Hands-on Activity 11.2 Classification using Logistic Regression

Name: Cuadra, Audrick Zander G.

Section: CPE22S3

Date: April 24, 2024

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.

```
# Install the ucimlrepo package
!pip install ucimlrepo
```

```
Requirement already satisfied: ucimlrepo in /usr/local/lib/python3.10/dist-packages (0.0.6)
```

```
# Import the dataset into your code
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)
```

```
    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
```

```
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
```

```
# display of the X dataframe
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	S s diagn
0	18	4.0	15.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	
1	15	1.0	14.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	
2	34	1.0	NaN	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	
3	52	5.0	16.0	4.0	1.0	37.0	37.0	1.0	3.0	0.0	...	NaN	
4	46	3.0	21.0	4.0	0.0	0.0	0.0	1.0	15.0	0.0	...	NaN	

5 rows × 36 columns

```
# display of the y dataframe
y
```

```
# importing of necessary libraries
import pandas as pd
import numpy as np

# concatinating the two dataframes
cercan_df = pd.concat([X, y], axis=1)
```

```
cercan_df.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	S s diagn
0	18	4.0	15.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	
1	15	1.0	14.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	
2	34	1.0	NaN	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	NaN	
3	52	5.0	16.0	4.0	1.0	37.0	37.0	1.0	3.0	0.0	...	NaN	
4	46	3.0	21.0	4.0	0.0	0.0	0.0	1.0	15.0	0.0	...	NaN	

5 rows × 36 columns

```
# creating a function that checks for duplicates
def cdup(data):
    if data.duplicated().any():
        return data.duplicated().sum()
    else:
        return "No Duplicates Found!"
```

```
# checking for duplicates
cdup(cercan_df)
```

```
23
```

```
# dropping all the duplicates in the dataframe
cercan_df = cercan_df.drop_duplicates()
```

```
# confirming if all the duplicates have been dropped
cdup(cercan_df)
```

```
'No Duplicates Found!'
```

```
# checking the nulls
cercan_df.isnull().sum()
```

```
Age                                   0
Number of sexual partners            25
First sexual intercourse              7
Num of pregnancies                   56
Smokes                               13
Smokes (years)                       13
Smokes (packs/year)                  13
Hormonal Contraceptives             103
Hormonal Contraceptives (years)     103
IUD                                 112
IUD (years)                         112
STDs                                100
STDs (number)                       100
STDs:condylomatosis                 100
STDs:cervical condylomatosis        100
STDs:vaginal condylomatosis         100
STDs:vulvo-perineal condylomatosis   100
STDs:syphilis                       100
STDs:pelvic inflammatory disease    100
STDs:genital herpes                 100
STDs:molluscum contagiosum          100
STDs:AIDS                           100
STDs:HIV                            100
STDs:Hepatitis B                    100
STDs:HPV                            100
STDs: Number of diagnosis             0
STDs: Time since first diagnosis    764
STDs: Time since last diagnosis     764
Dx:Cancer                             0
Dx:CIN                                0
Dx:HPV                                0
Dx                                    0
Hinselmann                            0
Schiller                              0
Citology                              0
Biopsy                                0
dtype: int64
```

```
# creating a function that detects if the column contains an outlier
def col_outliers(data):
    col_outliers=[]
    for i in data.columns:
        if data[i].isnull().any():
            low_bound = data[i].quantile(0.25) - (1.5 * (data[i].quantile(0.75) - data[i].quantile(0.25)))
            upper_bound = data[i].quantile(0.75) + (1.5 * (data[i].quantile(0.75) - data[i].quantile(0.25)))
            if ((data[i] < low_bound) |  (data[i] > upper_bound)).any():
                col_outliers.append(i)
    return col_outliers
```

```
col_outliers(cercan_df)
```

```
['Number of sexual partners',
 'First sexual intercourse',
 'Num of pregnancies',
 'Smokes',
 'Smokes (years)',
 'Smokes (packs/year)',
 'Hormonal Contraceptives (years)',
 'IUD',
```

```
'IUD (years)',
'STDs',
'STDs (number)',
'STDs:condylomatosis',
'STDs:vaginal condylomatosis',
'STDs:vulvo-perineal condylomatosis',
'STDs:syphilis',
'STDs:pelvic inflammatory disease',
'STDs:genital herpes',
'STDs:molluscum contagiosum',
'STDs:HIV',
'STDs:Hepatitis B',
'STDs:HPV',
'STDs: Time since first diagnosis',
'STDs: Time since last diagnosis']
```

```
# creating a function that would fill in all the nulls
def fnullsmedian(data):
    for i in data.columns:
        if data[i].isnull().any():
            low_bound = data[i].quantile(0.25) - (1.5 * (data[i].quantile(0.75) - data[i].quantile(0.25)))
            upper_bound = data[i].quantile(0.75) + (1.5 * (data[i].quantile(0.75) - data[i].quantile(0.25)))
            if ((data[i] < low_bound) |  (data[i] > upper_bound)).any():
                data[i].fillna(data[i].median(), inplace=True)
```

```
# filling the nulls
fnullsmedian(cercan_df)
```

```
# checking if all the nulls have been dropped
cercan_df.isnull().sum()
```

```
Age                                   0
Number of sexual partners             0
First sexual intercourse              0
Num of pregnancies                    0
Smokes                                0
Smokes (years)                        0
Smokes (packs/year)                   0
Hormonal Contraceptives             103
Hormonal Contraceptives (years)       0
IUD                                   0
IUD (years)                           0
STDs                                  0
STDs (number)                         0
STDs:condylomatosis                   0
STDs:cervical condylomatosis        100
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                           100
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
```

```
# creating a function that fills the rest of columns with nulls with mean
def fnullsmean(data):
    for i in data.columns:
        if data[i].isnull().any():
                data[i].fillna(data[i].mean(), inplace=True)
```

```
fnullsmean(cercan_df)
```

```
# checking if all nulls have been dropped
cercan_df.isnull().sum()
```

```
Age                                 0
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
```

```
def undscore(data):
    for i in data.columns:
        data.columns = [i.replace(' ', '_') for i in data.columns]
        data.columns = [i.strip('_') for i in data.columns]
    return data.columns
```

```
undscore(cercan_df)
```

```
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')
```

```
# checking the dataframe
cercan_df.head()
```

	Age	Number_of_sexual_partners	First_sexual_intercourse	Num_of_pregnancies	Smok
0	18	4.0	15.0	1.0	
1	15	1.0	14.0	1.0	
2	34	1.0	17.0	1.0	
3	52	5.0	16.0	4.0	
4	46	3.0	21.0	4.0	

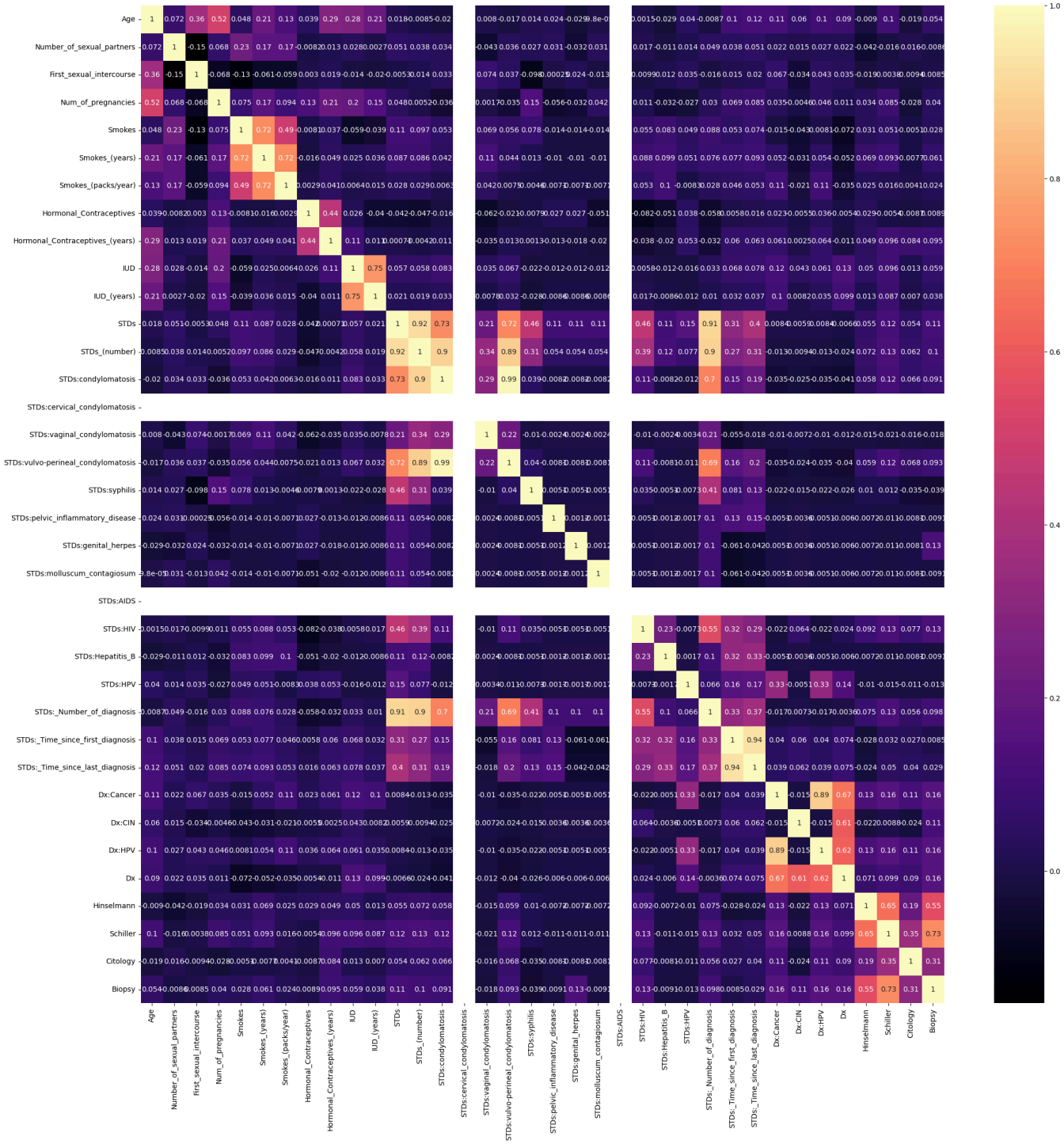
5 rows × 36 columns

```
# creating a heatmap
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(25,25))

sns.heatmap(cercan_df.corr(), annot=True, cmap='magma')
```

<Axes: >



Logistic Regresson

```
1 import pandas as pd
2 import numpy as np
3 import seaborn as sns
4 import matplotlib.pyplot as plt
5 %matplotlib inline
```