Jhillian M. Cabos CPE22S3

Hands-on Activity 11.2 Classification using Logistic Regression

Start Assignment

Due Apr 28 by 11:59pm **Points** 18 **Submitting** a file upload **File Types** pdf
Available Apr 24 at 12am - Apr 28 at 11:59pm

Objective(s):

- This activity aims to demonstrate how to apply simple linear regression analysis to solve regression problem

Intended Learning Outcomes (ILOs):

- Demonstrate how to solve classification problems using Logistic Regression
- Use the logistic regression model to perform classification

Resources:

- Jupyter Notebook
- Dataset: https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29

Submission Requirements:

- PDF containing initial EDA and Data Wrangling
- PDF showing demonstration of simple linear regression.
- Submit a link to the colab file through the comment section.

Import Dataset

```
pip install ucimlrepo
```

```
Collecting ucimlrepo
  Downloading ucimlrepo-0.0.6-py3-none-any.whl (8.0 kB)
Installing collected packages: ucimlrepo
Successfully installed ucimlrepo-0.0.6
```

```
from ucimlrepo import fetch_ucirepo

# fetch dataset
cervical_cancer_risk_factors = fetch_ucirepo(id=383)

# data (as pandas dataframes)
X = cervical_cancer_risk_factors.data.features
y = cervical_cancer_risk_factors.data.targets

# metadata
print(cervical_cancer_risk_factors.metadata)

# variable information
print(cervical_cancer_risk_factors.variables)
```

```
25          STDs: Number of diagnosis  Feature     Integer        None
26    STDs: Time since first diagnosis  Feature  Continuous        None
27     STDs: Time since last diagnosis  Feature  Continuous        None
28                           Dx:Cancer  Feature     Integer        None
29                              Dx:CIN  Feature     Integer        None
30                              Dx:HPV  Feature     Integer        None
31                                  Dx  Feature     Integer        None
32                          Hinselmann  Feature     Integer        None
33                            Schiller  Feature     Integer        None
34                            Citology  Feature     Integer        None
35                               Biopsy  Feature     Integer        None

   description units missing_values
0         None  None             no
1         None  None            yes
2         None  None            yes
3         None  None            yes
4         None  None            yes
5         None  None            yes
6         None  None            yes
7         None  None            yes
8         None  None            yes
9         None  None            yes
10        None  None            yes
11        None  None            yes
12        None  None            yes
13        None  None            yes
14        None  None            yes
15        None  None            yes
16        None  None            yes
17        None  None            yes
18        None  None            yes
19        None  None            yes
20        None  None            yes
21        None  None            yes
22        None  None            yes
23        None  None            yes
24        None  None            yes
25        None  None             no
26        None  None            yes
27        None  None            yes
28        None  None             no
29        None  None             no
30        None  None             no
31        None  None             no
32        None  None             no
33        None  None             no
34        None  None             no
35        None  None             no
```

```
import pandas as pd
```

Concat()

```
df = pd.concat([X, y], axis = 1)
```

Checking the first 5 of the dataset

```
df.head(5)
```

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD	...	STDs: Time since first diagnosis	STDs: Time since last diagnosis
0	18	4.0	15.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	NaN
1	15	1.0	14.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	NaN
2	34	1.0	NaN	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	NaN
3	52	5.0	16.0	4.0	1.0	37.0	37.0	1.0	3.0	0.0	...	NaN	NaN
4	46	3.0	21.0	4.0	0.0	0.0	0.0	1.0	15.0	0.0	...	NaN	NaN

5 rows × 36 columns

Provides a concise summary of the DataFrame df

+ Code + Text

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 858 entries, 0 to 857
Data columns (total 36 columns):
 #   Column                              Non-Null Count  Dtype  
---  ------                              --------------  -----  
 0   Age                                 858 non-null    int64  
 1   Number of sexual partners           832 non-null    float64
 2   First sexual intercourse            851 non-null    float64
 3   Num of pregnancies                  802 non-null    float64
 4   Smokes                              845 non-null    float64
 5   Smokes (years)                      845 non-null    float64
 6   Smokes (packs/year)                 845 non-null    float64
 7   Hormonal Contraceptives             750 non-null    float64
 8   Hormonal Contraceptives (years)     750 non-null    float64
 9   IUD                                 741 non-null    float64
 10  IUD (years)                         741 non-null    float64
 11  STDs                                753 non-null    float64
 12  STDs (number)                       753 non-null    float64
 13  STDs:condylomatosis                 753 non-null    float64
 14  STDs:cervical condylomatosis        753 non-null    float64
 15  STDs:vaginal condylomatosis         753 non-null    float64
 16  STDs:vulvo-perineal condylomatosis  753 non-null    float64
 17  STDs:syphilis                       753 non-null    float64
 18  STDs:pelvic inflammatory disease    753 non-null    float64
 19  STDs:genital herpes                 753 non-null    float64
 20  STDs:molluscum contagiosum          753 non-null    float64
 21  STDs:AIDS                           753 non-null    float64
 22  STDs:HIV                            753 non-null    float64
 23  STDs:Hepatitis B                    753 non-null    float64
 24  STDs:HPV                            753 non-null    float64
 25  STDs: Number of diagnosis           858 non-null    int64  
 26  STDs: Time since first diagnosis    71 non-null     float64
 27  STDs: Time since last diagnosis     71 non-null     float64
 28  Dx:Cancer                           858 non-null    int64  
 29  Dx:CIN                              858 non-null    int64  
 30  Dx:HPV                              858 non-null    int64  
 31  Dx                                  858 non-null    int64  
 32  Hinselmann                          858 non-null    int64  
 33  Schiller                            858 non-null    int64  
 34  Citology                            858 non-null    int64  
 35  Biopsy                              858 non-null    int64  
dtypes: float64(26), int64(10)
memory usage: 241.4 KB
```

Summarize the central tendency

```
df.describe()
```

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD
count	858.000000	832.000000	851.000000	802.000000	845.000000	845.000000	845.000000	750.000000	750.000000	741.000000
mean	26.820513	2.527644	16.995300	2.275561	0.145562	1.219721	0.453144	0.641333	2.256419	0.112011
std	8.497948	1.667760	2.803355	1.447414	0.352876	4.089017	2.226610	0.479929	3.764254	0.315593
min	13.000000	1.000000	10.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	20.000000	2.000000	15.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	25.000000	2.000000	17.000000	2.000000	0.000000	0.000000	0.000000	1.000000	0.500000	0.000000
75%	32.000000	3.000000	18.000000	3.000000	0.000000	0.000000	0.000000	1.000000	3.000000	0.000000
max	84.000000	28.000000	32.000000	11.000000	1.000000	37.000000	37.000000	1.000000	30.000000	1.000000

8 rows × 36 columns

Returns a tuple representing the dimensions of the DataFrame df

```
df.shape
```

```
(858, 36)
```

We already know what this does

```
df.columns
```

```
Index(['Age', 'Number of sexual partners', 'First sexual intercourse',
       'Num of pregnancies', 'Smokes', 'Smokes (years)', 'Smokes (packs/year)',
       'Hormonal Contraceptives', 'Hormonal Contraceptives (years)', 'IUD',
       'IUD (years)', 'STDs', 'STDs (number)', 'STDs:condylomatosis',
       'STDs:cervical condylomatosis', 'STDs:vaginal condylomatosis',
       'STDs:vulvo-perineal condylomatosis', 'STDs:syphilis',
       'STDs:pelvic inflammatory disease', 'STDs:genital herpes',
       'STDs:molluscum contagiosum', 'STDs:AIDS', 'STDs:HIV',
       'STDs:Hepatitis B', 'STDs:HPV', 'STDs: Number of diagnosis',
       'STDs: Time since first diagnosis', 'STDs: Time since last diagnosis',
       'Dx:Cancer', 'Dx:CIN', 'Dx:HPV', 'Dx', 'Hinselmann', 'Schiller',
       'Citology', 'Biopsy'],
      dtype='object')
```

Calculates the number of missing values

```
df.isnull().sum()
```

```
Age                                   0
Number of sexual partners            26
First sexual intercourse              7
Num of pregnancies                   56
Smokes                               13
Smokes (years)                       13
Smokes (packs/year)                  13
Hormonal Contraceptives             108
Hormonal Contraceptives (years)     108
IUD                                 117
IUD (years)                         117
STDs                                105
STDs (number)                       105
STDs:condylomatosis                 105
STDs:cervical condylomatosis        105
STDs:vaginal condylomatosis         105
STDs:vulvo-perineal condylomatosis   105
STDs:syphilis                       105
STDs:pelvic inflammatory disease    105
STDs:genital herpes                 105
STDs:molluscum contagiosum          105
STDs:AIDS                           105
STDs:HIV                            105
STDs:Hepatitis B                    105
STDs:HPV                            105
STDs: Number of diagnosis             0
STDs: Time since first diagnosis    787
STDs: Time since last diagnosis     787
Dx:Cancer                             0
Dx:CIN                                0
Dx:HPV                                0
Dx                                    0
Hinselmann                            0
Schiller                              0
Citology                              0
Biopsy                                0
dtype: int64
```

Replaces missing values in the DataFrame df with the mean value of each column, modifying the DataFrame in place.

```
df.fillna(df.mean(), inplace=True)
```

Returns a Series containing the data type of each column in the DataFrame df.

```
df.dtypes
```

```
Age                                   int64
Number of sexual partners           float64
```

```
First sexual intercourse             float64
Num of pregnancies                   float64
Smokes                               float64
Smokes (years)                       float64
Smokes (packs/year)                  float64
Hormonal Contraceptives              float64
Hormonal Contraceptives (years)      float64
IUD                                  float64
IUD (years)                          float64
STDs                                 float64
STDs (number)                        float64
STDs:condylomatosis                  float64
STDs:cervical condylomatosis         float64
STDs:vaginal condylomatosis          float64
STDs:vulvo-perineal condylomatosis    float64
STDs:syphilis                        float64
STDs:pelvic inflammatory disease     float64
STDs:genital herpes                  float64
STDs:molluscum contagiosum           float64
STDs:AIDS                            float64
STDs:HIV                             float64
STDs:Hepatitis B                     float64
STDs:HPV                             float64
STDs: Number of diagnosis              int64
STDs: Time since first diagnosis     float64
STDs: Time since last diagnosis      float64
Dx:Cancer                              int64
Dx:CIN                                 int64
Dx:HPV                                 int64
Dx                                     int64
Hinselmann                             int64
Schiller                               int64
Citology                               int64
Biopsy                                 int64
dtype: object
```

```
# Convert all columns to integers
df = df.astype(int)
```

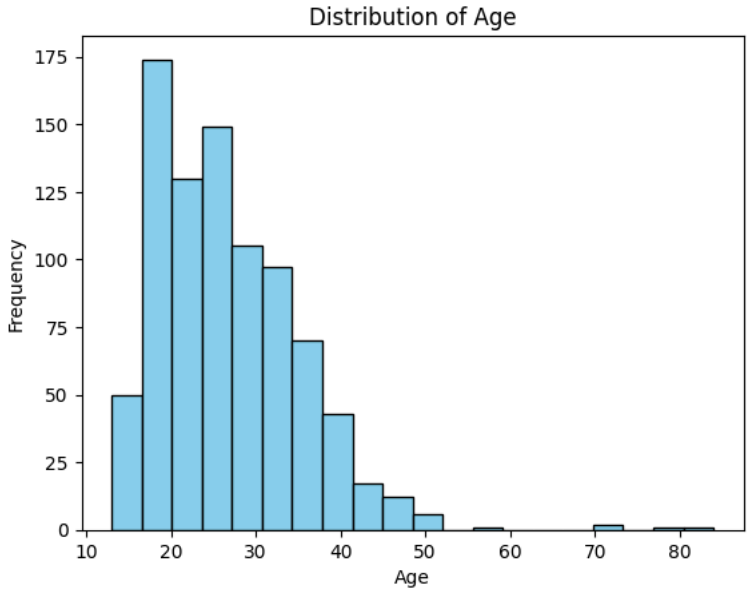
```
df.head(10)
```

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD	...	STDs: Time since first diagnosis	STDs: Time since last diagnosis
0	18	4	15	1	0	0	0	0	0	0	...	6	5
1	15	1	14	1	0	0	0	0	0	0	...	6	5
2	34	1	16	1	0	0	0	0	0	0	...	6	5
3	52	5	16	4	1	37	37	1	3	0	...	6	5
4	46	3	21	4	0	0	0	1	15	0	...	6	5
5	42	3	23	2	0	0	0	0	0	0	...	6	5
6	51	3	17	6	1	34	3	0	0	1	...	6	5
7	26	1	26	3	0	0	0	1	2	1	...	6	5
8	45	1	20	5	0	0	0	0	0	0	...	6	5
9	44	3	15	2	1	1	2	0	0	0	...	6	5

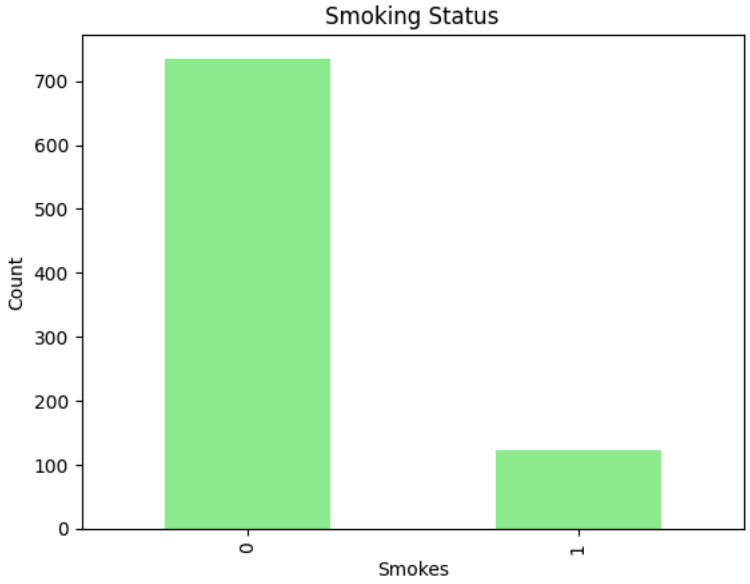
10 rows × 36 columns

This code generates a histogram to visualize the distribution of ages in the DataFrame df.

```
import matplotlib.pyplot as plt
plt.hist(df['Age'], bins=20, color='skyblue', edgecolor='black')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

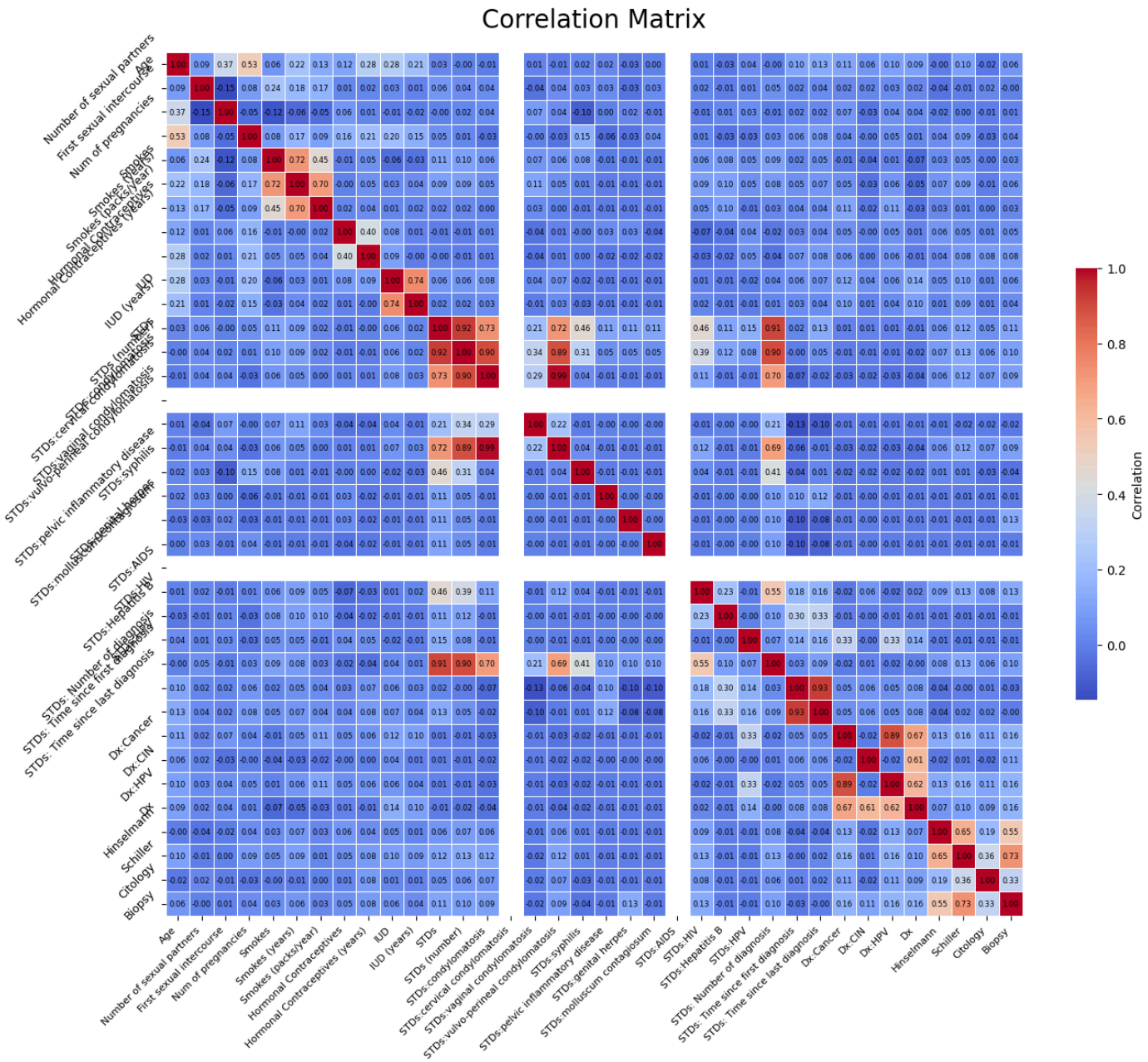



```
# Bar plot for Smokes
df['Smokes'].value_counts().plot(kind='bar', color='lightgreen')
plt.title('Smoking Status')
plt.xlabel('Smokes')
plt.ylabel('Count')
plt.show()
```



This code generates a heatmap to visualize the correlation matrix of the DataFrame df.

```
import seaborn as sns
import matplotlib.pyplot as plt
corr = df.corr()
plt.figure(figsize=(14, 12))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f", annot_kws={"size": 6, "color": "black"}, cbar_kws={"shrink": .5, "label": "Correla
plt.title('Correlation Matrix', fontsize=20, pad=20)
plt.xticks(rotation=45, ha='right', fontsize=8)
plt.yticks(rotation=45, fontsize=9)
plt.tight_layout()
plt.show()
```


Correlation Matrix
Correlation
1.0
0.8
0.6
0.4
0.2
0.0
Age
Number of sexual partners
First sexual intercourse
Num of pregnancies
Smokes
Smokes (years)
Smokes (packs/year)
Hormonal Contraceptives
Hormonal Contraceptives (years)
IUD
IUD (years)
STDs
STDs (number)
STDs:condylomatosis
STDs:cervical condylomatosis
STDs:vaginal condylomatosis
STDs:vulvo-perineal condylomatosis
STDs:pelvic inflammatory disease
STDs:syphilis
STDs:genital herpes
STDs:molluscum contagiosum
STDs:AIDS
STDs:HIV
STDs:Hepatitis B
STDs:HPV
STDs: Number of diagnosis
STDs: Time since first diagnosis
STDs: Time since last diagnosis
Dx:Cancer
Dx:CIN
Dx:HPV
Dx
Hinselmann
Schiller
Citology
Biopsy

x

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD	...	STDs: Time since first diagnosis	STD Ti sin la diagnos
0	18	4.0	15.0	1.0	0.0	0.0	0.0	0.0	0.00	0.0	...	NaN	Na
1	15	1.0	14.0	1.0	0.0	0.0	0.0	0.0	0.00	0.0	...	NaN	Na
2	34	1.0	NaN	1.0	0.0	0.0	0.0	0.0	0.00	0.0	...	NaN	Na
3	52	5.0	16.0	4.0	1.0	37.0	37.0	1.0	3.00	0.0	...	NaN	Na
4	46	3.0	21.0	4.0	0.0	0.0	0.0	1.0	15.00	0.0	...	NaN	Na
...	...	...	...	...	...	...	...	...	...	...	...	...	
853	34	3.0	18.0	0.0	0.0	0.0	0.0	0.0	0.00	0.0	...	NaN	Na
854	32	2.0	19.0	1.0	0.0	0.0	0.0	1.0	8.00	0.0	...	NaN	Na
855	25	2.0	17.0	0.0	0.0	0.0	0.0	1.0	0.08	0.0	...	NaN	Na
856	33	2.0	24.0	2.0	0.0	0.0	0.0	1.0	0.08	0.0	...	NaN	Na
857	29	2.0	20.0	1.0	0.0	0.0	0.0	1.0	0.50	0.0	...	NaN	Na

858 rows × 36 columns

```
y
```

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

X = df[['Age', 'Number of sexual partners']].values
y = df['Num of pregnancies'].values

X = np.nan_to_num(X)
y = np.nan_to_num(y)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("Mean Squared Error:", mse)
print("R-squared:", r2)
```

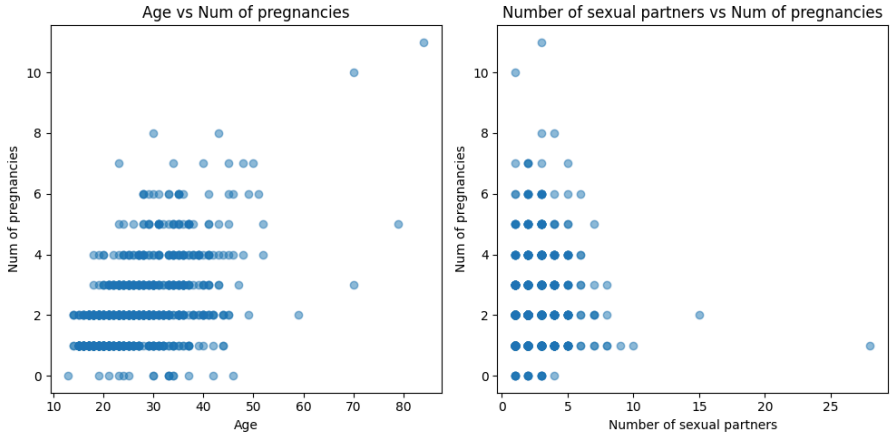
```
Mean Squared Error: 1.622961011457525
R-squared: 0.2556826613707128
```

```
import matplotlib.pyplot as plt

# Scatter plot of Age vs Num of pregnancies
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.scatter(df['Age'], df['Num of pregnancies'], alpha=0.5)
plt.title('Age vs Num of pregnancies')
plt.xlabel('Age')
plt.ylabel('Num of pregnancies')

# Scatter plot of Number of sexual partners vs Num of pregnancies
plt.subplot(1, 2, 2)
plt.scatter(df['Number of sexual partners'], df['Num of pregnancies'], alpha=0.5)
plt.title('Number of sexual partners vs Num of pregnancies')
plt.xlabel('Number of sexual partners')
plt.ylabel('Num of pregnancies')

plt.tight_layout()
plt.show()
```



From the scatter plots, it's clear that there's no straightforward relationship between age or number of sexual partners and the number of pregnancies. For age, we see a mix of high and low pregnancy counts across different ages, with no clear trend. Similarly, the number of sexual partners doesn't seem to predict the number of pregnancies consistently.

In short, these scatter plots suggest that age and number of sexual partners alone might not be reliable indicators of the number of pregnancies. Other factors might play a more significant role in determining pregnancy outcomes.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
# Handle missing values
df.fillna(df.mean(), inplace=True)

# Define a binary target variable based on the number of pregnancies
df['High_num_pregnancies'] = (df['Num of pregnancies'] > 3).astype(int)

# Select relevant features (e.g., Age and Number of sexual partners)
X = df[['Age', 'Number of sexual partners']].values
y = df['High_num_pregnancies'].values

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train the logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

print("Classification Report:")
```

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

def plot_decision_boundary(X, y, model, title):
    h = .02
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))

    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    plt.contourf(xx, yy, Z, alpha=0.3, cmap='coolwarm')

    plt.scatter(X[:, 0], X[:, 1], c=y, cmap='coolwarm')
    plt.xlabel('Age')
    plt.ylabel('Number of sexual partners')
    plt.title(title)

plt.figure(figsize=(8, 6))
plot_decision_boundary(X_train, y_train, model, "Logistic Regression (Training set)")
plt.show()
```