```
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
```

Import Data Set

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix
```

```python
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)

```

```
16  STDs:vulvo-perineal condylomatosis  Feature  Continuous        None
17                       STDs:syphilis  Feature  Continuous        None
18    STDs:pelvic inflammatory disease  Feature  Continuous        None
19                 STDs:genital herpes  Feature  Continuous        None
20          STDs:molluscum contagiosum  Feature  Continuous        None
21                           STDs:AIDS  Feature  Continuous        None
22                            STDs:HIV  Feature  Continuous        None
23                    STDs:Hepatitis B  Feature  Continuous        None
24                            STDs:HPV  Feature  Continuous        None
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
```

Exploratory Data Analysis

```python
print("Shape of X:", X.shape)
```

```
Shape of X: (858, 36)
```

```python
X.head()
```

	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	Dx:Cancer	Dx:CIN	Dx:HPV	Dx	Hinselmann	Schiller	Citology
0	18	4.0	15.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	NaN	0	0	0	0	0	0	0
1	15	1.0	14.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	NaN	0	0	0	0	0	0	0
2	34	1.0	NaN	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	NaN	0	0	0	0	0	0	0
3	52	5.0	16.0	4.0	1.0	37.0	37.0	1.0	3.0	0.0	...	NaN	NaN	1	0	1	0	0	0	0
4	46	3.0	21.0	4.0	0.0	0.0	0.0	1.0	15.0	0.0	...	NaN	NaN	0	0	0	0	0	0	0

Check for missing Values

```python
print("Missing values in X:", X.isnull().sum().sum())
```

```
Missing values in X: 3622
```

Check for categorical Values

```python
col_names = X.columns
```

```python
col_names
```

```
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')
```

```
1 print(X.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
```

```
1 categorical_columns = X.select_dtypes(include=['object']).columns.tolist()
```

```
1 if len(categorical_columns) > 0:
2     print("Categorical columns:")
3     print(categorical_columns)
4 else:
5     print("No categorical columns found in X.")
```

```
No categorical columns found in X.
```

No Categorical Values so Move on to Numerical Variables

Explore Numerical values

```
1 X_df = pd.DataFrame(X)
```

```
1 numerical = [var for var in X_df.columns if X_df[var].dtype != 'object']
```

```
1 print('There are {} numerical variables\n'.format(len(numerical)))
2 print('Numerical variables are: ', numerical)
```

```
There are 36 numerical variables

Numerical variables are:  ['Age', 'Number of sexual partners', 'First sexual intercourse', 'Num of pregnancies', 'Smokes', 'Smokes (years)', 'Smokes (packs/year)', 'Hormonal Contraceptives', 'Hormonal Contraceptives (years)', 'IUD', 'I
```

Explore Numerical values Problems

Recheck missing values

```
1 X_df[numerical].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
```

Check for Outliers

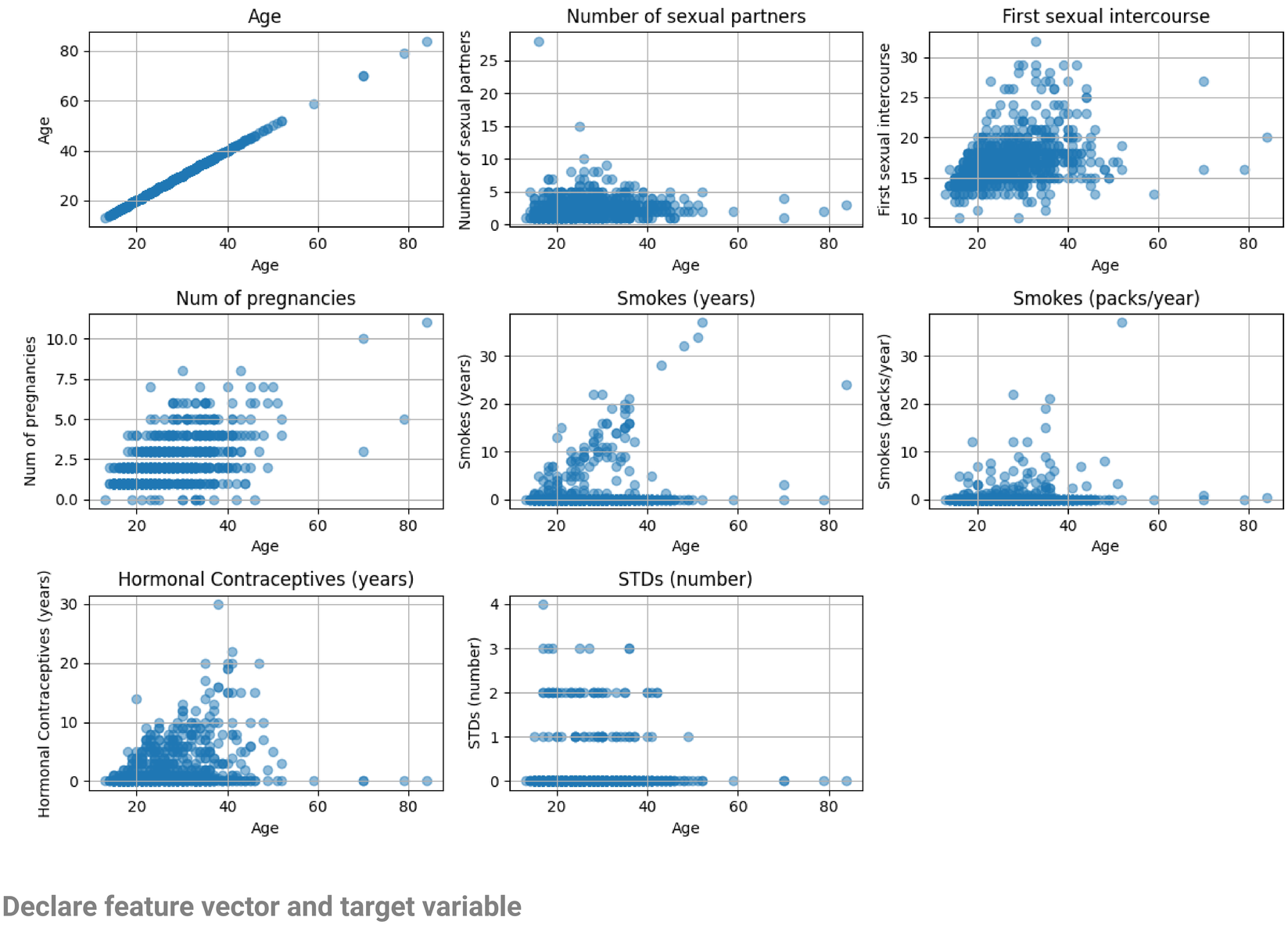
```
1 print(round(X_df[numerical].describe()),2)
```

```
std      8.0                       2.0                     3.0
min     13.0                       1.0                    10.0
25%     20.0                       2.0                    15.0
50%     25.0                       2.0                    17.0
75%     32.0                       3.0                    18.0
max     84.0                      28.0                    32.0

       Num of pregnancies  Smokes  Smokes (years)  Smokes (packs/year)  \
```

```
min                  0.0     0.0            0.0                 0.0
25%                  1.0     0.0            0.0                 0.0
50%                  2.0     0.0            0.0                 0.0
75%                  3.0     0.0            0.0                 0.0
max                 11.0     1.0           37.0                37.0

       Hormonal Contraceptives  Hormonal Contraceptives (years)    IUD  ...  \
count                    750.0                            750.0  741.0  ...
mean                       1.0                              2.0    0.0  ...
std                        0.0                              4.0    0.0  ...
min                        0.0                              0.0    0.0  ...
25%                        0.0                              0.0    0.0  ...
50%                        1.0                              0.0    0.0  ...
75%                        1.0                              3.0    0.0  ...
max                        1.0                             30.0    1.0  ...

       STDs: Time since first diagnosis  STDs: Time since last diagnosis  \
count                              71.0                             71.0
mean                                6.0                              6.0
std                                 6.0                              6.0
min                                 1.0                              1.0
25%                                 2.0                              2.0
50%                                 4.0                              3.0
75%                                 8.0                              8.0
max                                22.0                             22.0

       Dx:Cancer  Dx:CIN  Dx:HPV     Dx  Hinselmann  Schiller  Citology  \
count      858.0   858.0   858.0  858.0       858.0     858.0     858.0
mean         0.0     0.0     0.0    0.0         0.0       0.0       0.0
std          0.0     0.0     0.0    0.0         0.0       0.0       0.0
min          0.0     0.0     0.0    0.0         0.0       0.0       0.0
25%          0.0     0.0     0.0    0.0         0.0       0.0       0.0
50%          0.0     0.0     0.0    0.0         0.0       0.0       0.0
75%          0.0     0.0     0.0    0.0         0.0       0.0       0.0
max          1.0     1.0     1.0    1.0         1.0       1.0       1.0

       Biopsy
count   858.0
mean      0.0
std       0.0
min       0.0
25%       0.0
50%       0.0
75%       0.0
max       1.0

[8 rows x 36 columns] 2
```

Check for distribution of variables

```
import pandas as pd
import matplotlib.pyplot as plt

numerical_vars = [
    'Age', 'Number of sexual partners', 'First sexual intercourse',
    'Num of pregnancies', 'Smokes (years)', 'Smokes (packs/year)',
    'Hormonal Contraceptives (years)', 'STDs (number)'
]

X_G = pd.DataFrame(X, columns=numerical_vars)

plt.figure(figsize=(12, 8))
for i, var in enumerate(numerical_vars, start=1):
    plt.subplot(3, 3, i)

    plt.scatter(X_df['Age'], X_df[var], alpha=0.5)

    plt.title(var)
    plt.xlabel('Age')
    plt.ylabel(var)
    plt.grid(True)

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



Declare feature vector and target variable

```
X = X_df.drop(['Age'], axis = 1)

y = X_df['Age']
```

Split data into separate training and test set

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
```

```
X_train.shape, X_test.shape
```

```
((600, 35), (258, 35))
```

Feature Engineering

```
X_train.dtypes
```

```
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
```

```
categorical = [col for col in X_train.columns if X_train[col].dtypes == '0']

categorical
```

```
[]
```

```
numerical = [col for col in X_train.columns if X_train[col].dtypes != '0']

numerical
```

```
['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']
```

Engineering missing values in numerical varibales

```
X_train[numerical].isnull().sum()
```

```
Number of sexual partners             18
First sexual intercourse               4
Num of pregnancies                    36
Smokes                                 9
Smokes (years)                         9
Smokes (packs/year)                    9
Hormonal Contraceptives               70
Hormonal Contraceptives (years)       70
IUD                                   76
IUD (years)                           76
STDs                                  68
STDs (number)                         68
STDs:condylomatosis                   68
STDs:cervical condylomatosis          68
STDs:vaginal condylomatosis           68
STDs:vulvo-perineal condylomatosis     68
STDs:syphilis                         68
STDs:pelvic inflammatory disease      68
STDs:genital herpes                   68
STDs:molluscum contagiosum            68
STDs:AIDS                             68
STDs:HIV                              68
STDs:Hepatitis B                      68
STDs:HPV                              68
STDs: Number of diagnosis              0
STDs: Time since first diagnosis     549
STDs: Time since last diagnosis      549
Dx:Cancer                              0
Dx:CIN                                 0
Dx:HPV                                 0
Dx                                     0
Hinselmann                             0
Schiller                               0
Citology                               0
Biopsy                                 0
dtype: int64
```

```
X_test[numerical].isnull().sum()
```

```
Number of sexual partners              8
First sexual intercourse               3
Num of pregnancies                    20
Smokes                                 4
Smokes (years)                         4
Smokes (packs/year)                    4
Hormonal Contraceptives               38
Hormonal Contraceptives (years)       38
IUD                                   41
IUD (years)                           41
STDs                                  37
STDs (number)                         37
STDs:condylomatosis                   37
STDs:cervical condylomatosis          37
```

```
STDs:vaginal condylomatosis            37
STDs:vulvo-perineal condylomatosis     37
STDs:syphilis                          37
STDs:pelvic inflammatory disease       37
STDs:genital herpes                    37
STDs:molluscum contagiosum             37
STDs:AIDS                              37
STDs:HIV                               37
STDs:Hepatitis B                       37
STDs:HPV                               37
STDs: Number of diagnosis               0
STDs: Time since first diagnosis      238
STDs: Time since last diagnosis       238
Dx:Cancer                               0
Dx:CIN                                  0
Dx:HPV                                  0
Dx                                      0
Hinselmann                              0
Schiller                                0
Citology                                0
Biopsy                                  0
dtype: int64
```

```
for col in numerical:

    if X_train[col].isnull().mean() > 0:

        missing_percentage = round(X_train[col].isnull().mean() * 100, 2)

        print(f"{col:<30} {missing_percentage:>10}% missing values")
```

```
Number of sexual partners             3.0% missing values
First sexual intercourse             0.67% missing values
Num of pregnancies                    6.0% missing values
Smokes                                1.5% missing values
Smokes (years)                        1.5% missing values
Smokes (packs/year)                   1.5% missing values
Hormonal Contraceptives              11.67% missing values
Hormonal Contraceptives (years)       11.67% missing values
IUD                                  12.67% missing values
IUD (years)                          12.67% missing values
STDs                                 11.33% missing values
STDs (number)                        11.33% missing values
STDs:condylomatosis                  11.33% missing values
STDs:cervical condylomatosis         11.33% missing values
STDs:vaginal condylomatosis          11.33% missing values
STDs:vulvo-perineal condylomatosis      11.33% missing values
STDs:syphilis                        11.33% missing values
STDs:pelvic inflammatory disease      11.33% missing values
STDs:genital herpes                  11.33% missing values
STDs:molluscum contagiosum           11.33% missing values
STDs:AIDS                            11.33% missing values
STDs:HIV                             11.33% missing values
STDs:Hepatitis B                     11.33% missing values
STDs:HPV                             11.33% missing values
STDs: Time since first diagnosis        91.5% missing values
STDs: Time since last diagnosis       91.5% missing values
```

```
for df1 in [X_train, X_test]:
  for col in numerical:
    col_median=X_train[col].median()
    df1[col].fillna(col_median, inplace=True)
```

```
X_train[numerical].isnull().sum()
```

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

```
X_test[numerical].isnull().sum()
```

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

Feature Scaling

```
1 X_train.describe()
```

	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: Time since first diagnosis	STDs: Time since last diagnosis	Dx:Cancer	Dx:CIN	Dx:HPV
count	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	...	600.000000	600.000000	600.000000	600.000000	600.000000
mean	2.478333	16.893333	2.276667	0.143333	1.252103	0.484163	0.676667	2.005900	0.090000	0.415533	...	4.268333	4.230000	0.015000	0.010000	0.013333
std	1.362335	2.567023	1.436510	0.350705	4.161544	2.351749	0.468139	3.465857	0.286421	1.808633	...	2.060762	1.980818	0.121654	0.099582	0.114793
min	1.000000	11.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	1.000000	1.000000	0.000000	0.000000	0.000000
25%	2.000000	15.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	4.000000	4.000000	0.000000	0.000000	0.000000
50%	2.000000	17.000000	2.000000	0.000000	0.000000	0.000000	1.000000	0.420000	0.000000	0.000000	...	4.000000	4.000000	0.000000	0.000000	0.000000
75%	3.000000	18.000000	3.000000	0.000000	0.000000	0.000000	1.000000	2.000000	0.000000	0.000000	...	4.000000	4.000000	0.000000	0.000000	0.000000
max	10.000000	29.000000	11.000000	1.000000	37.000000	37.000000	1.000000	22.000000	1.000000	19.000000	...	22.000000	22.000000	1.000000	1.000000	1.000000

8 rows × 35 columns

```
1 cols = X_train.columns
```

```
1 from sklearn.preprocessing import MinMaxScaler
2
3 scaler = MinMaxScaler()
4
5 X_train = scaler.fit_transform(X_train)
6
7 X_test = scaler.transform(X_test)
```

```
1 X_train = pd.DataFrame(X_train, columns=[cols])
```

```
1 X_test = pd.DataFrame(X_test, columns=[cols])
```

```
1 X_train.describe()
```

	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: Time since first diagnosis	STDs: Time since last diagnosis	Dx:Cancer	Dx:CIN	Dx:HPV
count	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	600.000000	...	600.000000	600.000000	600.000000	600.000000	600.000000
mean	0.164259	0.327407	0.206970	0.143333	0.033841	0.013085	0.676667	0.091177	0.090000	0.021870	...	0.155635	0.153810	0.015000	0.010000	0.013333
std	0.151371	0.142612	0.130592	0.350705	0.112474	0.063561	0.468139	0.157539	0.286421	0.095191	...	0.098132	0.094325	0.121654	0.099582	0.114793
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.111111	0.222222	0.090909	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.142857	0.142857	0.000000	0.000000	0.000000
50%	0.111111	0.333333	0.181818	0.000000	0.000000	0.000000	1.000000	0.019091	0.000000	0.000000	...	0.142857	0.142857	0.000000	0.000000	0.000000
75%	0.222222	0.388889	0.272727	0.000000	0.000000	0.000000	1.000000	0.090909	0.000000	0.000000	...	0.142857	0.142857	0.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	...	1.000000	1.000000	1.000000	1.000000	1.000000

8 rows × 35 columns

Model training

```
1 from sklearn.linear_model import LogisticRegression
2
3 logreg = LogisticRegression(solver='liblinear', random_state = 0)
4
5 logreg.fit(X_train, y_train)
```

```
LogisticRegression(random_state=0, solver='liblinear')
```

Predict Results

```
1 y_pred_test = logreg.predict(X_test)
2
3 y_pred_test
```

```
array([26, 19, 18, 24, 18, 26, 18, 20, 21, 41, 19, 19, 23, 23, 19, 18, 19,
       18, 31, 23, 18, 23, 35, 30, 19, 23, 34, 18, 23, 19, 23, 19, 35, 19,
       19, 28, 19, 36, 19, 19, 18, 21, 21, 19, 26, 18, 19, 23, 26, 34, 26,
       19, 23, 19, 18, 24, 19, 18, 19, 34, 35, 19, 19, 17, 19, 18, 19, 25,
       19, 21, 28, 18, 23, 18, 18, 27, 27, 35, 25, 19, 18, 19, 19, 31, 19,
       24, 21, 18, 24, 19, 24, 18, 19, 19, 19, 19, 19, 18, 27, 19, 19, 24,
       28, 26, 19, 19, 28, 19, 40, 24, 18, 17, 19, 33, 19, 34, 18, 18, 18,
       24, 18, 21, 23, 23, 26, 19, 21, 21, 26, 19, 23, 30, 24, 19, 25, 25,
       34, 19, 25, 19, 18, 18, 19, 25, 18, 18, 34, 19, 19, 18, 19, 18, 18,
       19, 23, 18, 26, 19, 24, 25, 19, 18, 24, 25, 18, 19, 19, 23, 19, 25,
       24, 18, 23, 34, 19, 23, 21, 17, 19, 19, 35, 34, 18, 26, 21, 23, 20,
       30, 19, 18, 18, 19, 30, 19, 24, 20, 18, 21, 19, 20, 30, 19, 19, 18,
       19, 24, 19, 19, 27, 19, 23, 18, 26, 24, 19, 30, 34, 19, 20, 26, 19,
       19, 18, 40, 18, 34, 19, 18, 18, 18, 18, 18, 19, 18, 19, 19, 23, 19,
       19, 18, 27, 18, 18, 34, 35, 23, 19, 19, 25, 30, 27, 26, 23, 35, 21,
       18, 19, 31])
```

```
1 logreg.predict_proba(X_test)[:,0]
```

```
array([0.00257353, 0.00399272, 0.0100894 , 0.00281407, 0.01054731,
       0.00301097, 0.0108993 , 0.00401071, 0.00698407, 0.00197939,
       0.0039962 , 0.00376588, 0.00268822, 0.00267909, 0.00377783,
       0.01188848, 0.00367413, 0.01062505, 0.00177621, 0.0028448 ,
       0.01205777, 0.00314333, 0.00324846, 0.00817091, 0.00369901,
       0.0024679 , 0.00931322, 0.01211857, 0.00331648, 0.00371361,
       0.00361802, 0.00389594, 0.0022301 , 0.00377806, 0.00389483,
       0.00318388, 0.00354974, 0.00183991, 0.00362241, 0.00409199,
       0.01109818, 0.00245209, 0.00244971, 0.00423392, 0.00964493,
       0.00716898, 0.00379468, 0.00236979, 0.00299913, 0.00338255,
       0.00262294, 0.00389388, 0.00312507, 0.00370023, 0.01002922,
       0.00371741, 0.00362241, 0.01253573, 0.00198429, 0.00831954,
       0.00295478, 0.00375978, 0.00341799, 0.00432336, 0.00379353,
       0.00279343, 0.0036232 , 0.00274162, 0.00376753, 0.00160651,
       0.00281734, 0.00323203, 0.00193474, 0.01192904, 0.00818507,
       0.00668169, 0.00811841, 0.00192768, 0.00302717, 0.00367362,
       0.00819972, 0.0039632 , 0.00389597, 0.00193123, 0.00381839,
       0.00311583, 0.00375742, 0.01163531, 0.00329315, 0.00386511,
       0.00363324, 0.01143476, 0.00375644, 0.00406183, 0.00395334,
       0.00390253, 0.00409195, 0.00748637, 0.00843297, 0.00426326,
       0.00409819, 0.00375328, 0.00337207, 0.00234114, 0.00356334,
       0.00386127, 0.00307698, 0.00368968, 0.00160235, 0.00297859,
       0.01211857, 0.00709824, 0.0039921 , 0.00261502, 0.00393049,
       0.0032861 , 0.01114771, 0.01020928, 0.01063606, 0.0027868 ,
       0.01045774, 0.00260124, 0.00154839, 0.00251186, 0.00281193,
       0.00399535, 0.00202947, 0.00297378, 0.0031609 , 0.00374543,
       0.00256887, 0.00262858, 0.00367052, 0.00371996, 0.00330418,
       0.00326767, 0.00347829, 0.00412538, 0.004063  , 0.00347878,
       0.01253573, 0.00442786, 0.00386127, 0.00313026, 0.01119107,
       0.01192904, 0.00622656, 0.00358372, 0.00382706, 0.01138947,
       0.00347308, 0.00181484, 0.01086708, 0.00352611, 0.00271463,
       0.00589394, 0.00303631, 0.00400982, 0.00377865, 0.00272063,
       0.00397087, 0.01063495, 0.00172006, 0.00139274, 0.01211857,
       0.00385998, 0.00382622, 0.00276962, 0.0036893 , 0.00287509,
```

```
0.00331733, 0.01119107, 0.00247816, 0.00901803, 0.00355445,
0.00284556, 0.00247073, 0.00739079, 0.00385014, 0.00405102,
0.00231681, 0.00346085, 0.01061802, 0.00325816, 0.00535199,
0.0032561 , 0.0028696 , 0.00889089, 0.0037197 , 0.01167532,
0.00872058, 0.00421804, 0.00165302, 0.00422237, 0.00341956,
0.00260314, 0.01088185, 0.00252159, 0.00380342, 0.00287363,
0.00905023, 0.00377161, 0.00409195, 0.01129274, 0.00354895,
0.00345177, 0.00340351, 0.00385781, 0.0069115 , 0.0040011 ,
0.00231112, 0.01032239, 0.00325576, 0.00347078, 0.00358387,
0.00215412, 0.00867153, 0.00201578, 0.00284118, 0.00246355,
0.0041833 , 0.00368229, 0.01215939, 0.00494756, 0.00872203,
0.00242668, 0.00394467, 0.01238621, 0.01113286, 0.01167532,
0.00794232, 0.00368802, 0.00405972, 0.01033033, 0.00400732,
0.00340279, 0.00337131, 0.00368989, 0.00371361, 0.01139514,
0.00314854, 0.01113317, 0.00769975, 0.00944234, 0.00343343,
0.00387602, 0.00332257, 0.00363648, 0.00219793, 0.00302444,
0.00816752, 0.00253343, 0.00353349, 0.00291908, 0.00229916,
0.00998748, 0.00366023, 0.00197514])
```

```
1 logreg.predict_proba(X_test)[:,1]
```

```
array([0.00506526, 0.00831363, 0.01133216, 0.00511732, 0.01312885,
       0.00628249, 0.01277106, 0.00327126, 0.00802251, 0.00329502,
       0.00829591, 0.00825   , 0.00483418, 0.00475953, 0.00791952,
       0.01395741, 0.00801414, 0.01201613, 0.00299856, 0.00505894,
       0.01403951, 0.00614599, 0.00282286, 0.00986657, 0.00794252,
       0.00459406, 0.01029969, 0.01427534, 0.00676839, 0.00774996,
       0.00758795, 0.00774025, 0.0037773 , 0.00760526, 0.00774599,
       0.00668483, 0.00712003, 0.0029705 , 0.00762487, 0.00852288,
       0.01331499, 0.00428484, 0.00453113, 0.00854162, 0.01016459,
       0.00796022, 0.00782504, 0.0042218 , 0.00654752, 0.00648468,
       0.00572759, 0.00791082, 0.00541788, 0.00760288, 0.01126262,
       0.00730486, 0.00762487, 0.0142366 , 0.00383479, 0.00951155,
       0.00610509, 0.00770442, 0.00750834, 0.00407335, 0.00785004,
       0.00536282, 0.00761552, 0.00537021, 0.0082452 , 0.00264583,
       0.00588868, 0.00588641, 0.00371077, 0.01416907, 0.00874983,
       0.00732669, 0.00881835, 0.00324628, 0.00665879, 0.00789032,
       0.00875446, 0.00813072, 0.00807307, 0.00363125, 0.00799555,
       0.00625326, 0.00712481, 0.01421858, 0.00654085, 0.00790305,
       0.00715968, 0.0134906 , 0.00747273, 0.00826188, 0.00791843,
       0.00767994, 0.00818425, 0.00812101, 0.00919319, 0.00896873,
       0.00816824, 0.00746484, 0.0075591 , 0.00489178, 0.00762601,
       0.00792279, 0.00711529, 0.00758282, 0.00251099, 0.00560808,
       0.01427534, 0.00730102, 0.00831633, 0.00502763, 0.00790184,
       0.00612761, 0.01254777, 0.01338658, 0.01256611, 0.00509296,
       0.01328321, 0.00457991, 0.00229808, 0.00464869, 0.00547328,
       0.0084214 , 0.00319268, 0.00542808, 0.00620055, 0.00757583,
       0.00516929, 0.00498859, 0.00712622, 0.00742809, 0.00681879,
       0.0070041 , 0.00649831, 0.00836625, 0.00613904, 0.00754386,
       0.0142366 , 0.00407309, 0.00792279, 0.00690929, 0.01274492,
       0.01416907, 0.00604704, 0.00736218, 0.00764123, 0.01387179,
       0.00749749, 0.0028124 , 0.0131464 , 0.00791689, 0.00492715,
       0.00604165, 0.0061462 , 0.007823  , 0.00727651, 0.00595695,
       0.00790514, 0.012713  , 0.00285238, 0.00206033, 0.01427534,
       0.00792824, 0.00773152, 0.00494495, 0.00803479, 0.00705039,
       0.00611255, 0.01274492, 0.00456316, 0.00971078, 0.00721604,
       0.00504609, 0.00402121, 0.00742429, 0.00825582, 0.00836756,
       0.00436999, 0.006563  , 0.01239852, 0.00664143, 0.00524587,
       0.00706436, 0.00521485, 0.01049036, 0.00741639, 0.01323309,
       0.00874256, 0.00828443, 0.00275227, 0.00857369, 0.00720588,
       0.00529287, 0.01331928, 0.00481397, 0.00774532, 0.00521352,
       0.00985906, 0.00803079, 0.00818425, 0.01403381, 0.0075455 ,
       0.00677569, 0.00742197, 0.00847059, 0.00748735, 0.00825633,
       0.00444617, 0.01254855, 0.00662291, 0.00703836, 0.00736605,
       0.00382976, 0.0100292 , 0.00383995, 0.00491749, 0.00503272,
       0.00839919, 0.00747692, 0.01449089, 0.00472945, 0.00931414,
       0.00480535, 0.00848652, 0.01418315, 0.01293739, 0.01323309,
       0.0083658 , 0.00690966, 0.00828643, 0.01164182, 0.00777119,
       0.00814893, 0.00689767, 0.00760981, 0.00774996, 0.0132877 ,
       0.00619931, 0.01351407, 0.00831133, 0.01051104, 0.00700143,
       0.00799463, 0.00783646, 0.00755754, 0.0039613 , 0.00585141,
       0.00977211, 0.0047226 , 0.0028935 , 0.00644094, 0.00392189,
       0.01181859, 0.00743373, 0.00394042])
```

Check accuracy score

```
1 from sklearn.metrics import accuracy_score
2
3 print ('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))
```

```
Model accuracy score: 0.0543
```

```
1 y_pred_train = logreg.predict(X_train)
2
3 y_pred_train
```

```
array([25, 23, 21, 19, 18, 18, 24, 31, 40, 35, 19, 38, 19, 19, 24, 18, 19,
       28, 34, 18, 34, 18, 23, 18, 19, 18, 19, 23, 35, 35, 21, 19, 26, 19,
       24, 19, 30, 23, 36, 23, 19, 18, 19, 36, 30, 35, 34, 19, 19, 19, 23,
       18, 18, 19, 24, 18, 19, 18, 19, 23, 19, 35, 18, 19, 18, 21, 18, 27,
       19, 18, 35, 30, 18, 18, 40, 19, 30, 24, 24, 23, 18, 23, 23, 19, 19,
       27, 18, 18, 18, 19, 24, 18, 18, 23, 27, 23, 23, 18, 24, 20, 27, 24,
       19, 27, 41, 18, 19, 30, 30, 19, 23, 19, 35, 18, 24, 19, 20, 18, 18,
       24, 23, 18, 19, 30, 18, 19, 27, 19, 19, 17, 19, 21, 19, 25, 19, 37,
       35, 19, 26, 19, 35, 19, 36, 19, 31, 18, 24, 19, 24, 19, 18, 21, 18,
       34, 20, 19, 19, 23, 18, 18, 23, 17, 18, 23, 19, 30, 19, 19, 19, 19,
       27, 27, 23, 18, 18, 24, 23, 23, 19, 19, 24, 18, 23, 35, 19, 18, 18,
       28, 18, 34, 20, 31, 19, 19, 20, 19, 19, 19, 19, 18, 18, 18, 19, 19,
       18, 23, 23, 30, 35, 18, 19, 18, 18, 21, 24, 34, 21, 33, 18, 19, 19,
       19, 19, 30, 26, 18, 24, 18, 18, 18, 37, 35, 25, 35, 18, 24, 19, 18,
       19, 19, 18, 19, 18, 24, 19, 23, 23, 18, 19, 27, 19, 19, 30, 23, 34,
       23, 18, 35, 18, 23, 19, 19, 18, 18, 19, 19, 18, 26, 30, 19, 19, 19,
       26, 18, 18, 23, 20, 28, 24, 34, 23, 19, 27, 19, 19, 19, 33, 38, 18,
       18, 35, 19, 26, 24, 20, 18, 30, 19, 18, 19, 18, 35, 18, 23, 23, 19,
       19, 24, 18, 18, 34, 18, 23, 19, 19, 18, 21, 20, 18, 21, 19, 24, 19,
       19, 19, 19, 19, 35, 18, 35, 18, 19, 18, 35, 23, 35, 19, 19, 41, 24,
       19, 24, 18, 19, 19, 28, 23, 35, 19, 19, 19, 23, 21, 19, 23, 19, 23,
       18, 19, 19, 28, 19, 23, 18, 26, 24, 23, 34, 24, 18, 38, 19, 18, 19,
       35, 18, 30, 18, 34, 24, 18, 24, 19, 18, 28, 19, 19, 23, 24, 23, 23,
       19, 23, 19, 26, 19, 19, 19, 19, 20, 19, 19, 18, 23, 18, 19, 19, 24,
       23, 24, 18, 19, 21, 18, 37, 18, 27, 18, 26, 19, 19, 35, 25, 19, 19,
       30, 19, 18, 27, 35, 19, 19, 19, 35, 18, 19, 23, 28, 36, 19, 18, 18,
       19, 23, 23, 19, 18, 19, 23, 21, 19, 18, 18, 19, 18, 19, 19, 41, 26,
       20, 19, 19, 23, 24, 20, 19, 18, 23, 18, 19, 18, 18, 21, 18, 19, 18,
       18, 41, 23, 19, 25, 20, 18, 18, 24, 18, 18, 18, 19, 23, 18, 19, 19,
       19, 26, 38, 34, 19, 26, 19, 19, 19, 27, 18, 19, 31, 18, 17, 18, 18,
       20, 18, 34, 25, 19, 23, 18, 30, 19, 21, 19, 34, 21, 25, 18, 18, 19,
       18, 23, 26, 19, 19, 30, 18, 19, 19, 38, 18, 28, 19, 34, 30, 19, 35,
       20, 19, 19, 18, 18, 19, 19, 23, 18, 18, 35, 18, 18, 19, 19, 24, 23,
       19, 20, 23, 18, 34, 18, 20, 23, 20, 28, 18, 21, 19, 19, 17, 41, 23,
       19, 23, 24, 19, 27, 18, 27, 19, 18, 18, 19, 19, 24, 23, 19, 18, 26,
       20, 18, 24, 35, 19])
```

Confusion matrix

```
1 from sklearn.metrics import confusion_matrix
2
3 cm = confusion_matrix(y_test, y_pred_test)
4
5 print('Confusion matrix\n\n', cm)
6
7 print('\nTrue Positives(TP) =', cm[0,0])
8
9 print('\nTrue Negatives(TN) =', cm[1,1])
10
11 print('\nFalse Positives(FP) =', cm[0,1])
12
13 print('nFalse Negatives(FN) =', cm[1,0])
```

```
Confusion matrix
```

```
 [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]

True Positives(TP) = 0

True Negatives(TN) = 0

False Positives(FP) = 0
nFalse Negatives(FN) = 0
```

Classification metrics

```python
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred_test))
```

```
              precision    recall  f1-score   support

          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          16       0.00      0.00      0.00         9
          17       0.33      0.08      0.13        12
          18       0.04      0.18      0.06        11
          19       0.06      0.42      0.11        12
          20       0.00      0.00      0.00        12
          21       0.17      0.13      0.15        15
          22       0.00      0.00      0.00        10
          23       0.05      0.06      0.05        17
          24       0.07      0.11      0.08         9
          25       0.10      0.07      0.08        15
          26       0.00      0.00      0.00        12
          27       0.00      0.00      0.00        10
          28       0.00      0.00      0.00        15
          29       0.00      0.00      0.00        17
          30       0.00      0.00      0.00        10
          31       0.00      0.00      0.00         8
          32       0.00      0.00      0.00         4
          33       0.00      0.00      0.00        10
          34       0.00      0.00      0.00         4
          35       0.14      0.14      0.14         7
          36       0.00      0.00      0.00         6
          37       0.00      0.00      0.00         5
          38       0.00      0.00      0.00         4
          39       0.00      0.00      0.00         4
          40       0.00      0.00      0.00         3
          41       0.00      0.00      0.00         3
          42       0.00      0.00      0.00         1
          44       0.00      0.00      0.00         2
          45       0.00      0.00      0.00         3
          50       0.00      0.00      0.00         1
          51       0.00      0.00      0.00         1
          59       0.00      0.00      0.00         1

    accuracy                           0.05       258
   macro avg       0.03      0.04      0.02       258
weighted avg       0.04      0.05      0.04       258

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to c
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to c
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to c
  _warn_prf(average, modifier, msg_start, len(result))
```

```python
TP = cm[0, 0]
TN = cm[1, 1]
FP = cm[0, 1]
FN = cm[1, 0]

classification_accuracy = (TP + TN) / float(TP + TN + FP + FN)

print('Classification accuracy: {:.4f}'.format(classification_accuracy))
```

```
Classification accuracy: nan
<ipython-input-56-b5213b263dc3>:6: RuntimeWarning: invalid value encountered in divide
  classification_accuracy = (TP + TN) / float(TP + TN + FP + FN)
```

```python
precision = TP / float(TP + FP)

print('Precision : {0:0.4f}'.format(precision))
```

```
Precision : nan
<ipython-input-58-de8ba741fd3c>:1: RuntimeWarning: invalid value encountered in divide
  precision = TP / float(TP + FP)
```

```python
recall = TP / float(TP + FN)

print('Recall or Sensitivity : {0:0.4f}'.format(recall))
```

```
Recall or Sensitivity : nan
<ipython-input-60-4541e477f8ab>:1: RuntimeWarning: invalid value encountered in divide
  recall = TP / float(TP + FN)
```

```python
true_positive_rate = TP / float(TP + FN)

print('True Positive Rate : {0:0.4f}'.format(true_positive_rate))
```

```
True Positive Rate : nan
<ipython-input-61-934b41082672>:1: RuntimeWarning: invalid value encountered in divide
  true_positive_rate = TP / float(TP + FN)
```

```python
false_positive_rate = FP / float(FP + TN)

print('False Positive Rate : {0:0.4f}'.format(false_positive_rate))
```

```
False Positive Rate : nan
<ipython-input-62-d365adb81363>:1: RuntimeWarning: invalid value encountered in divide
  false_positive_rate = FP / float(FP + TN)
```

```python
specifity = TN / (TN + FP)

print('Specifity : {0:0.4f}'.format(specifity))
```

```
Specifity : nan
<ipython-input-63-9df300ba775b>:1: RuntimeWarning: invalid value encountered in scalar divide
  specifity = TN / (TN + FP)
```

Adjusting the threshold level

```python
y_pred_prob = logreg.predict_proba(X_test)[0:10]

y_pred_prob
```

```
 0.0470696 , 0.03358698, 0.03607685, 0.03287692, 0.05027265,
 0.02572361, 0.02362664, 0.03158231, 0.02311375, 0.01813953,
 0.02466383, 0.03207744, 0.03229352, 0.02612535, 0.02346675,
 0.01007846, 0.01659781, 0.0179266 , 0.01700269, 0.01485147,
 0.01527205, 0.01748044, 0.01071097, 0.01450441, 0.00835209,
 0.00892269, 0.01162729, 0.01034523, 0.01326837, 0.01113003,
 0.01041808],
[0.00301097, 0.00628249, 0.01467561, 0.0128878 , 0.01521758,
 0.02886378, 0.04804109, 0.04420596, 0.03764325, 0.03666953,
 0.05876453, 0.05601355, 0.06101961, 0.06727193, 0.04954531,
 0.04632431, 0.02445255, 0.03923192, 0.03494606, 0.03392672,
 0.03827367, 0.02986914, 0.0435277 , 0.03178577, 0.02537132,
 0.0076255 , 0.01075573, 0.01301053, 0.01420212, 0.00842915,
 0.00743243, 0.01158656, 0.00744534, 0.0050201 , 0.00371695,
 0.00469437, 0.00435505, 0.00390935, 0.00377208, 0.00320467,
 0.0030179 ],
[0.0108993 , 0.01277106, 0.02649481, 0.02995219, 0.04992006,
 0.06329829, 0.0253621 , 0.04745813, 0.04782545, 0.02936823,
 0.04891048, 0.0341762 , 0.02960972, 0.02994064, 0.04719404,
 0.02321735, 0.02139433, 0.03203769, 0.02066678, 0.01758341,
 0.02728984, 0.03226241, 0.02782497, 0.0265793 , 0.02301641,
 0.010665  , 0.01676969, 0.02027387, 0.01714669, 0.01504276,
 0.01499303, 0.01869575, 0.01074177, 0.01523498, 0.00865332,
 0.00895877, 0.0116124 , 0.01026935, 0.01385533, 0.01135745,
 0.01067668],
[0.00401071, 0.00327126, 0.00528848, 0.00614952, 0.05378486,
 0.14014187, 0.02565009, 0.15199161, 0.06373131, 0.01210715,
 0.05115941, 0.01002655, 0.01397046, 0.00669747, 0.00356197,
 0.05350392, 0.01951351, 0.01995641, 0.01159611, 0.00232678,
 0.04931752, 0.01271681, 0.03766907, 0.01686448, 0.02662826,
 0.00943711, 0.00537729, 0.06478945, 0.01140073, 0.02396362,
 0.00420892, 0.00449774, 0.00248091, 0.00401404, 0.0090517 ,
 0.01364493, 0.00579792, 0.01539647, 0.00770197, 0.00410186,
 0.0124997 ],
[0.00698407, 0.00802251, 0.0067231 , 0.02747781, 0.0329448 ,
 0.03378263, 0.01925299, 0.02757315, 0.11100597, 0.05219863,
 0.017278  , 0.01827968, 0.03317279, 0.01229916, 0.04051347,
 0.05073459, 0.01921546, 0.0236776 , 0.00834538, 0.00694485,
 0.04629445, 0.03121206, 0.05761897, 0.01050183, 0.05823635,
 0.03258692, 0.01788441, 0.03045571, 0.01289743, 0.00779571,
 0.00946373, 0.01121243, 0.00654281, 0.00963927, 0.0182577 ,
 0.02026957, 0.00770775, 0.02232498, 0.00960439, 0.00792822,
 0.01513867],
[0.00197939, 0.00329502, 0.00431781, 0.00446281, 0.00371306,
 0.0041456 , 0.00595496, 0.00769956, 0.01118166, 0.02451003,
 0.04094198, 0.05554524, 0.0140117 , 0.0180278 , 0.01179027,
 0.05600174, 0.02184583, 0.0770507 , 0.02227597, 0.0424647 ,
 0.04502056, 0.06415137, 0.06103282, 0.06529298, 0.03424272,
 0.03691899, 0.01936133, 0.05971053, 0.09917713, 0.01321761,
 0.02144522, 0.00590487, 0.0105633 , 0.00458077, 0.00430564,
```

Conclusion

In this exercise, we used a confusion matrix to evaluate the performance of a classification model. Even though we faced difficulties and had limited time, we learned important lessons. True positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) were all counted in the confusion matrix. To evaluate the efficacy of the model, we computed important metrics such as recall (sensitivity) and classification accuracy. But due to time constraints, certain scheduled tasks—like visualizing the confusion matrix—were not finished. In order to improve predicted accuracy and reliability going forward, more model improvement and modification are advised. The significance of iterative model evaluation and improvement in machine learning is emphasized by this work.