Logistic Regression:

The **`pima-indians-diabetes.csv`** file contains medical records for female patients of Pima Indian heritage. This dataset is often used in health-related machine learning projects to analyze and predict the likelihood of diabetes.
Here's an explanation of the typical columns you might find in this file:

- **Pregnancies** → Number of times the patient has been pregnant. This is a marker for maternal history and can correlate with diabetes risk.
- **Glucose** → Plasma glucose concentration a 2-hour test in an oral glucose tolerance test. Higher levels of glucose are a strong indicator of diabetes.
- **BloodPressure** → Diastolic blood pressure (mm Hg). It's a common measure used to assess cardiovascular health.
- **SkinThickness** → Triceps skinfold thickness (mm). This measure is used to estimate body fat, which is relevant to diabetes risk.
- **Insulin** → 2-hour serum insulin (mu U/ml). High insulin levels can indicate insulin resistance and a higher risk of diabetes.
- **BMI** → Body mass index (weight in kg/(height in m)^2). BMI is used to assess whether a person has healthy body weight for a given height, with higher BMI often associated with higher diabetes risk.
- **DiabetesPedigreeFunction** → A function that scores likelihood of diabetes based on family history. It incorporates genetic relationships and the variation in age distribution among family members.
- **Age** → Age of the patient in years. Older age is a risk factor for diabetes.
- **Outcome** → Class variable (0 or 1). This is the target variable where 0 indicates the absence of diabetes and 1 indicates the presence of diabetes.

Exercise instructions:

Use the **`pima-indians-diabetes.csv`** file.

You can load it from the following URL:

https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv

Use the following code to extract the dataset into your Python dataframe:

```
url = "https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv"
df = pd.read_csv(url)
```

Data preparation:

1. Check for missing values in the dataset.
2. Check for duplicate rows in the dataset.
3. In case you found any of those, remove them from the df.

Data Exploration:

1. Display the first few rows of the dataset.
2. Get a summary of the dataset using descriptive statistics.
3. Print the number of rows and columns in the dataset.
4. Calculate the mean, median, standard deviation, and other descriptive statistics for each column.
5. Plot the distribution of the target variable (`Outcome`).
   Do you see a risk for accuracy paradox?
6. Calculate and plot the correlation matrix to understand the relationships between features.
7. Calculate and plot the correlation between each feature and the outcome label.
   Do you see features that have no linear relation?

Logistic Regression Machine Learning:

1. Use the your preprocessing dataset
2. Apply feature scaling on the entire features of type Standardization.
3. Apply Simple Logistic Regression to classify whether a person has diabetes.
4. Print your model predictions on the testset.
5. Print your model beta coefficient values for each feature.
6. Print your model probability values for each prediction.
   Does the model coefficient values match your expectations from the data exploration?
7. Print the accuracy, precision, recall, and F1-score of the model.
   According to your model accuracy score do you see any risk for accuracy paradox?
8. Plot the confusion matrix of your model.

Model Deployment:

1. Train your Logistic Regression model on the entire dataset.
2. Print the final beta coefficient the model found for each feature.
3. Export your final model into a joblib file.
   Make sure you also export other relevant preprocessing instances such as the standard scaler.
4. Import your final model and the preprocessing instances from the joblib files and load them back to your working area.
5. Use the import model to predict the ‘close’ value of the following unknown data points:
   a. **Pregnancies:** 4, **Glucose:** 112, **Blood Pressure:** 78, **Skin Thickness:** 31, **Insulin:** 0, **BMI:** 39.4, **Diabetes Pedigree Function:** 0.236, **Age:** 33
   b. **Pregnancies:** 7, **Glucose:** 150, **Blood Pressure:** 66, **Skin Thickness:** 42, **Insulin:** 342, **BMI:** 34.7, **Diabetes Pedigree Function:** 0.718, **Age:** 42
   c. **Pregnancies:** 1, **Glucose:** 99, **Blood Pressure:** 58, **Skin Thickness:** 10, **Insulin:** 0, **BMI:** 25.4, **Diabetes Pedigree Function:** 0.551, **Age:** 21

For this exercise you can use the following code for convenient:

```
data = {
        'Pregnancies': [4, 7, 1],
        'Glucose': [112, 150, 99],
        'BloodPressure': [78, 66, 58],
        'SkinThickness': [31, 42, 10],
        'Insulin': [0, 342, 0],
        'BMI': [39.4, 34.7, 25.4],
        'DiabetesPedigreeFunction': [0.236, 0.718, 0.551],
        'Age': [33, 42, 21]
}
new_data_df = pd.DataFrame(data)
```