DSC 680 - Project 1 Code

September 20, 2024

1 DSC680 - Project 1

1.1 Does the number of pregnancies affect the glucose level in blood

[35]:
```
import pandas as pd
import scipy
from matplotlib import pyplot  as plt
import numpy as np
import math
from scipy.stats import pareto
from thinkstats2 import HypothesisTest
from numpy.random import default_rng
import thinkstats2
import thinkplot
```

[36]:
```
diabetes_df = pd.read_csv("diabetes.csv")
```

[37]:
```
diabetes_df.describe()
```

[37]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin
count	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479
std	3.369578	31.972618	19.355807	15.952218	115.244002
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000
75%	6.000000	140.250000	80.000000	32.000000	127.250000
max	17.000000	199.000000	122.000000	99.000000	846.000000

	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000
mean	31.992578	0.471876	33.240885	0.348958
std	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.078000	21.000000	0.000000
25%	27.300000	0.243750	24.000000	0.000000
50%	32.000000	0.372500	29.000000	0.000000
75%	36.600000	0.626250	41.000000	1.000000
max	67.100000	2.420000	81.000000	1.000000

[38]:
```
diabetes_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

2 Column Non-Null Count Dtype

---

0 Pregnancies 768 non-null int64
1 Glucose 768 non-null int64
2 BloodPressure 768 non-null int64
3 SkinThickness 768 non-null int64
4 Insulin 768 non-null int64
5 BMI 768 non-null float64 6 DiabetesPedigreeFunction 768 non-null float64 7 Age 768 non-null int64
8 Outcome 768 non-null int64
dtypes: float64(2), int64(7) memory usage: 54.1 KB

[39]:
```
diabetes_df
```

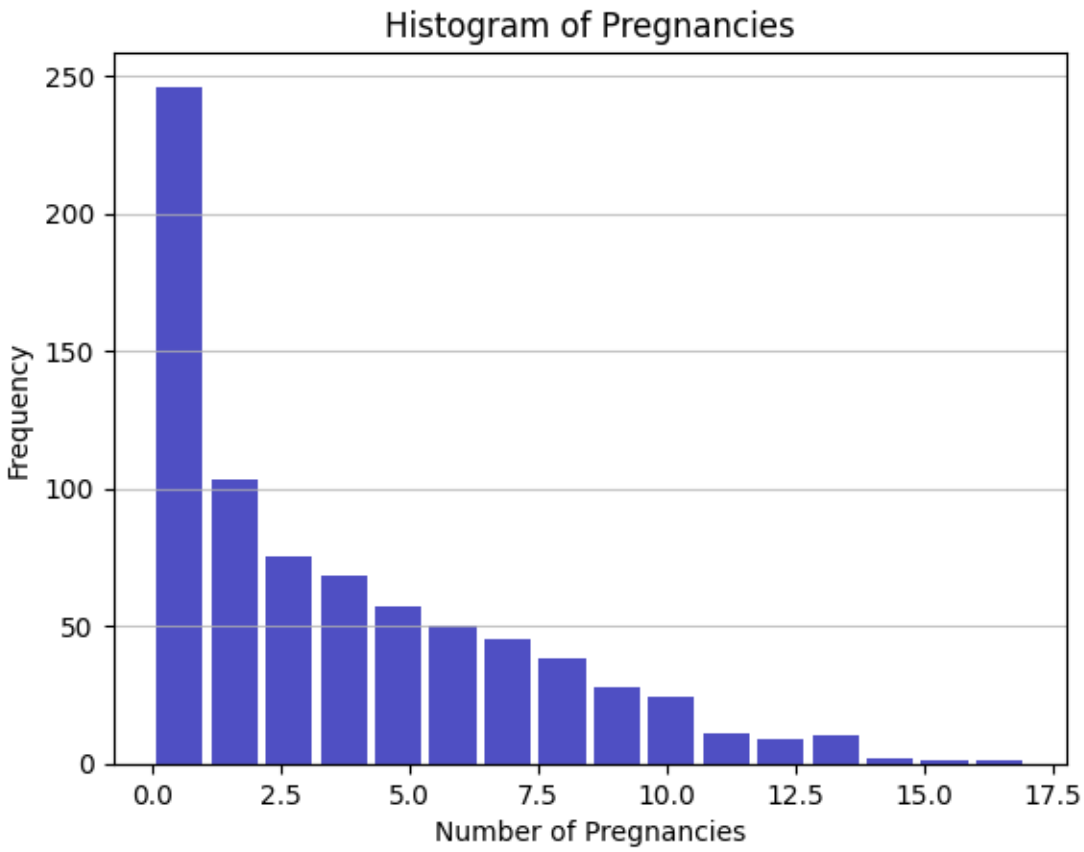
[39]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
..	…	…	…	…	…	…	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

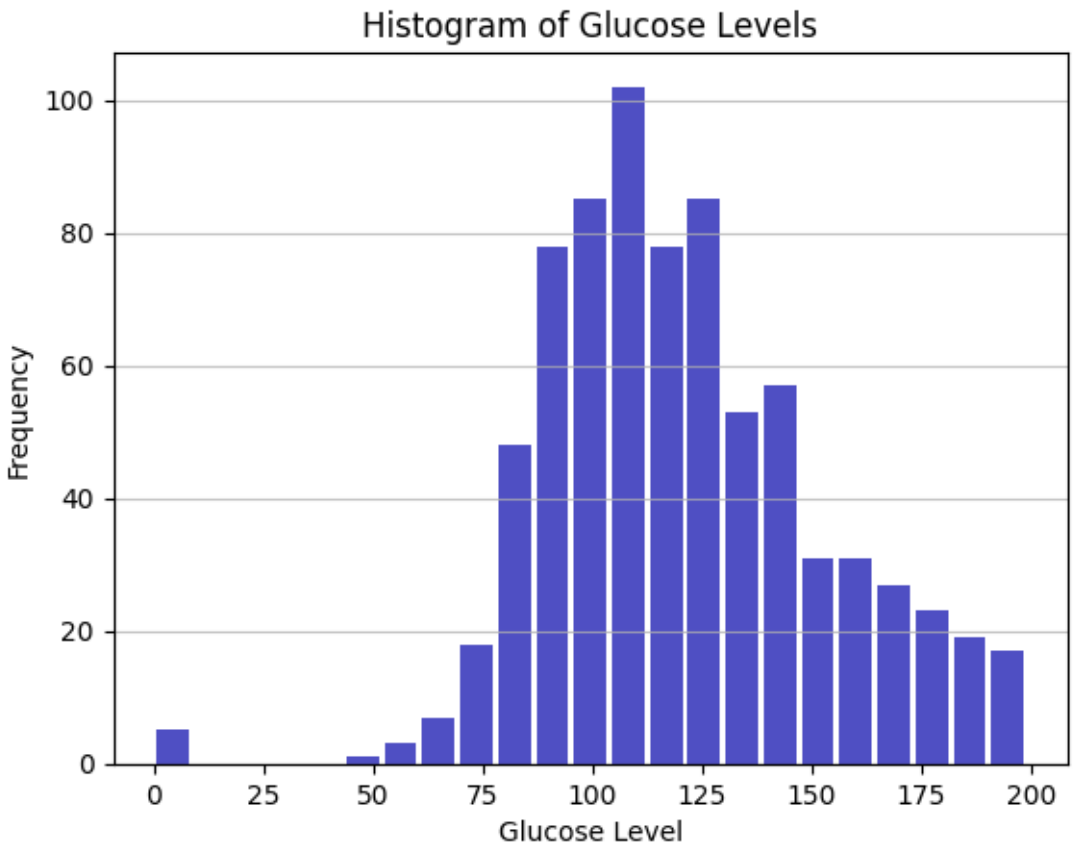
```
     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
1                       0.351   31        0
2                       0.672   32        1
3                       0.167   21        0
4                       2.288   33        1
..                        …  …       …
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0

[768 rows x 9 columns]
```

2.1 Histogram for 5 variables

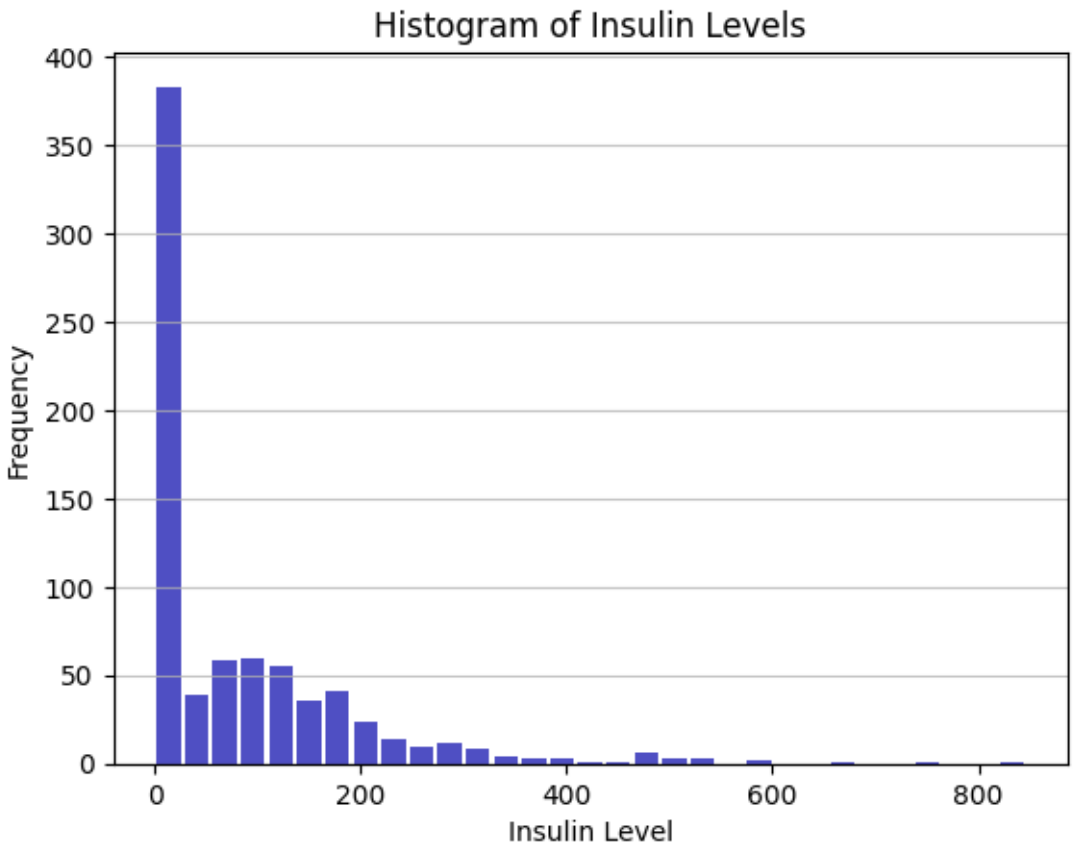
[40]:
```
# Histogram for number of pregnancies
plt.hist(diabetes_df['Pregnancies'], bins='auto', color='#0504aa', alpha=0.7,␣
 ↪rwidth=0.85)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('Number of Pregnancies')
plt.ylabel('Frequency')
plt.title('Histogram of Pregnancies')
plt.show()
```



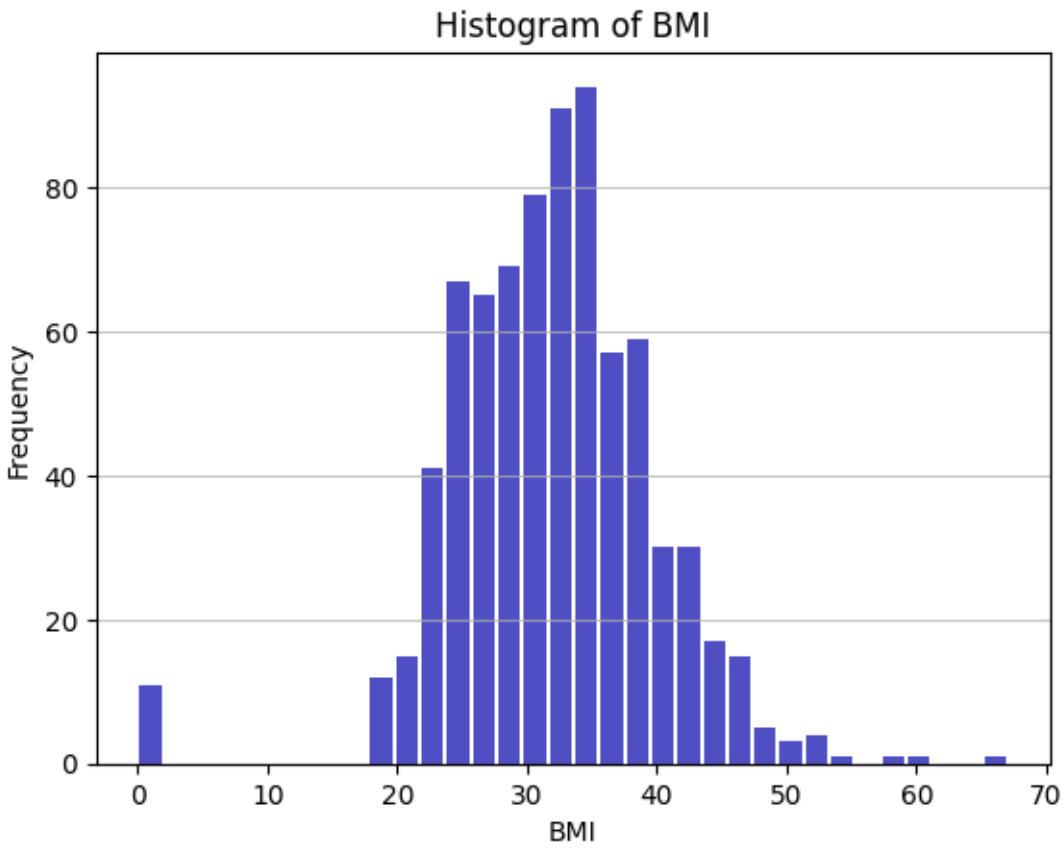
[41]:
```
# Histogram for Glucose Levels
plt.hist(diabetes_df['Glucose'], bins='auto', color='#0504aa', alpha=0.7,␣
 ↪rwidth=0.85)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('Glucose Level')
plt.ylabel('Frequency')
plt.title('Histogram of Glucose Levels')
plt.show()
```



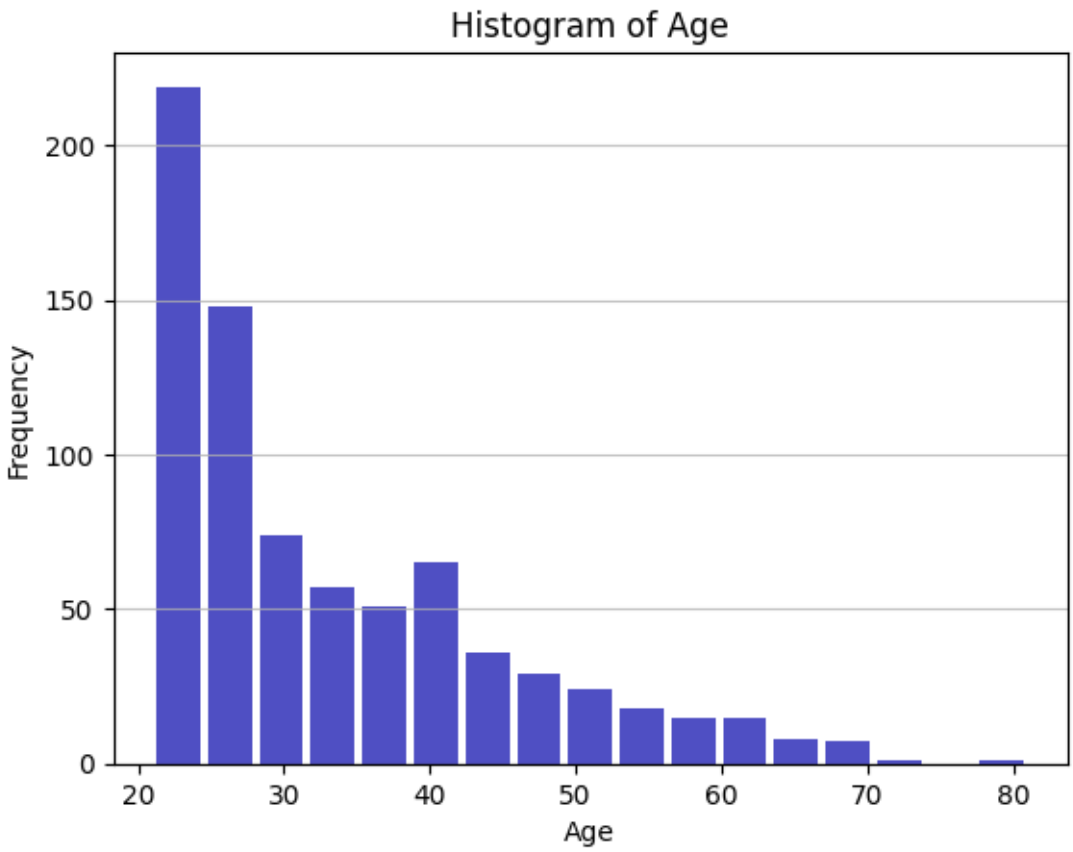
[42]:
```
# Histogram for Insulin levels
plt.hist(diabetes_df['Insulin'], bins='auto', color='#0504aa', alpha=0.7,␣
 ↪rwidth=0.85)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('Insulin Level')
plt.ylabel('Frequency')
plt.title('Histogram of Insulin Levels')
plt.show()
```



[43]:
```
# Histogram of the BMI
plt.hist(diabetes_df['BMI'], bins='auto', color='#0504aa', alpha=0.7, rwidth=0.
↪85)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('BMI')
plt.ylabel('Frequency')
plt.title('Histogram of BMI')
plt.show()
```



[44]:
```
# Histogram for the Age
plt.hist(diabetes_df['Age'], bins='auto', color='#0504aa', alpha=0.7, rwidth=0.
↪85)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Histogram of Age')
plt.show()
```



2.2 Descriptive Characteristics

[45]:
```
# Descriptive Characteristics for Pregnancies
mean_pregnancies = diabetes_df['Pregnancies'].mean()
mode_pregnancies = diabetes_df['Pregnancies'].mode()[0]
range_pregnancies = diabetes_df['Pregnancies'].max() -␣
 ↪diabetes_df['Pregnancies'].min()
skewness_pregnancies = diabetes_df['Pregnancies'].skew()
kurtosis_pregnancies = diabetes_df['Pregnancies'].kurtosis()

# Print the calculated statistics
print(f"'Pregnancies' Mean: {mean_pregnancies}, Mode: {mode_pregnancies}, Range:
 ↪ {range_pregnancies}, Skewness: {skewness_pregnancies}, Kurtosis:␣
 ↪{kurtosis_pregnancies}")
```

```
'Pregnancies' Mean: 3.8450520833333335, Mode: 1, Range: 17, Skewness:
0.9016739791518588, Kurtosis: 0.15921977754746486
```

[46]:
```
# Descriptive Characteristics for Glucose
mean_glucose = diabetes_df['Glucose'].mean()
```

```
mode_glucose = diabetes_df['Glucose'].mode()[0]
range_glucose = diabetes_df['Glucose'].max() - diabetes_df['Glucose'].min()
skewness_glucose = diabetes_df['Glucose'].skew()
kurtosis_glucose = diabetes_df['Glucose'].kurtosis()

# Print the calculated statistics
print(f"'Glucose' Mean: {mean_glucose}, Mode: {mode_glucose}, Range:␣
 ↪{range_glucose}, Skewness: {skewness_glucose}, Kurtosis: {kurtosis_glucose}")
```

```
'Glucose' Mean: 120.89453125, Mode: 99, Range: 199, Skewness:
0.17375350179188992, Kurtosis: 0.6407798203735053
```

[47]:
```
# Descriptive Characteristics for BMI
mean_bmi = diabetes_df['BMI'].mean()
mode_bmi = diabetes_df['BMI'].mode()[0]
range_bmi = diabetes_df['BMI'].max() - diabetes_df['BMI'].min()
skewness_bmi = diabetes_df['BMI'].skew()
kurtosis_bmi = diabetes_df['BMI'].kurtosis()

# Print the calculated statistics
print(f"'BMI' Mean: {mean_bmi}, Mode: {mode_bmi}, Range: {range_bmi}, Skewness:␣
 ↪{skewness_bmi}, Kurtosis: {kurtosis_bmi}")
```

```
'BMI' Mean: 31.992578124999998, Mode: 32.0, Range: 67.1, Skewness:
-0.42898158845356543, Kurtosis: 3.290442900816981
```

[48]:
```
# Descriptive Characteristics for Age
mean_age = diabetes_df['Age'].mean()
mode_age = diabetes_df['Age'].mode()[0]
range_age = diabetes_df['Age'].max() - diabetes_df['Age'].min()
skewness_age = diabetes_df['Age'].skew()
kurtosis_age = diabetes_df['Age'].kurtosis()

# Print the calculated statistics
print(f"'Age' Mean: {mean_age}, Mode: {mode_age}, Range: {range_age}, Skewness:␣
 ↪{skewness_age}, Kurtosis: {kurtosis_age}")
```

```
'Age' Mean: 33.240885416666664, Mode: 22, Range: 60, Skewness:
1.1295967011444805, Kurtosis: 0.6431588885398942
```

[49]:
```
# Descriptive Characteristics for Insulin
mean_insulin = diabetes_df['Insulin'].mean()
mode_insulin = diabetes_df['Insulin'].mode()[0]
range_insulin = diabetes_df['Insulin'].max() - diabetes_df['Insulin'].min()
skewness_insulin = diabetes_df['Insulin'].skew()
kurtosis_insulin = diabetes_df['Insulin'].kurtosis()

# Print the calculated statistics
```

```
print(f"'Insulin' Mean: {mean_insulin}, Mode: {mode_insulin}, Range:␣
↪{range_insulin}, Skewness: {skewness_insulin}, Kurtosis: {kurtosis_insulin}")
```

```
'Insulin' Mean: 79.79947916666667, Mode: 0, Range: 846, Skewness:
2.272250858431574, Kurtosis: 7.2142595543487715
```

[ ]:

2.3 Two scenarios for PMF

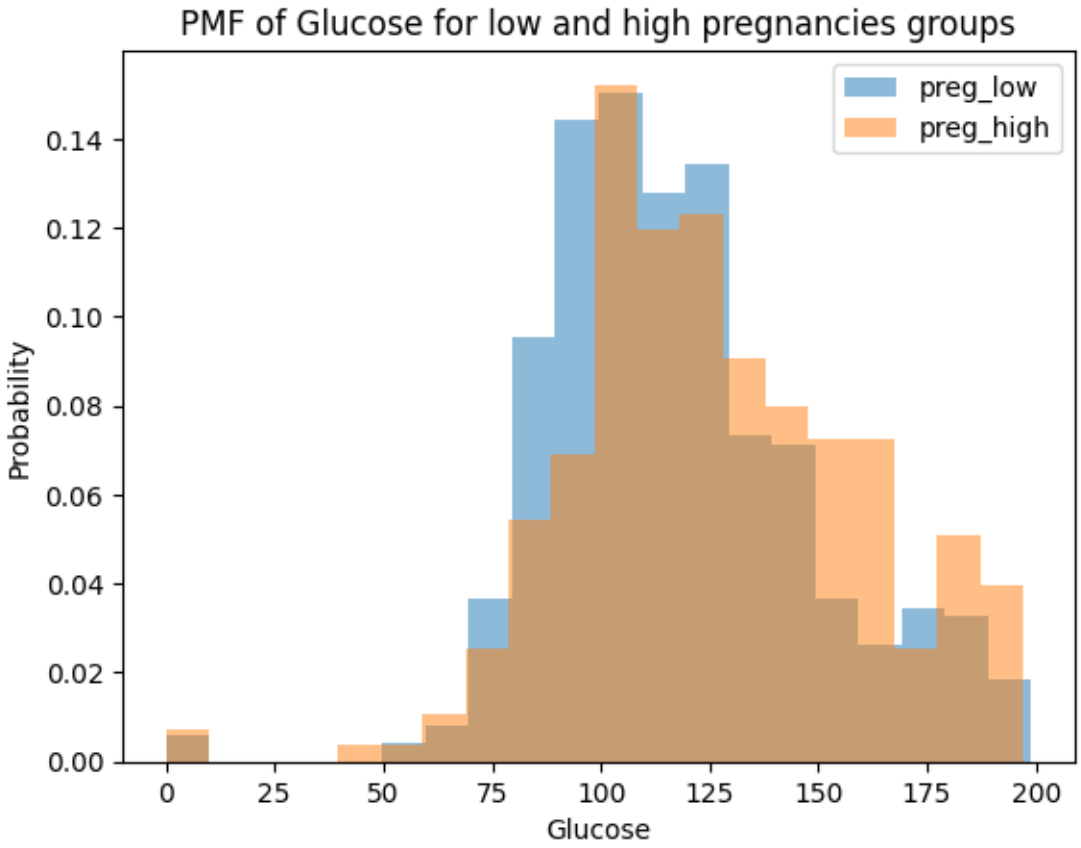
[50]:
```
# Compare the variables Pregnancies with pregnancise more than 4 and equal or␣
↪less than 4
preg_low = diabetes_df[diabetes_df['Pregnancies'] <= 4]
preg_high = diabetes_df[diabetes_df['Pregnancies'] > 4]
```

[51]:
```
preg_high
```

[51]:
```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
2              8      183             64              0        0  23.3
5              5      116             74              0        0  25.6
7             10      115              0              0        0  35.3
9              8      125             96              0        0   0.0
..           ...      ...            ...            ...      ...   ...
759            6      190             92              0        0  35.5
761            9      170             74             31        0  44.0
762            9       89             62              0        0  22.5
763           10      101             76             48      180  32.9
765            5      121             72             23      112  26.2

     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
2                       0.672   32        1
5                       0.201   30        0
7                       0.134   29        0
9                       0.232   54        1
..                        ...  ...      ...
759                     0.278   66        1
761                     0.403   43        1
762                     0.142   33        0
763                     0.171   63        0
765                     0.245   30        0

[276 rows x 9 columns]
```

[52]:
```
weights_low = np.ones_like(preg_low['Glucose']) /␣
↪float(len(preg_low['Glucose']))
```

```
# calculate weights for preg_high group to get a probability for PMF
weights_high = np.ones_like(preg_high['Glucose']) /␣
 ↪float(len(preg_high['Glucose']))

# plot the PMFs
plt.hist(preg_low['Glucose'], weights=weights_low, alpha=0.5, label='preg_low',␣
 ↪bins=20)
plt.hist(preg_high['Glucose'], weights=weights_high, alpha=0.5,␣
 ↪label='preg_high', bins=20)

plt.xlabel('Glucose')
plt.ylabel('Probability')
plt.title('PMF of Glucose for low and high pregnancies groups')
plt.legend(loc='upper right')

plt.show()
```



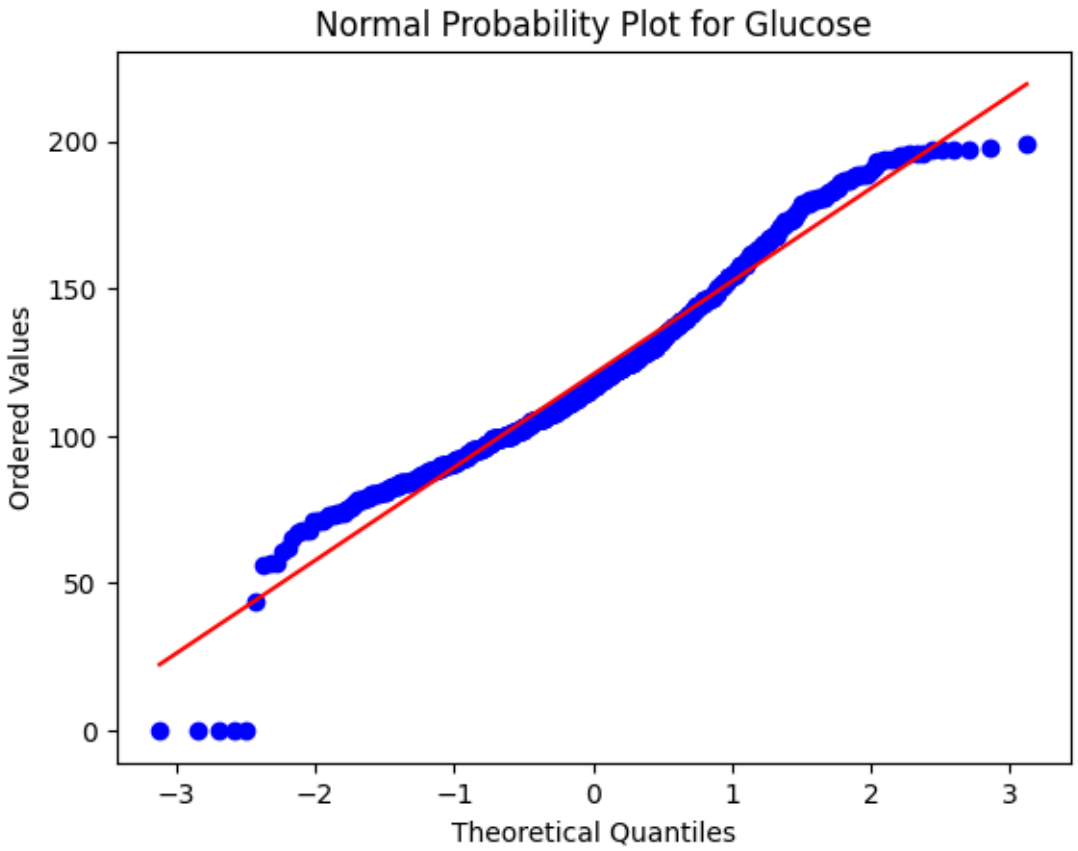
2.4 CDF of Glucose for low and high pregnancies groups

[ ]:

2.5 Plot analytical distribution

[53]:
```
# import library
from scipy.stats import probplot
```

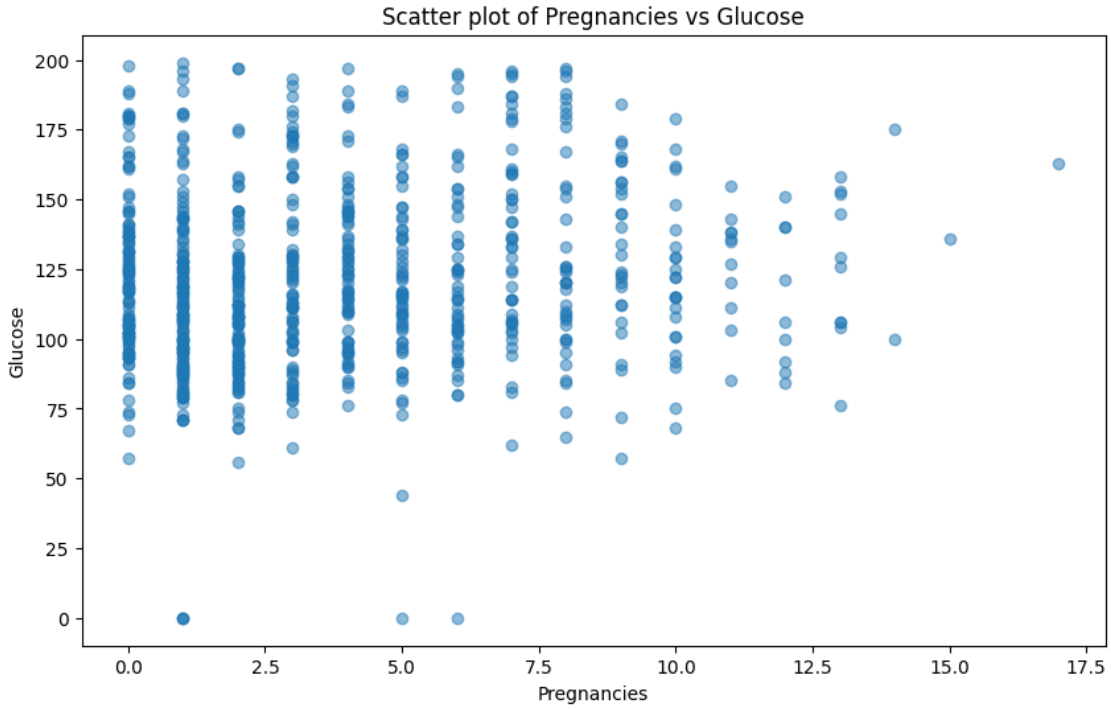
[54]:
```
glucose_data = pd.concat([preg_low['Glucose'], preg_high['Glucose']])

# Remove NaN values, if any
glucose_data = glucose_data.dropna()

# Create a normal probability plot
probplot(glucose_data, dist='norm', plot=plt)

plt.title('Normal Probability Plot for Glucose')
plt.xlabel('Theoretical Quantiles')
plt.ylabel('Ordered Values')

plt.show()
```



[55]:
```
# The data points deviate from the straight line, especially at the tails
# It indicates that the Glucose may not be perfectly normally distributed
```

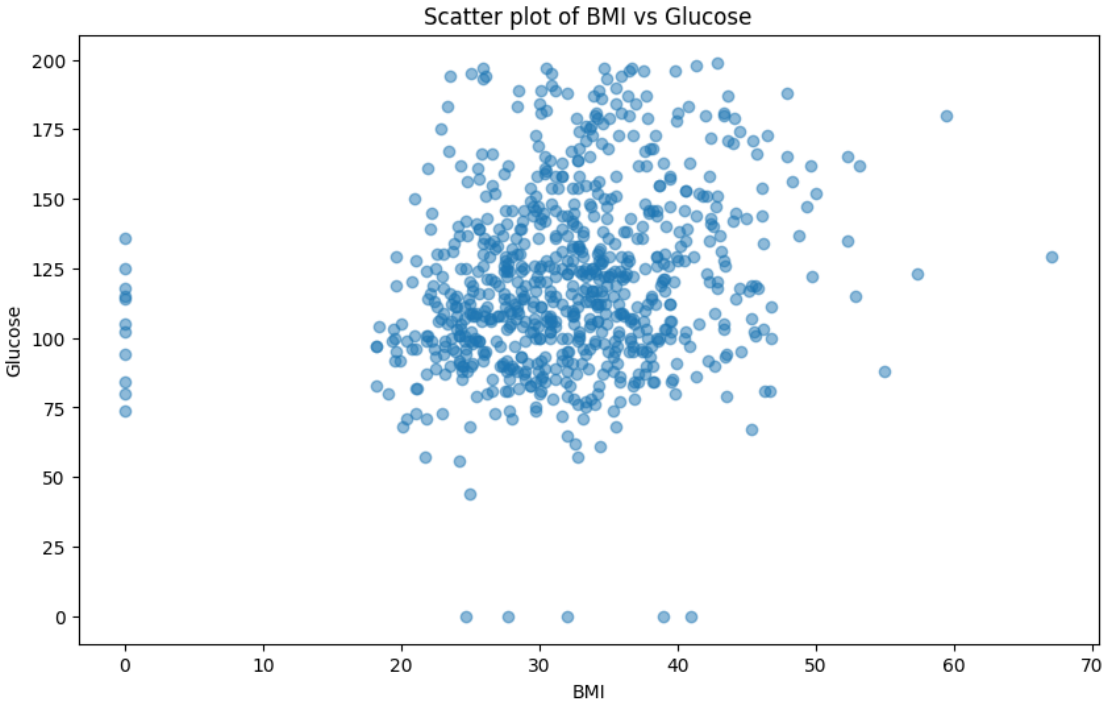
2.6 Scatter plots comparing two variables

[56]:
```
plt.figure(figsize=(10,6))
plt.scatter(diabetes_df['Pregnancies'], diabetes_df['Glucose'], alpha=0.5)
plt.title('Scatter plot of Pregnancies vs Glucose')
plt.xlabel('Pregnancies')
plt.ylabel('Glucose')
plt.show()
```



[ ]:
```
```

[57]:
```
plt.figure(figsize=(10,6))
plt.scatter(diabetes_df['BMI'], diabetes_df['Glucose'], alpha=0.5)
plt.title('Scatter plot of BMI vs Glucose')
plt.xlabel('BMI')
plt.ylabel('Glucose')
plt.show()
```



[ ]:

[58]:
```
cov_matrix = diabetes_df.cov()

# Print the covariance matrix
print(cov_matrix)
```

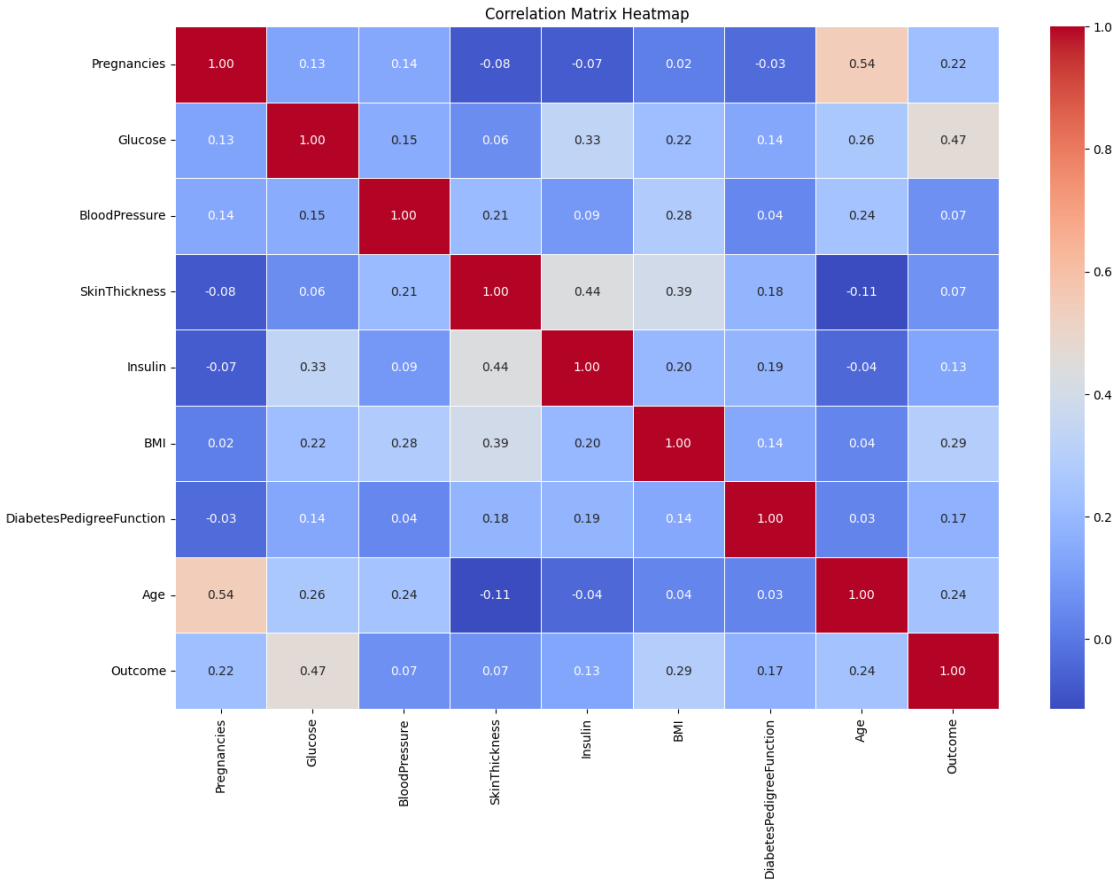
```
                          Pregnancies      Glucose  BloodPressure  \
Pregnancies                 11.354056    13.947131       9.214538
Glucose                     13.947131  1022.248314      94.430956
BloodPressure                9.214538    94.430956     374.647271
SkinThickness               -4.390041    29.239183      64.029396
Insulin                    -28.555231  1220.935799     198.378412
BMI                          0.469774    55.726987      43.004695
DiabetesPedigreeFunction    -0.037426     1.454875       0.264638
Age                         21.570620    99.082805      54.523453
Outcome                      0.356618     7.115079       0.600697

                          SkinThickness       Insulin         BMI  \
Pregnancies                   -4.390041    -28.555231    0.469774
Glucose                       29.239183   1220.935799   55.726987
BloodPressure                 64.029396    198.378412   43.004695
SkinThickness                254.473245    802.979941   49.373869
Insulin                      802.979941  13281.180078  179.775172
BMI                           49.373869    179.775172   62.159984
```

```
DiabetesPedigreeFunction       0.972136      7.066681     0.367405
Age                          -21.381023    -57.143290     3.360330
Outcome                        0.568747      7.175671     1.100638
```

	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	-0.037426	21.570620	0.356618
Glucose	1.454875	99.082805	7.115079
BloodPressure	0.264638	54.523453	0.600697
SkinThickness	0.972136	-21.381023	0.568747
Insulin	7.066681	-57.143290	7.175671
BMI	0.367405	3.360330	1.100638
DiabetesPedigreeFunction	0.109779	0.130772	0.027472
Age	0.130772	138.303046	1.336953
Outcome	0.027472	1.336953	0.227483

[59]:
```
import seaborn as sns
```

[60]:
```
corr = diabetes_df.corr()
plt.figure(figsize=(15, 10))
sns.heatmap(corr, annot=True, cmap='coolwarm', linewidths=0.5, fmt='.2f')
plt.title("Correlation Matrix Heatmap")
plt.show()
```



2.7 Hypothesis Test

[ ]:
```
# Conduct a HypothesisTest
# Import libraries and find the spearman rank correlation coeffficient between␣
 ↪Pregnancies and Glucose
# Use Shapiro-Wilk test for normality testing
from scipy.stats import shapiro
from scipy.stats import spearmanr

pregnancies_normality_pvalue = shapiro(diabetes_df['Pregnancies'])[1]
glucose_normality_pvalue = shapiro(diabetes_df['Glucose'])[1]

print(f'p-value for Pregnancies is {pregnancies_normality_pvalue}')
print(f'p-value for Glucose is {glucose_normality_pvalue}')
```

[62]:
```
# Find the spearman rank corr and p-value
spearman_corr, spearman_pvalue = spearmanr(diabetes_df['Pregnancies'],␣
 ↪diabetes_df['Glucose'])
print(f'The spearman rank correlation coefficient is {spearman_corr}')
print(f'The p-value is {spearman_pvalue}')
```

```
The spearman rank correlation coefficient is 0.13073352406886674
The p-value is 0.0002804774625063403
```

2.8 Regression analysis

[63]:
```
# Regression Analysis for the Agae and BloodPressure
# Age is the independent variable and BloodPressure is the Outcome
# Import statsmodels library
import statsmodels.formula.api as smf
```

[64]:
```
# Perform the regression analysis
model_age_bp = smf.ols('BloodPressure~Age', data=diabetes_df).fit()
model_age_bp.summary()
```

[64]:

Dep. Variable:	BloodPressure	**R-squared:**	0.057
Model:	OLS	**Adj. R-squared:**	0.056
Method:	Least Squares	**F-statistic:**	46.62
Date:	Thu, 19 Sep 2024	**Prob (F-statistic):**	1.75e-11
Time:	02:15:17	**Log-Likelihood:**	-3342.1
No. Observations:	768	**AIC:**	6688.
Df Residuals:	766	**BIC:**	6698.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> |t|	[0.025	0.975]
Intercept	56.0009	2.036	27.510	0.000	52.005	59.997
Age	0.3942	0.058	6.828	0.000	0.281	0.508

Omnibus:	328.846	**Durbin-Watson:**	1.968
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1541.275
Skew:	-1.950	**Prob(JB):**	0.00
Kurtosis:	8.741	**Cond. No.**	106.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Coefficient for Age: 0.3942 (Age has a positive effect on blood pressure. It represents that the is an average increase of 0.3942 of blood presure with every additional year increase in age)The relationship between age an blood pressure is statistically significant. However, R-squared value(0.057) indicates that age is only a small fraction of the variablility in blood pressure, and there may have other factors affecting the changing of blood pressure

[ ]:

[ ]: