Student Name: Dina Chowdhury

Student ID: 500786553

Toronto Metropolitan University

CIND860 DAH - Advanced Data Analytics Project - P2025

Supervisor: Dr. Ashok Bhowmick

Descriptive Statistics

Importing libraries

```
import os
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from nbconvert import HTMLExporter
import nbformat
```

Load the data file

```
# Mount google drive to access the file
from google.colab import drive
drive.mount('/content/drive')

# Import the data file
file_path = '/content/drive/MyDrive/Colab Notebooks/diabetes_binary_5050split_health_indicators_BRFSS2015.csv'
os.chdir(os.path.dirname(file_path))
```

```
Mounted at /content/drive
```

```
# Load the data file
data = pd.read_csv(file_path)
print(data)
```

```
       Diabetes_binary  HighBP  HighChol  CholCheck   BMI  Smoker  Stroke  \
0                  0.0     1.0       0.0        1.0  26.0     0.0     0.0
1                  0.0     1.0       1.0        1.0  26.0     1.0     1.0
2                  0.0     0.0       0.0        1.0  26.0     0.0     0.0
3                  0.0     1.0       1.0        1.0  28.0     1.0     0.0
4                  0.0     0.0       0.0        1.0  29.0     1.0     0.0
...                ...     ...       ...        ...   ...     ...     ...
70687              1.0     0.0       1.0        1.0  37.0     0.0     0.0
70688              1.0     0.0       1.0        1.0  29.0     1.0     0.0
70689              1.0     1.0       1.0        1.0  25.0     0.0     0.0
70690              1.0     1.0       1.0        1.0  18.0     0.0     0.0
70691              1.0     1.0       1.0        1.0  25.0     0.0     0.0

       HeartDiseaseorAttack  PhysActivity  Fruits  ...  AnyHealthcare  \
0                       0.0           1.0     0.0  ...            1.0
1                       0.0           0.0     1.0  ...            1.0
2                       0.0           1.0     1.0  ...            1.0
3                       0.0           1.0     1.0  ...            1.0
4                       0.0           1.0     1.0  ...            1.0
...                     ...           ...     ...  ...            ...
70687                   0.0           0.0     0.0  ...            1.0
70688                   1.0           0.0     1.0  ...            1.0
70689                   1.0           0.0     1.0  ...            1.0
70690                   0.0           0.0     0.0  ...            1.0
70691                   1.0           1.0     1.0  ...            1.0

       NoDocbcCost  GenHlth  MentHlth  PhysHlth  DiffWalk  Sex   Age  \
0              0.0      3.0       5.0      30.0       0.0  1.0   4.0
1              0.0      3.0       0.0       0.0       0.0  1.0  12.0
2              0.0      1.0       0.0      10.0       0.0  1.0  13.0
3              0.0      3.0       0.0       3.0       0.0  1.0  11.0
4              0.0      2.0       0.0       0.0       0.0  0.0   8.0
...            ...      ...       ...       ...       ...  ...   ...
70687          0.0      4.0       0.0       0.0       0.0  0.0   6.0
70688          0.0      2.0       0.0       0.0       1.0  1.0  10.0
70689          0.0      5.0      15.0       0.0       1.0  0.0  13.0
70690          0.0      4.0       0.0       0.0       1.0  0.0  11.0
70691          0.0      2.0       0.0       0.0       0.0  0.0   9.0

       Education  Income
0            6.0     8.0
1            6.0     8.0
2            6.0     8.0
3            6.0     8.0
4            5.0     8.0
...          ...     ...
70687        4.0     1.0
70688        3.0     6.0
70689        6.0     4.0
70690        2.0     4.0
70691        6.0     2.0
```

```
[70692 rows x 22 columns]
```

```
# Summary of data frame
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70692 entries, 0 to 70691
Data columns (total 22 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Diabetes_binary       70692 non-null  float64
 1   HighBP                70692 non-null  float64
 2   HighChol              70692 non-null  float64
 3   CholCheck             70692 non-null  float64
 4   BMI                   70692 non-null  float64
 5   Smoker                70692 non-null  float64
 6   Stroke                70692 non-null  float64
 7   HeartDiseaseorAttack  70692 non-null  float64
 8   PhysActivity          70692 non-null  float64
 9   Fruits                70692 non-null  float64
 10  Veggies               70692 non-null  float64
 11  HvyAlcoholConsump     70692 non-null  float64
 12  AnyHealthcare         70692 non-null  float64
 13  NoDocbcCost           70692 non-null  float64
 14  GenHlth               70692 non-null  float64
 15  MentHlth              70692 non-null  float64
 16  PhysHlth              70692 non-null  float64
 17  DiffWalk              70692 non-null  float64
 18  Sex                   70692 non-null  float64
 19  Age                   70692 non-null  float64
 20  Education             70692 non-null  float64
 21  Income                70692 non-null  float64
dtypes: float64(22)
memory usage: 11.9 MB
```

```
# Check for missing value
data.isna().sum()
```

	0
Diabetes_binary	0
HighBP	0
HighChol	0
CholCheck	0
BMI	0
Smoker	0
Stroke	0
HeartDiseaseorAttack	0
PhysActivity	0
Fruits	0
Veggies	0
HvyAlcoholConsump	0
AnyHealthcare	0
NoDocbcCost	0
GenHlth	0
MentHlth	0
PhysHlth	0
DiffWalk	0
Sex	0
Age	0
Education	0
Income	0

```
#Overall Summary

data.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Diabetes_binary	70692.0	0.500000	0.500004	0.0	0.0	0.5	1.0	1.0
HighBP	70692.0	0.563458	0.495960	0.0	0.0	1.0	1.0	1.0
HighChol	70692.0	0.525703	0.499342	0.0	0.0	1.0	1.0	1.0
CholCheck	70692.0	0.975259	0.155336	0.0	1.0	1.0	1.0	1.0
BMI	70692.0	29.856985	7.113954	12.0	25.0	29.0	33.0	98.0
Smoker	70692.0	0.475273	0.499392	0.0	0.0	0.0	1.0	1.0
Stroke	70692.0	0.062171	0.241468	0.0	0.0	0.0	0.0	1.0
HeartDiseaseorAttack	70692.0	0.147810	0.354914	0.0	0.0	0.0	0.0	1.0
PhysActivity	70692.0	0.703036	0.456924	0.0	0.0	1.0	1.0	1.0
Fruits	70692.0	0.611795	0.487345	0.0	0.0	1.0	1.0	1.0
Veggies	70692.0	0.788774	0.408181	0.0	1.0	1.0	1.0	1.0
HvyAlcoholConsump	70692.0	0.042721	0.202228	0.0	0.0	0.0	0.0	1.0
AnyHealthcare	70692.0	0.954960	0.207394	0.0	1.0	1.0	1.0	1.0
NoDocbcCost	70692.0	0.093914	0.291712	0.0	0.0	0.0	0.0	1.0
GenHlth	70692.0	2.837082	1.113565	1.0	2.0	3.0	4.0	5.0
MentHlth	70692.0	3.752037	8.155627	0.0	0.0	0.0	2.0	30.0
PhysHlth	70692.0	5.810417	10.062261	0.0	0.0	0.0	6.0	30.0
DiffWalk	70692.0	0.252730	0.434581	0.0	0.0	0.0	1.0	1.0
Sex	70692.0	0.456997	0.498151	0.0	0.0	0.0	1.0	1.0
Age	70692.0	8.584055	2.852153	1.0	7.0	9.0	11.0	13.0
Education	70692.0	4.920953	1.029081	1.0	4.0	5.0	6.0	6.0

```
# Separate numerical and categorical columns
numerical_cols = data.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_cols = data.select_dtypes(include=['object', 'category']).columns.tolist()


# Check the unique valuse in the columns
for col in data.columns:
```

```
    print(col, ":", data[col].nunique())
```

```
Diabetes_binary : 2
HighBP : 2
HighChol : 2
CholCheck : 2
BMI : 80
Smoker : 2
Stroke : 2
HeartDiseaseorAttack : 2
PhysActivity : 2
Fruits : 2
Veggies : 2
HvyAlcoholConsump : 2
AnyHealthcare : 2
NoDocbcCost : 2
GenHlth : 5
MentHlth : 31
PhysHlth : 31
DiffWalk : 2
Sex : 2
Age : 13
Education : 6
Income : 8
```

```
# Convert numerical to categorical variable
data['Diabetes_binary'] = data['Diabetes_binary'].astype('category')
data['HighBP'] = data['HighBP'].astype('category')
data['HighChol'] = data['HighChol'].astype('category')
data['CholCheck'] = data['CholCheck'].astype('category')
data['Smoker'] = data['Smoker'].astype('category')
data['Stroke'] = data['Stroke'].astype('category')
data['HeartDiseaseorAttack'] = data['HeartDiseaseorAttack'].astype('category')
data['PhysActivity'] = data['PhysActivity'].astype('category')
data['Fruits'] = data['Fruits'].astype('category')
data['Veggies'] = data['Veggies'].astype('category')
data['HvyAlcoholConsump'] = data['HvyAlcoholConsump'].astype('category')
data['AnyHealthcare'] = data['AnyHealthcare'].astype('category')
data['NoDocbcCost'] = data['NoDocbcCost'].astype('category')
data['GenHlth'] = data['GenHlth'].astype('category')
data['DiffWalk'] = data['DiffWalk'].astype('category')
data['Sex'] = data['Sex'].astype('category')
#data['Age'] = data['Age'].astype('category')
data['Education'] = data['Education'].astype('category')
data['Income'] = data['Income'].astype('category')
```

```
# Separate numerical and categorical columns
numerical_cols = data.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_cols = data.select_dtypes(include=['object', 'category']).columns.tolist()


print("Numerical columns:", numerical_cols)
print("Categorical columns:", categorical_cols)
```

```
Numerical columns: ['BMI', 'MentHlth', 'PhysHlth', 'Age']
Categorical columns: ['Diabetes_binary', 'HighBP', 'HighChol', 'CholCheck', 'Smoker', 'Stroke', 'HeartDiseaseorAttack', 'PhysActivity', 'Fruit
```

Double-click (or enter) to edit

```
#Overall Summary

data.describe().T
```

	count	mean	std	min	25%	50%	75%	max
BMI	70692.0	29.856985	7.113954	12.0	25.0	29.0	33.0	98.0
MentHlth	70692.0	3.752037	8.155627	0.0	0.0	0.0	2.0	30.0
PhysHlth	70692.0	5.810417	10.062261	0.0	0.0	0.0	6.0	30.0

```
# Categorical Feature Distribution
data['HighBP'].value_counts(normalize=True)
```

HighBP	proportion
1.0	0.563458
0.0	0.436542

```
# HighBP
highbp_percent = data['HighBP'].value_counts(normalize=True).get(1, 0) * 100
print(f"Percentage with High Blood Pressure: {highbp_percent:.2f}%")
```

```
Percentage with High Blood Pressure: 56.35%
```

```
data['HighChol'].value_counts(normalize=True)
```

HighChol	proportion
1.0	0.525703
0.0	0.474297

```
#HighChol
highchol_percent = data['HighChol'].value_counts(normalize=True).get(1, 0) * 100
print(f"Percentage with High Cholesterol: {highchol_percent:.2f}%")
```

```
Percentage with High Cholesterol: 52.57%
```

```
data['Smoker'].value_counts(normalize=True)
```

Smoker	proportion
0.0	0.524727
1.0	0.475273

```
data['Stroke'].value_counts(normalize=True)
```

Stroke	proportion
0.0	0.937829
1.0	0.062171

```
data['HeartDiseaseorAttack'].value_counts(normalize=True)
```

HeartDiseaseorAttack	proportion
0.0	0.85219
1.0	0.14781

```
data['PhysActivity'].value_counts(normalize=True)
```

PhysActivity	proportion
1.0	0.703036
0.0	0.296964

```
data['Fruits'].value_counts(normalize=True)
```

Fruits	proportion
1.0	0.611795
0.0	0.388205

```
data['Veggies'].value_counts(normalize=True)
```

Veggies	proportion
1.0	0.788774
0.0	0.211226

```
data['HvyAlcoholConsump'].value_counts(normalize=True)
```

HvyAlcoholConsump	proportion
0.0	0.957279
1.0	0.042721

```
data['AnyHealthcare'].value_counts(normalize=True)
```

AnyHealthcare	proportion
1.0	0.95496
0.0	0.04504

```
data['NoDocbcCost'].value_counts(normalize=True)
```

NoDocbcCost	proportion
0.0	0.906086
1.0	0.093914

```
data['GenHlth'].value_counts(normalize=True)
```

GenHlth	proportion
3.0	0.331395
2.0	0.281107
4.0	0.188183
1.0	0.117156
5.0	0.082159

```
data['DiffWalk'].value_counts(normalize=True)
```

DiffWalk	proportion
0.0	0.74727
1.0	0.25273

```
data['Sex'].value_counts(normalize=True)
```

Sex	proportion
0.0	0.543003
1.0	0.456997

```
data['Education'].value_counts(normalize=True)
```

Education	proportion
6.0	0.368076
5.0	0.283342
4.0	0.275463
3.0	0.048761
2.0	0.023298
1.0	0.001061

```
data['Income'].value_counts(normalize=True)
```

Income	proportion
8.0	0.292056
7.0	0.161617
6.0	0.145519
5.0	0.113308
4.0	0.094183
3.0	0.078609
2.0	0.063628
1.0	0.051081

```
#Income
income_percentage_lowest = data['Income'].value_counts(normalize=True).get(1, 0) * 100
print(f"Lowest income level percentage: {income_percentage_lowest:.2f}%")

income_percentage_highest = data['Income'].value_counts(normalize=True).get(8, 0) * 100
print(f"Highest income level percentage: {income_percentage_highest:.2f}%")
```

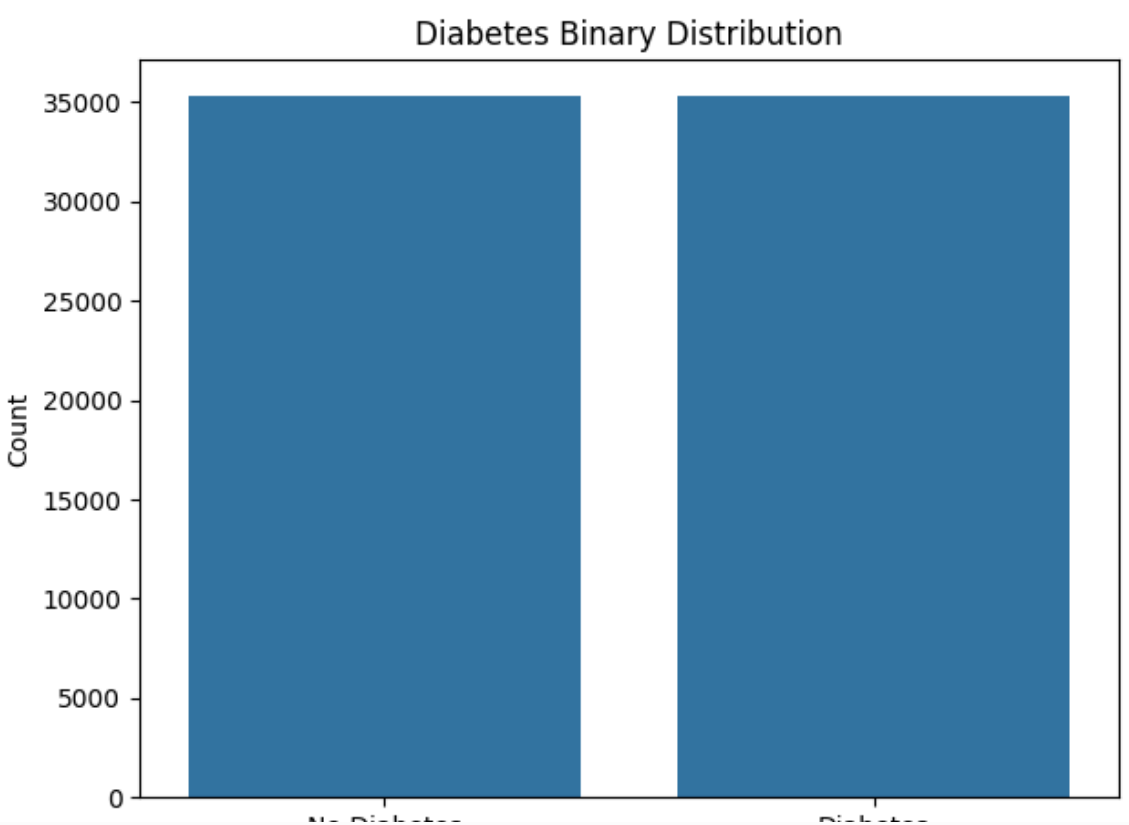
```
Lowest income level percentage: 5.11%
Highest income level percentage: 29.21%
```

```
#Diabetes_binary Distribution
data['Diabetes_binary'].value_counts(normalize=True)
```

Diabetes_binary	proportion
0.0	0.5
1.0	0.5

```
# Visualization
# Target variable distribution
```

```
sns.countplot(x='Diabetes_binary', data=data)
plt.title('Diabetes Binary Distribution')
plt.xticks([0,1], ['No Diabetes', 'Diabetes'])
plt.xlabel('')
plt.ylabel('Count')
plt.tight_layout()
plt.show()
```




```
# BMI distribution - Histogram
sns.histplot(data['BMI'], bins=30, kde=True)
plt.title('BMI Distribution')
plt.xlabel('BMI')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()

# BMI distribution - Boxplot
sns.boxplot(x='Diabetes_binary', y='BMI', data=data)
plt.xticks([0,1], ['No Diabetes', 'Diabetes'])
```

```
plt.title('BMI Distribution by Diabetes Status')
plt.xlabel('')
plt.tight_layout()
plt.show()
```