**✅ Cell 1**

python

CopyEdit

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

* **Problem Step**: *Import required Python libraries.*
* **Definitions**:
  + pandas: For handling and manipulating structured data (DataFrames).
  + numpy: For numerical operations, especially on arrays and matrices.
  + matplotlib.pyplot: A library for making visualizations like histograms and line charts.
  + seaborn: Built on matplotlib; it simplifies complex visualizations and provides better aesthetics.
* **Why Used**:  
  These libraries are essential for any data science task—handling data (pandas), performing numerical operations (numpy), and visualizing patterns or distributions (matplotlib and seaborn).

**✅ Cell 2**

python

CopyEdit

df = pd.read\_csv("diabetes.csv")

df

* **Problem Step**: *Load dataset.*
* **Definition**:
  + pd.read\_csv(): Reads a CSV file into a pandas DataFrame.
* **Why Used**:  
  To load the diabetes dataset into memory so it can be explored and analyzed further.

**✅ Cell 3**

python

CopyEdit

df.info()

* **Problem Step**: *Understand data structure and check for missing values.*
* **Definition**:
  + .info(): Displays the number of non-null entries, column names, data types, and memory usage.
* **Why Used**:  
  To quickly inspect the dataset’s structure and detect missing or malformed data.

**✅ Cell 4**

python

CopyEdit

df.describe()

* **Problem Step**: *Get statistical summary of dataset.*
* **Definition**:
  + .describe(): Provides descriptive statistics (mean, std dev, min, max, etc.) for numerical columns.
* **Why Used**:  
  To understand the distribution and central tendency of features, useful for normalization and anomaly detection.

**✅ Cell 5**

python

CopyEdit

df.isnull().sum()

* **Problem Step**: *Check for null/missing values in each column.*
* **Definition**:
  + isnull(): Returns a DataFrame with True for null values.
  + sum(): Adds up the True values, giving count of nulls per column.
* **Why Used**:  
  Ensures the dataset has no missing values before model building.

**✅ Cell 6**

python

CopyEdit

df.hist(bins=10, figsize=(10,10))

plt.show()

* **Problem Step**: *Visualize distribution of each feature.*
* **Definition**:
  + .hist(): Creates histograms for each numeric column.
  + bins=10: Divides values into 10 bins (intervals).
  + figsize: Sets size of the plot.
  + plt.show(): Renders the plot.
* **Why Used**:  
  To identify skewness or patterns in feature distributions—useful for feature engineering or transformation.

**✅ Cell 7**

python

CopyEdit

sns.boxplot(data=df)

* **Problem Step**: *Visualize outliers.*
* **Definition**:
  + sns.boxplot(): Creates box plots for each feature showing median, quartiles, and outliers.
* **Why Used**:  
  To identify which features have outliers. This helps in deciding whether to apply transformations or clipping.

**✅ Cell 8**

python

CopyEdit

from sklearn.preprocessing import StandardScaler

* **Problem Step**: *Prepare for normalization.*
* **Definition**:
  + StandardScaler: A preprocessing tool from sklearn that standardizes features by removing the mean and scaling to unit variance.
* **Why Used**:  
  Many ML algorithms (like k-NN, SVM) perform better when data is normalized.

**✅ Cell 9**

python

CopyEdit

scaler = StandardScaler()

X = df.drop(columns=["Outcome"])

y = df["Outcome"]

X\_scaled = scaler.fit\_transform(X)

* **Problem Step**: *Normalize feature set (excluding target).*
* **Definitions**:
  + drop(columns=["Outcome"]): Removes the target column from features.
  + fit\_transform(): Calculates mean and std dev, then transforms the features.
* **Why Used**:  
  Standardizing ensures all features contribute equally to the model—especially important for distance-based models.

**✅ Cell 10**

python

CopyEdit

from sklearn.model\_selection import train\_test\_split

* **Problem Step**: *Prepare for training/testing.*
* **Definition**:
  + train\_test\_split: Function that splits dataset into training and testing sets.
* **Why Used**:  
  To evaluate model performance on unseen data and avoid overfitting.

**✅ Cell 11**

python

CopyEdit

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

* **Problem Step**: *Split dataset.*
* **Definitions**:
  + test\_size=0.2: 20% of data goes to test set.
  + random\_state: Seed for reproducibility.
* **Why Used**:  
  Ensures reproducible and valid evaluation of the machine learning model.

**✅ Cell 12**

python

CopyEdit

from sklearn.linear\_model import LogisticRegression

* **Problem Step**: *Import classification model.*
* **Definition**:
  + LogisticRegression: A linear model used for binary classification.
* **Why Used**:  
  Suitable for the diabetes prediction task, as the target variable ("Outcome") is binary.

**✅ Cell 13**

python

CopyEdit

model = LogisticRegression()

model.fit(X\_train, y\_train)

* **Problem Step**: *Train the model.*
* **Definitions**:
  + fit(): Trains the logistic regression model using training data.
* **Why Used**:  
  To build a model that learns patterns in data to predict diabetes outcome.

**✅ Cell 14**

python

CopyEdit

y\_pred = model.predict(X\_test)

* **Problem Step**: *Make predictions.*
* **Definition**:
  + predict(): Generates predicted labels for the test data.
* **Why Used**:  
  To evaluate the model’s performance on unseen data.

**✅ Cell 15**

python

CopyEdit

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

* **Problem Step**: *Import evaluation metrics.*
* **Definitions**:
  + accuracy\_score: Measures percentage of correct predictions.
  + classification\_report: Gives precision, recall, f1-score.
  + confusion\_matrix: Summarizes prediction results with true vs predicted values.
* **Why Used**:  
  To assess how well the model is performing in various metrics.

**✅ Cell 16**

python

CopyEdit

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print("Classification Report:\n", classification\_report(y\_test, y\_pred))

print("Confusion Matrix:\n", confusion\_matrix(y\_test, y\_pred))

* **Problem Step**: *Display model evaluation.*
* **Definitions**:
  + print(): Displays output in the console.
* **Why Used**:  
  Provides a quantitative summary of model effectiveness, helps interpret strengths and weaknesses.