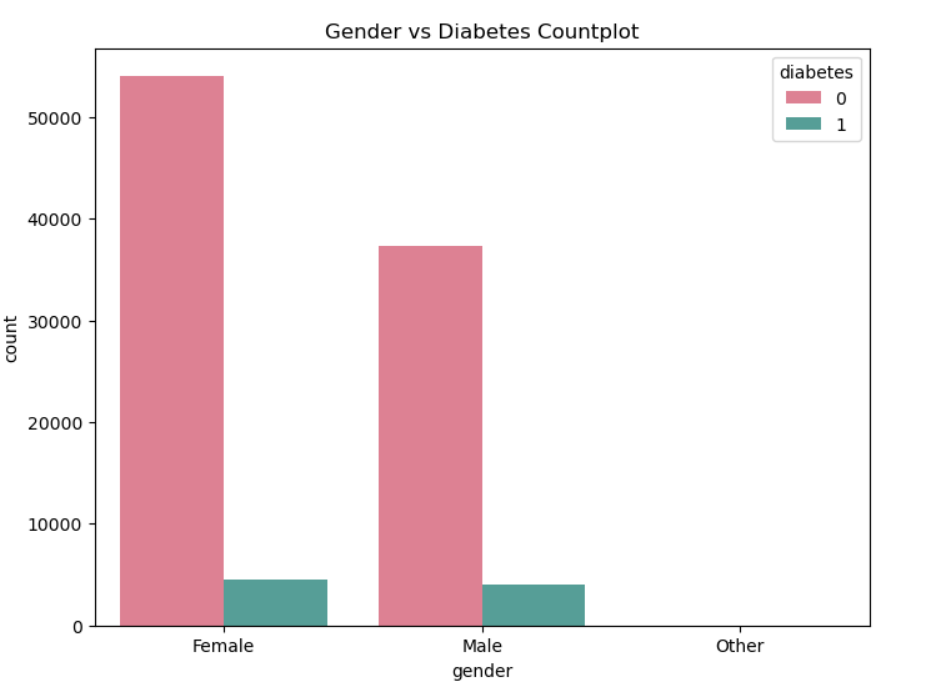
****

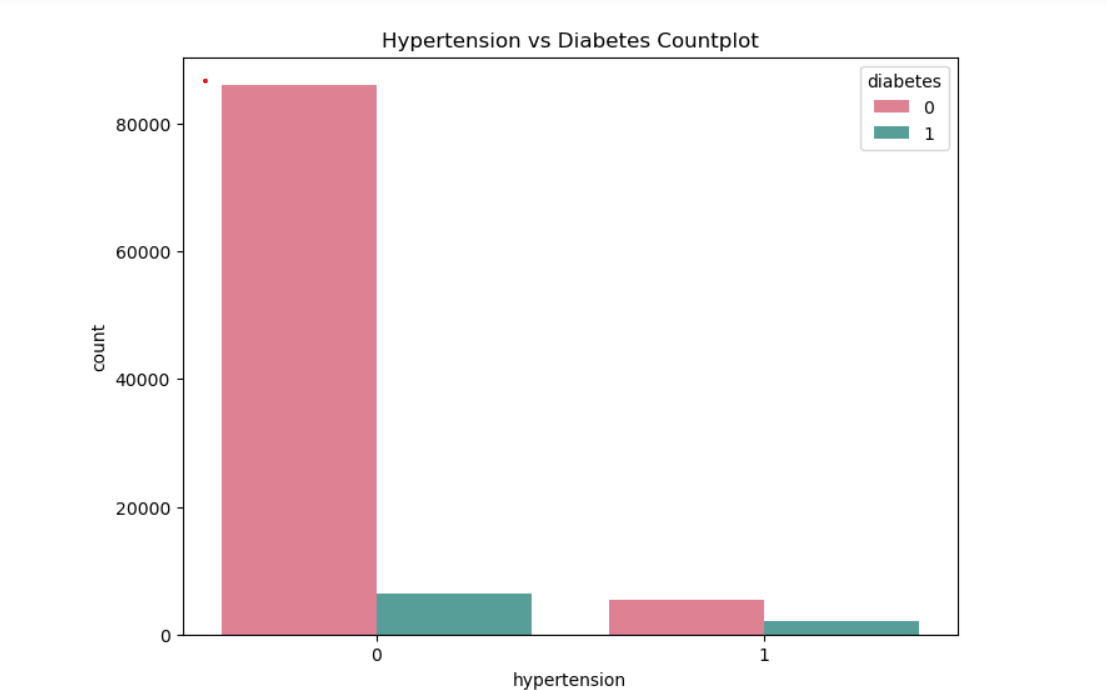
****

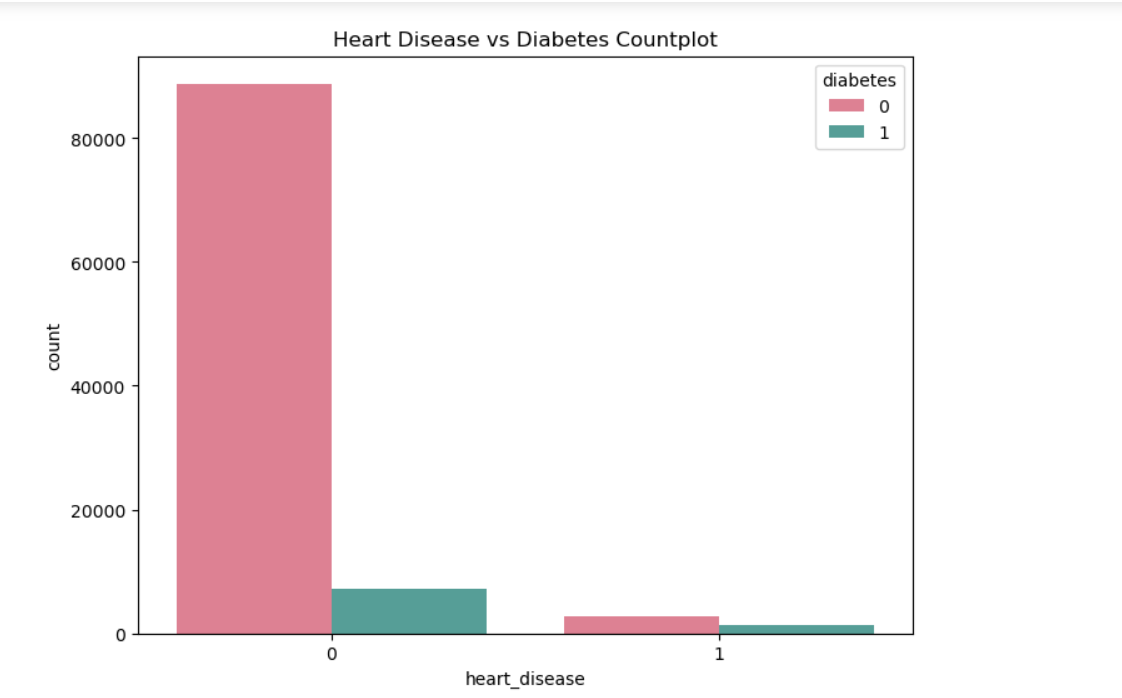
****

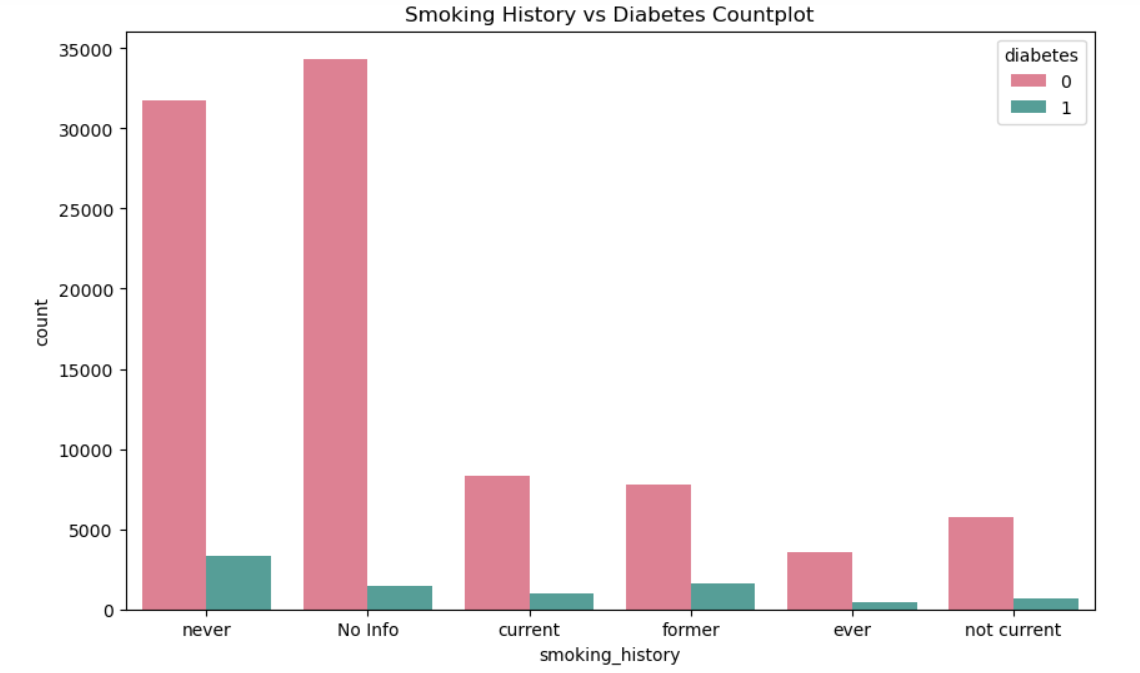
****

****

****

****

****

****

**Prepare the dataset:**

**Handling Text and Categorical Variables:**

Encode categorical variables: Convert categorical variables into numerical representations using techniques like one-hot encoding or label encoding.

**Cleaning the Data:**

Handle missing values: Impute missing values using strategies such as mean, median, or mode imputation, or drop rows/columns with missing values depending on the context.

**Handle outliers:** Detect and handle outliers by either removing them or transforming them using techniques such as winsorization or robust scaling.

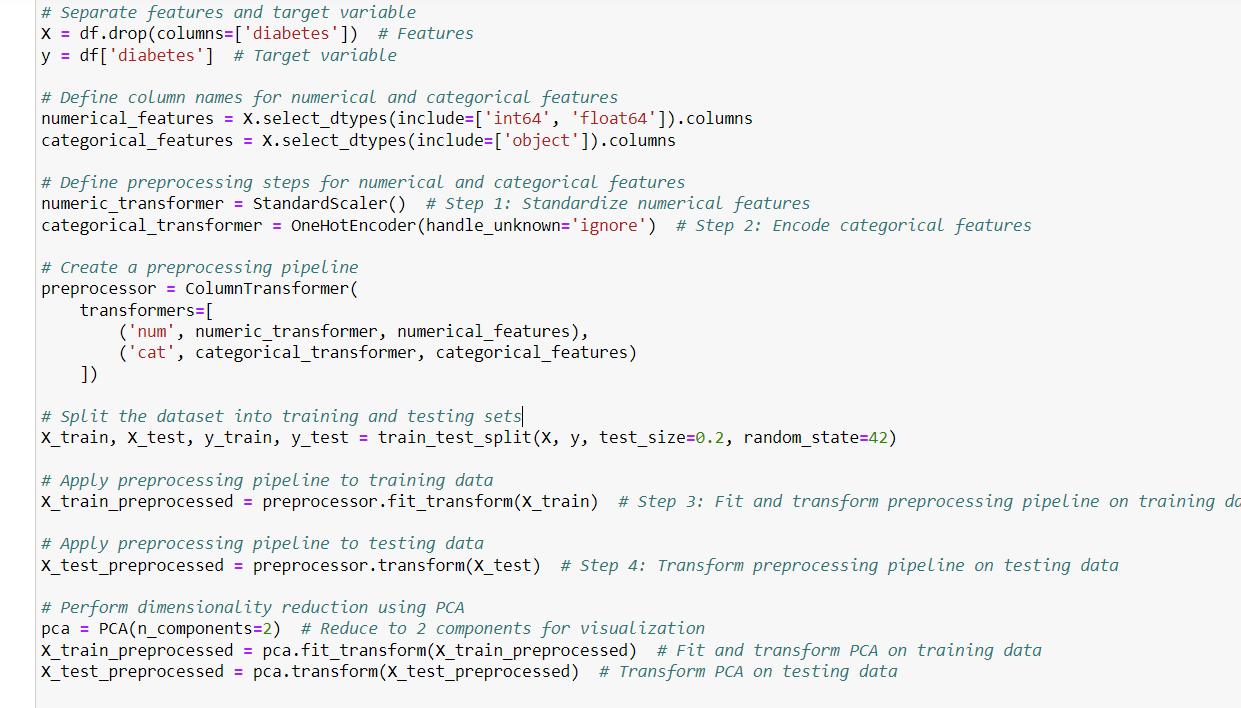
**Data Standardization:**

Standardize numerical features: Scale numerical features to have a mean of 0 and a standard deviation of 1 using techniques like z-score standardization or Min-Max scaling. This step ensures that all variables are on the same scale, which is important for K-NN.

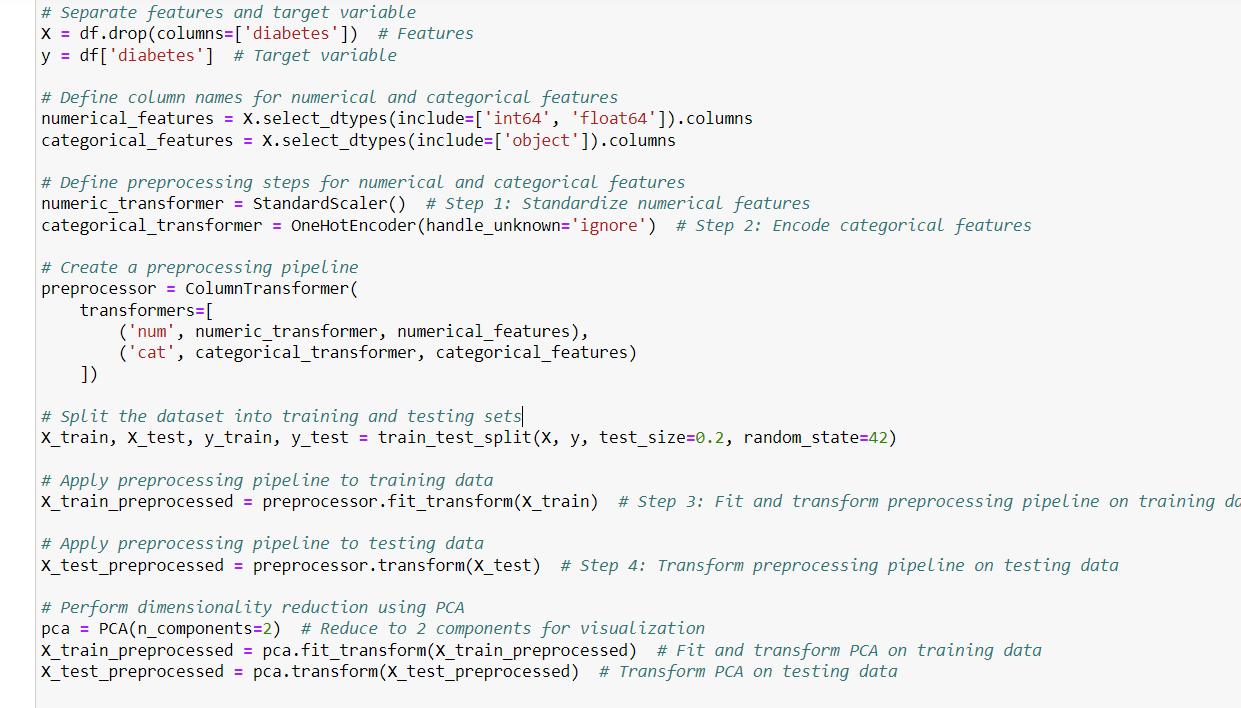
**Normalize numerical features:** Scale numerical features to a range between 0 and 1 using Min-Max scaling.

**Dimension Reduction:**

If the dataset has high-dimensional features, consider dimensionality reduction techniques such as Principal Component Analysis (PCA) to reduce the number of features while preserving most of the variance in the data.



**Data partitioning:**



In this code:

* X contains all the features except for the target variable 'diabetes'.
* y contains only the target variable 'diabetes'.
* The train\_test\_split function is used to split X and y into training and testing sets.
* The parameter test\_size=0.2 specifies that 20% of the data will be used for testing, while the remaining 80% will be used for training. random\_state=42 ensures reproducibility of the split.
* Now you have X\_train, X\_test, y\_train, and y\_test containing the respective sets for training and testing your machine learning model.

5.**Choosing three different values of K:**

* **K=2**
* **K= 7**
* **K=10**

**Reasons for choosing the different values of K:**

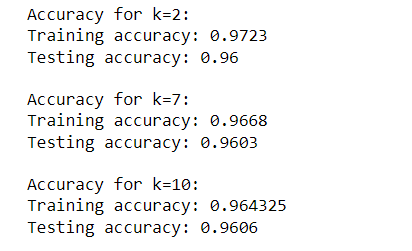
k=2:

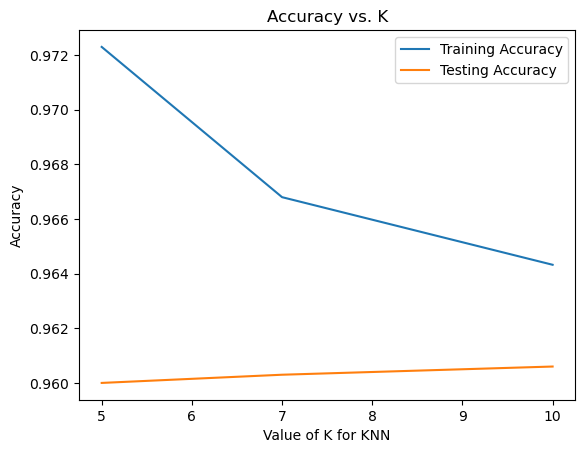
Choosing 𝑘=2 is a common starting point for clustering tasks, especially when the data might naturally form two distinct groups. In our dataset, there might be clear distinctions between individuals with and without diabetes. Therefore, starting with 𝑘=2. It allows us to explore whether there are indeed two main clusters in the data.

K=7

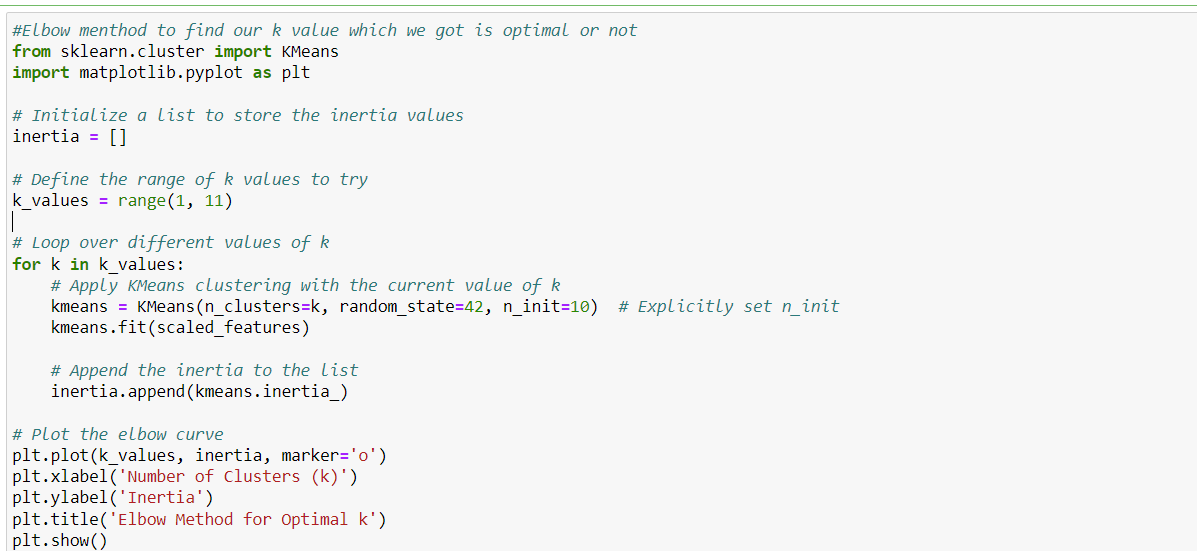
K=10

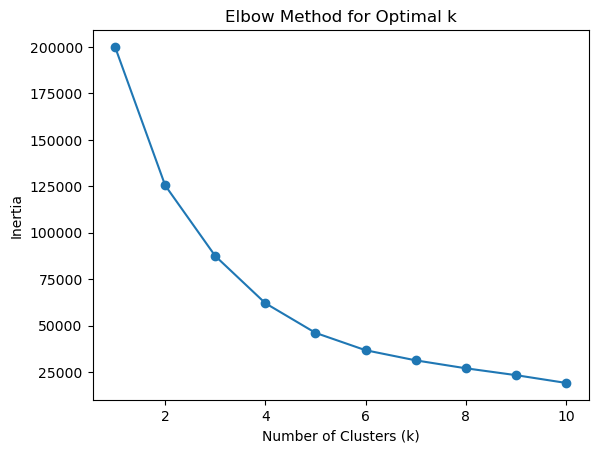
```````





**The best value k=10:**

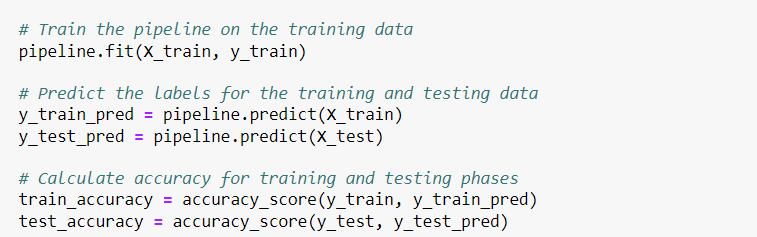


  
The elbow method is a technique used to determine the optimal number of clusters (𝑘k) in a KMeans clustering algorithm. It is based on the concept that as the number of clusters increases, the within-cluster sum of squared distances (inertia) decreases. However, adding more clusters will eventually lead to diminishing returns in terms of explaining the variance in the data. The elbow method helps identify the point where this decrease in inertia slows down, indicating that adding more clusters does not significantly improve the quality of clustering.

The term "inertia" in the context of KMeans clustering refers to the within-cluster sum of squared distances to the centroid. In other words, it measures how tightly the clusters are packed around their centroids.

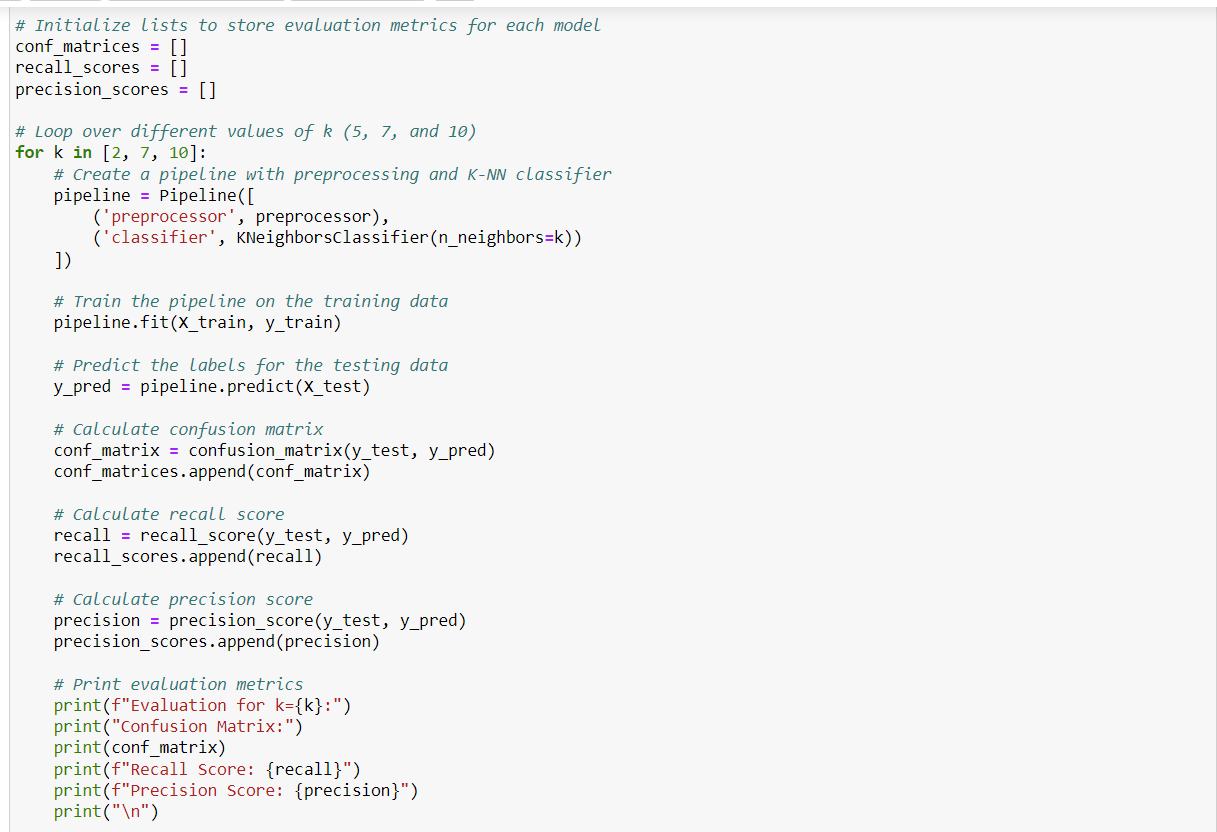
When we perform KMeans clustering, the algorithm aims to minimize the inertia. It does this by iteratively assigning data points to the nearest centroid and updating the centroids to minimize the total squared distance of each point to its assigned centroid.

**Training Phase** &**Testing Phase:**

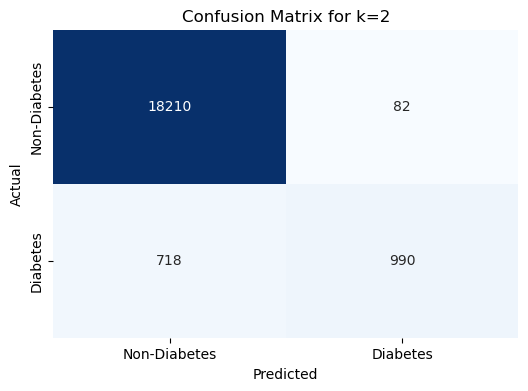


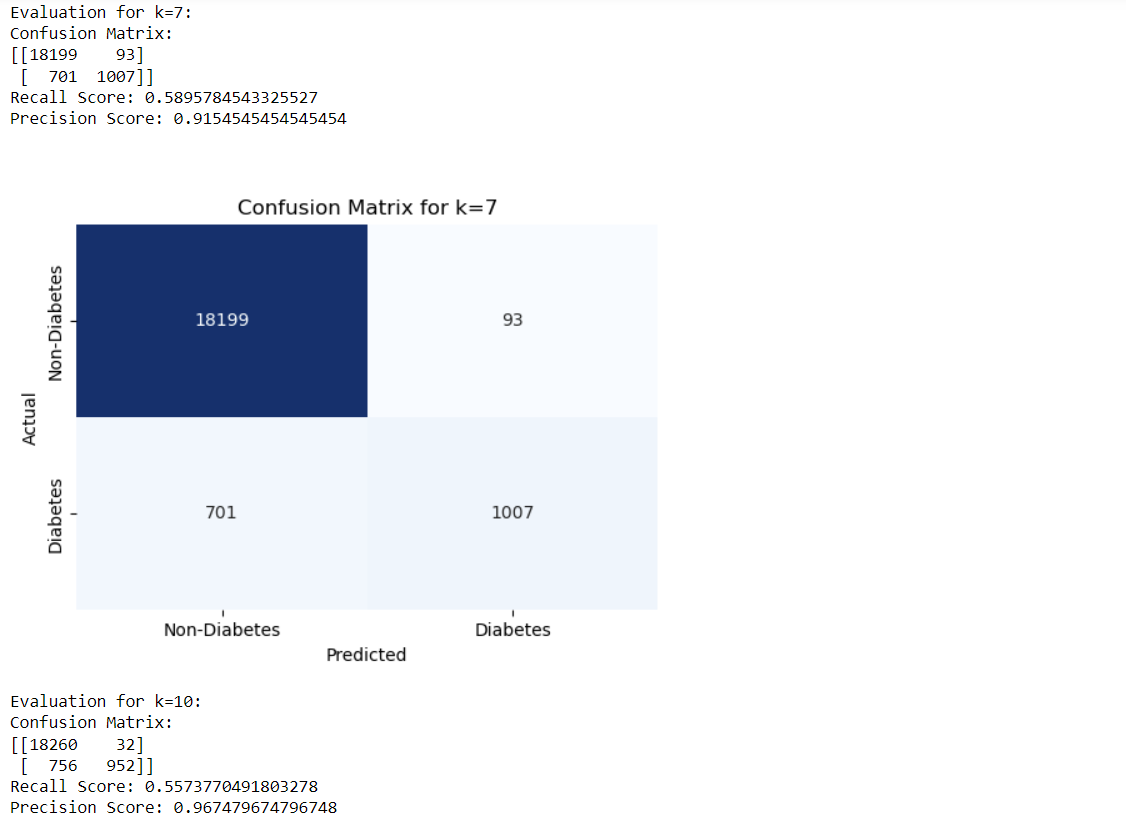
In the testing phase, the trained K-NN model is evaluated on a separate dataset called the testing dataset. The testing dataset contains instances that were not used during the training phase and serves as an independent measure of the model's performance. The accuracy of the model is assessed by comparing its predictions on the testing dataset with the ground truth labels or values. The accuracy metric provides insights into how well the model generalizes to unseen data. By comparing the accuracy between the training and testing phases for different values of 𝑘k, we can determine the optimal 𝑘k value that balances model complexity and performance.

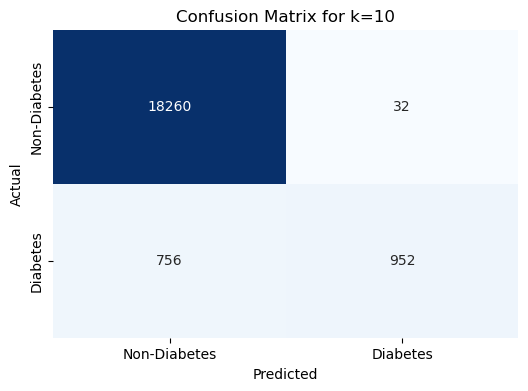
**choose the best model you found based on the results from  
the evaluation phase. Think of any improvement that can be made to get better results**

****

****



****



**Discuss your final results and conclusion about the model**

.