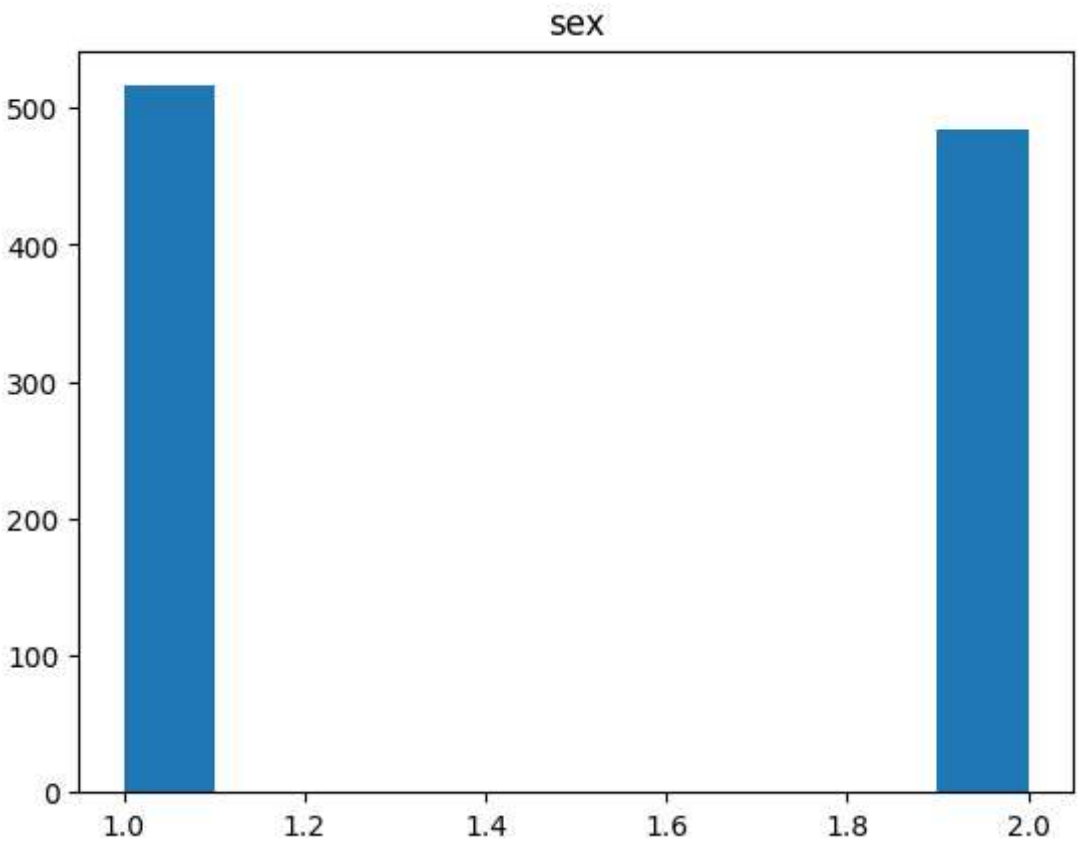
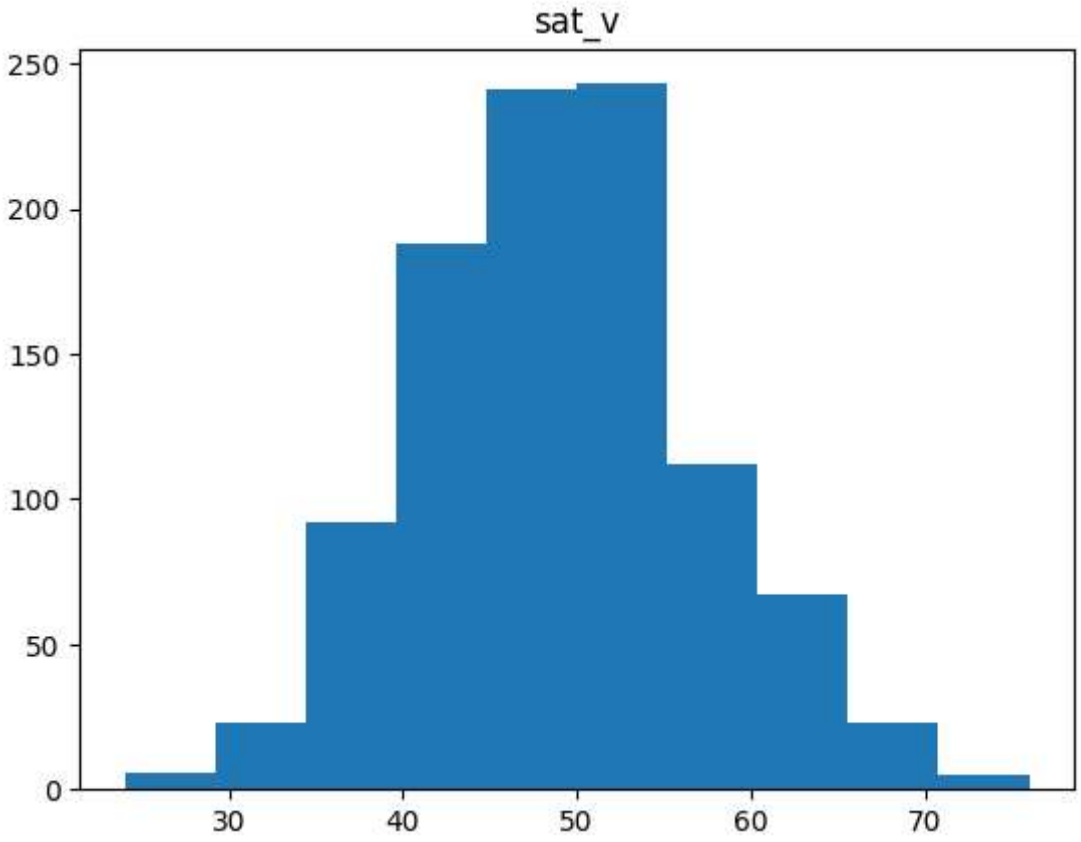
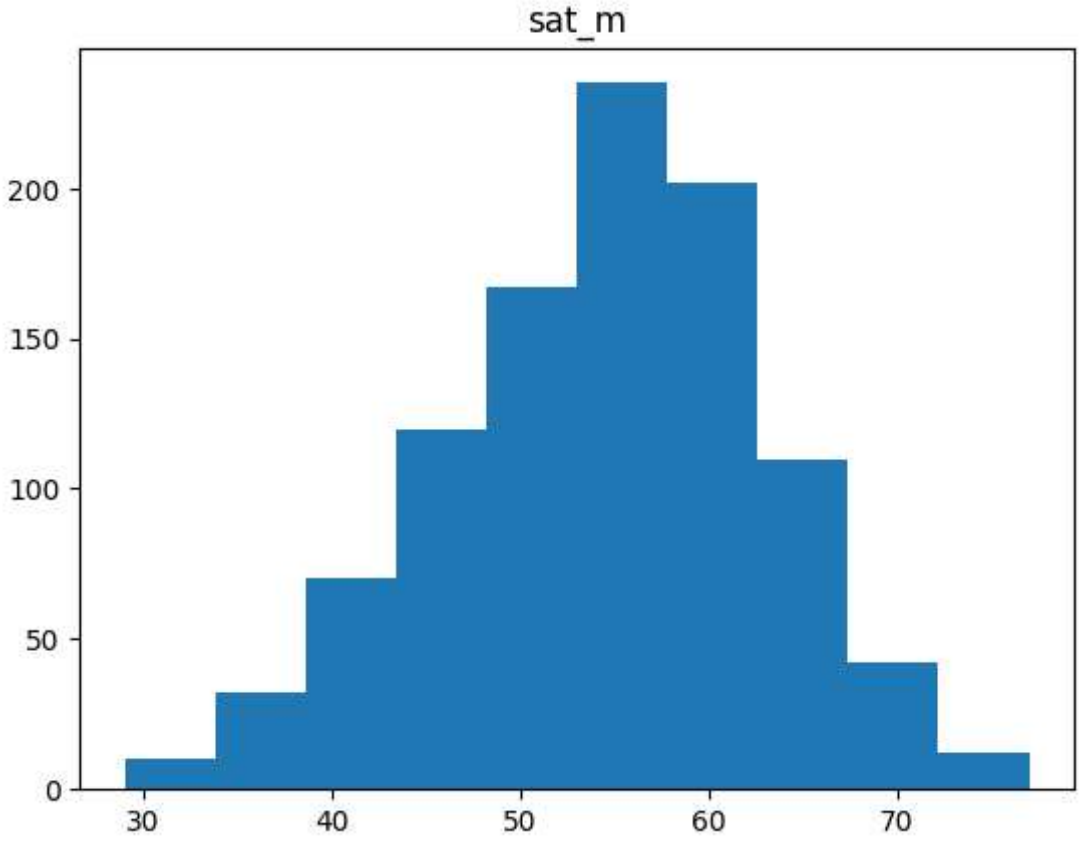
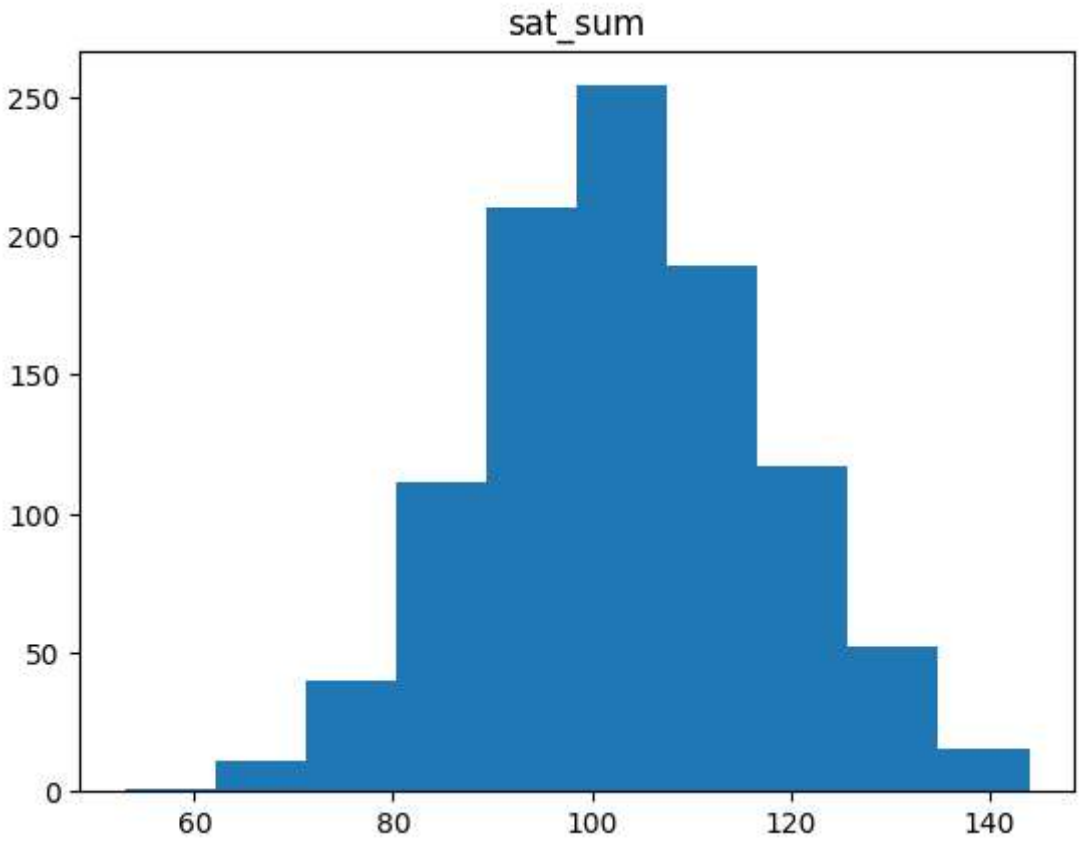
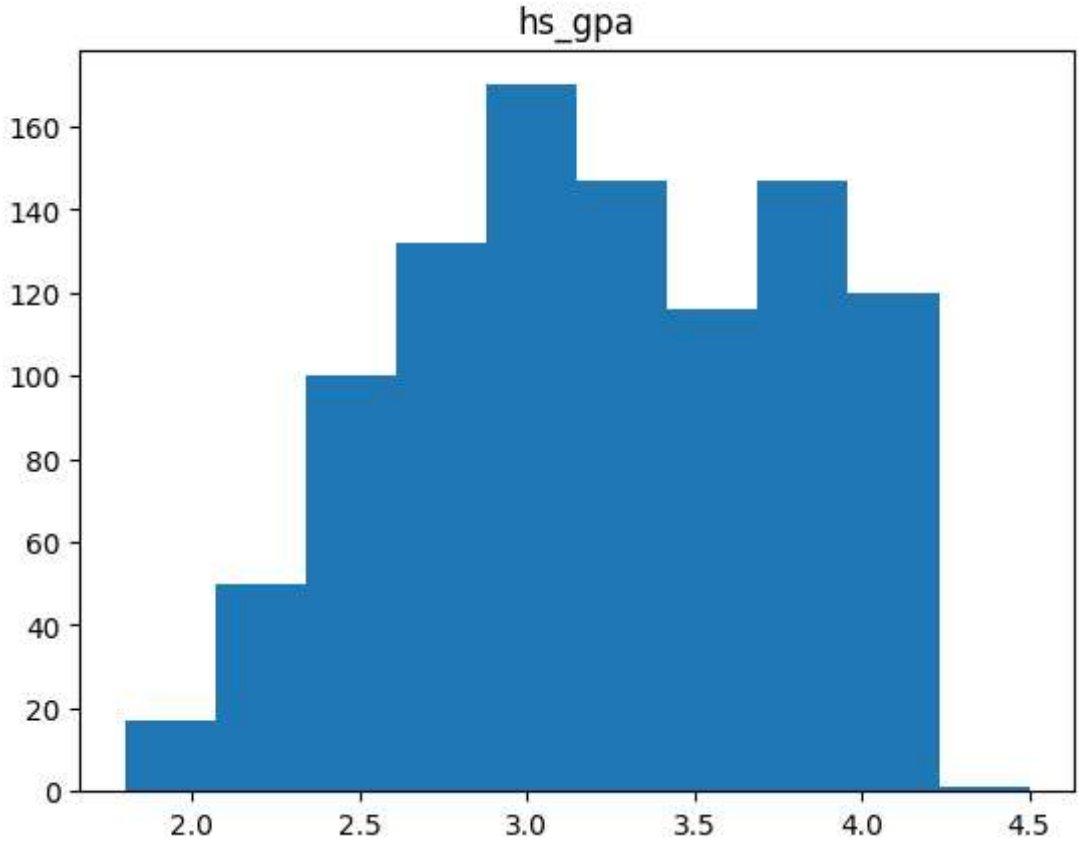
In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read CSV file into a DataFrame
df = pd.read_csv('satgpa.csv')

# Create a histogram for each column
for col in df.columns:
    plt.hist(df[col])
    plt.title(col)
    plt.show()
```

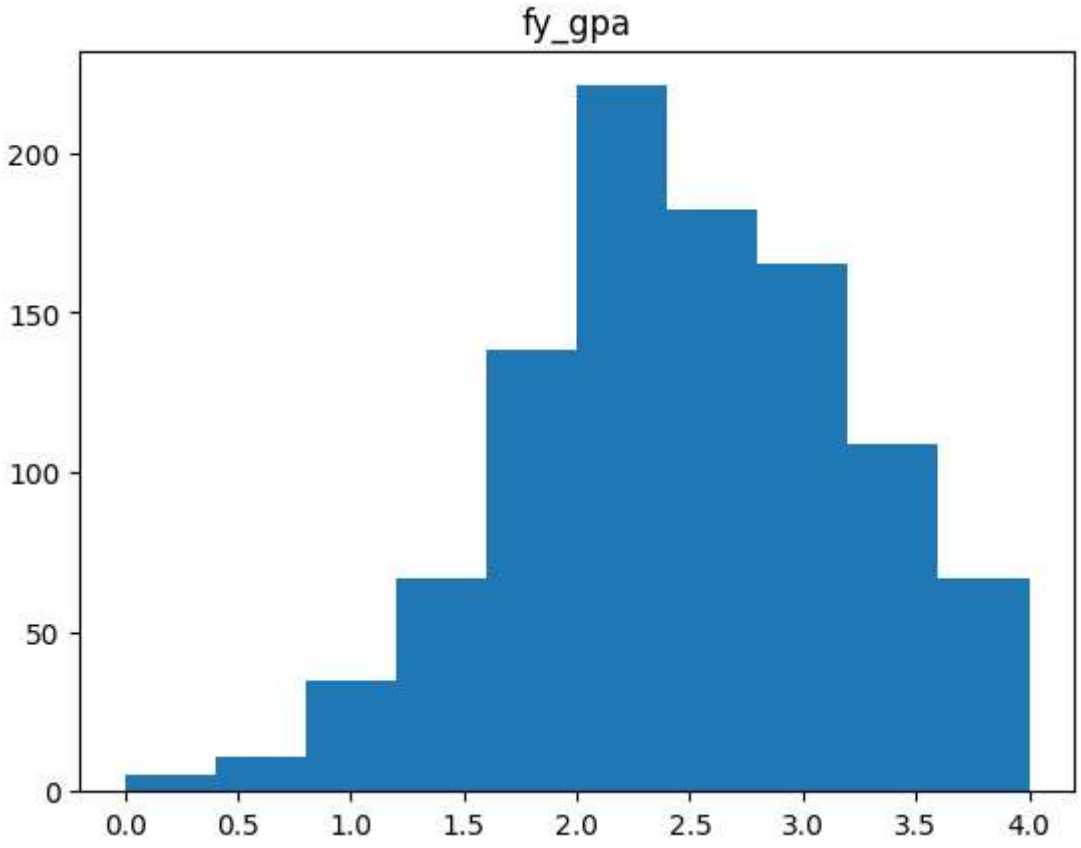



sat_v
250
200
150
100
50
0
30
40
50
60
70


sat_m
200
150
100
50
0
30
40
50
60
70


sat_sum
250
200
150
100
50
0
60
80
100
120
140

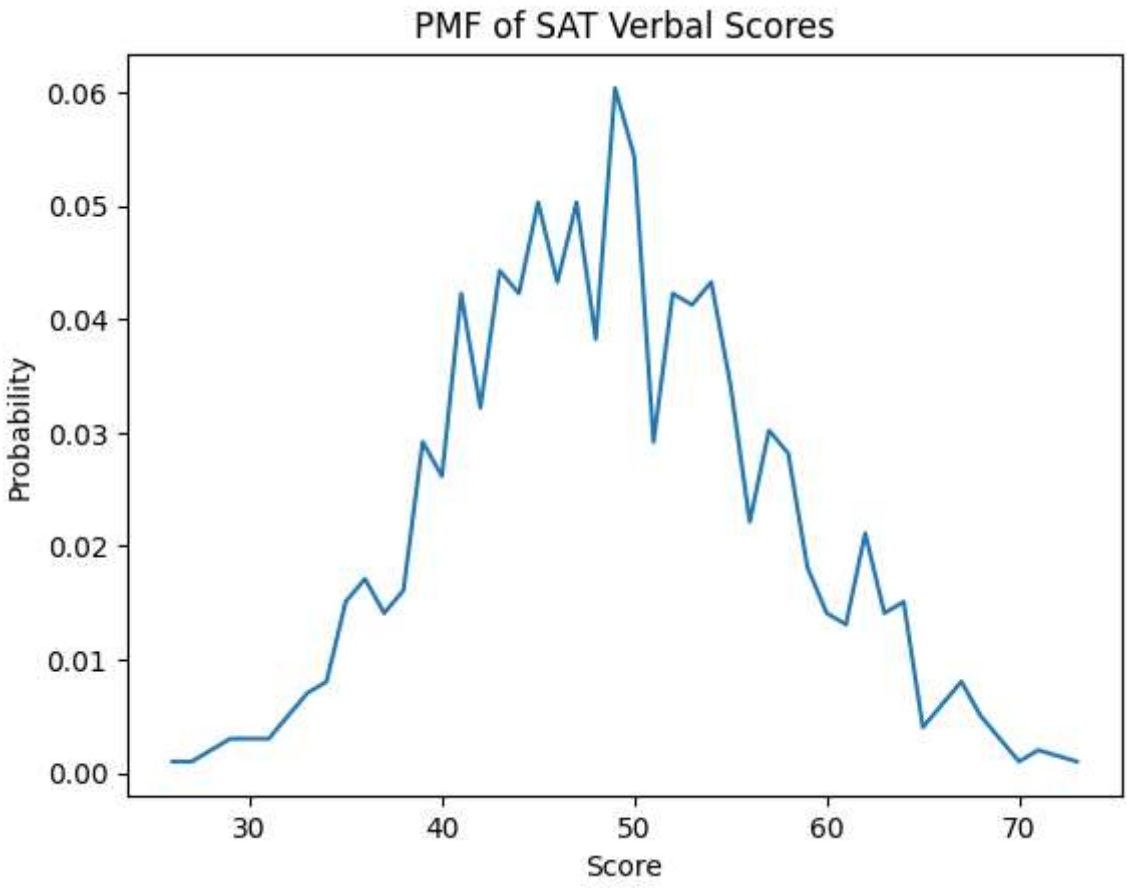

hs_gpa
160
140
120
100
80
60
40
20
0
2.0
2.5
3.0
3.5
4.0
4.5



In [2]:
```python
import pandas as pd

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# compute descriptive statistics for each column
for col in df.columns:
    mean = df[col].mean()
    mode = df[col].mode().iloc[0]
    std = df[col].std()
    min_val = df[col].min()
    max_val = df[col].max()
    q1 = df[col].quantile(0.25)
    q3 = df[col].quantile(0.75)
    iqr = q3 - q1

    print(f"Column '{col}':")
    print(f"\tMean: {mean:.2f}")
    print(f"\tMode: {mode}")
    print(f"\tStandard Deviation: {std:.2f}")
    print(f"\tMinimum Value: {min_val}")
    print(f"\tMaximum Value: {max_val}")
    print(f"\t1st Quartile: {q1:.2f}")
    print(f"\t3rd Quartile: {q3:.2f}")
    print(f"\tInterquartile Range: {iqr:.2f}")
```

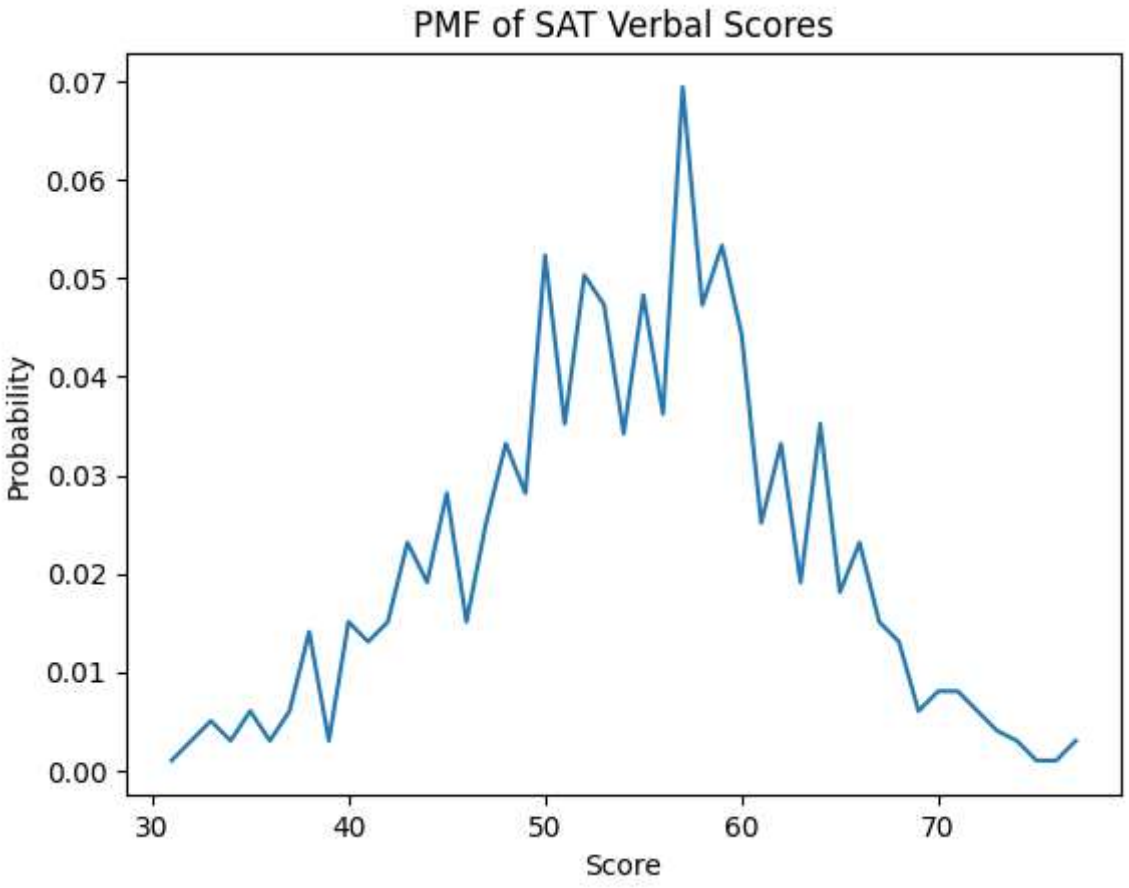
```
Column 'sex':
        Mean: 1.48
        Mode: 1
        Standard Deviation: 0.50
        Minimum Value: 1
        Maximum Value: 2
        1st Quartile: 1.00
        3rd Quartile: 2.00
        Interquartile Range: 1.00
Column 'sat_v':
        Mean: 48.93
        Mode: 49
        Standard Deviation: 8.12
        Minimum Value: 26
        Maximum Value: 73
        1st Quartile: 43.00
        3rd Quartile: 54.00
        Interquartile Range: 11.00
Column 'sat_m':
        Mean: 54.43
        Mode: 57
        Standard Deviation: 8.41
        Minimum Value: 31
        Maximum Value: 77
        1st Quartile: 49.00
        3rd Quartile: 60.00
        Interquartile Range: 11.00
Column 'sat_sum':
        Mean: 103.36
        Mode: 103
        Standard Deviation: 14.12
        Minimum Value: 65
        Maximum Value: 144
        1st Quartile: 93.00
        3rd Quartile: 112.75
        Interquartile Range: 19.75
Column 'hs_gpa':
        Mean: 3.20
        Mode: 4.0
        Standard Deviation: 0.54
        Minimum Value: 1.8
        Maximum Value: 4.5
        1st Quartile: 2.80
        3rd Quartile: 3.70
        Interquartile Range: 0.90
Column 'fy_gpa':
        Mean: 2.47
        Mode: 2.24
        Standard Deviation: 0.73
        Minimum Value: 0.36
        Maximum Value: 4.0
        1st Quartile: 1.98
        3rd Quartile: 3.02
        Interquartile Range: 1.04
```

In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# compute PMF for sat_v
pmf_v = df['sat_v'].value_counts(normalize=True).sort_index()

# plot PMF for sat_v
fig, ax = plt.subplots()
ax.plot(pmf_v.index, pmf_v.values)
ax.set_xlabel('Score')
ax.set_ylabel('Probability')
ax.set_title('PMF of SAT Verbal Scores')
plt.show()
```

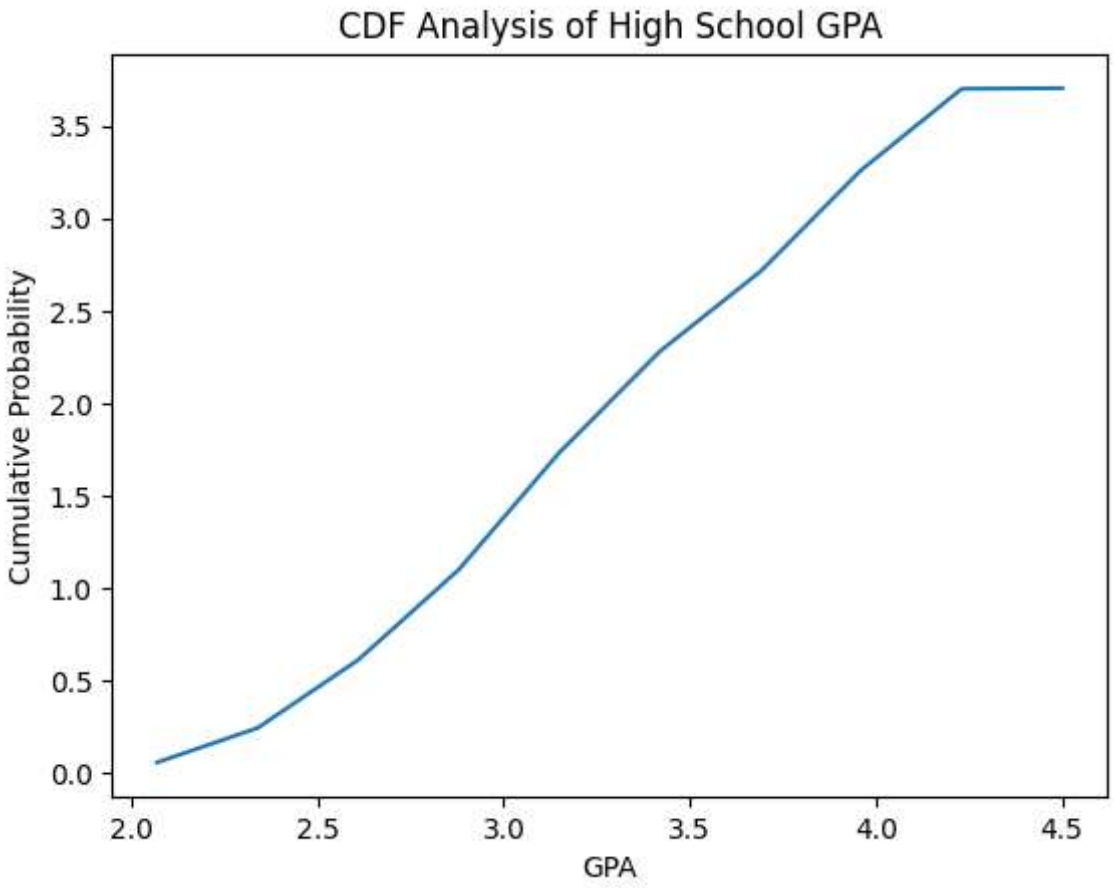


In [4]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# compute PMF for sat_m
pmf_v = df['sat_m'].value_counts(normalize=True).sort_index()

# plot PMF for sat_m
fig, ax = plt.subplots()
```

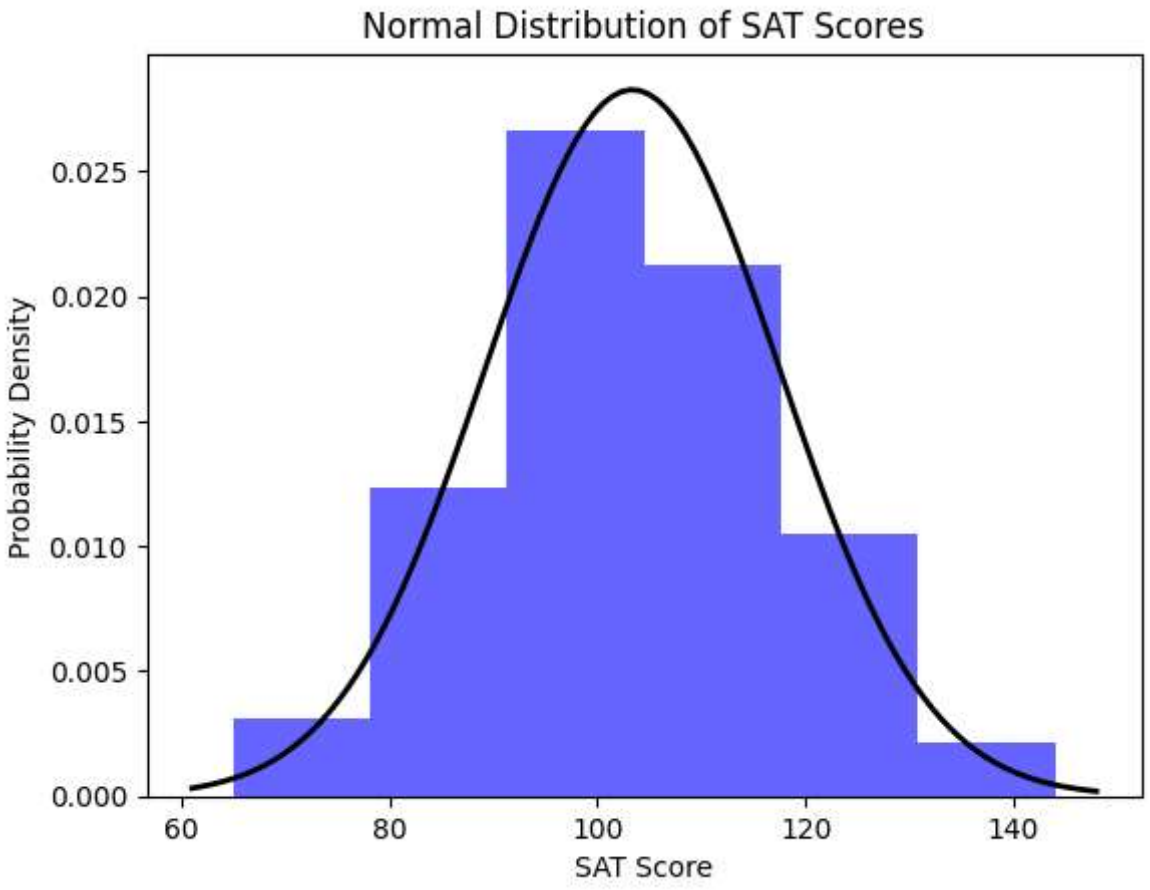
```
ax.plot(pmf_v.index, pmf_v.values)
ax.set_xlabel('Score')
ax.set_ylabel('Probability')
ax.set_title('PMF of SAT Verbal Scores')
plt.show()
```



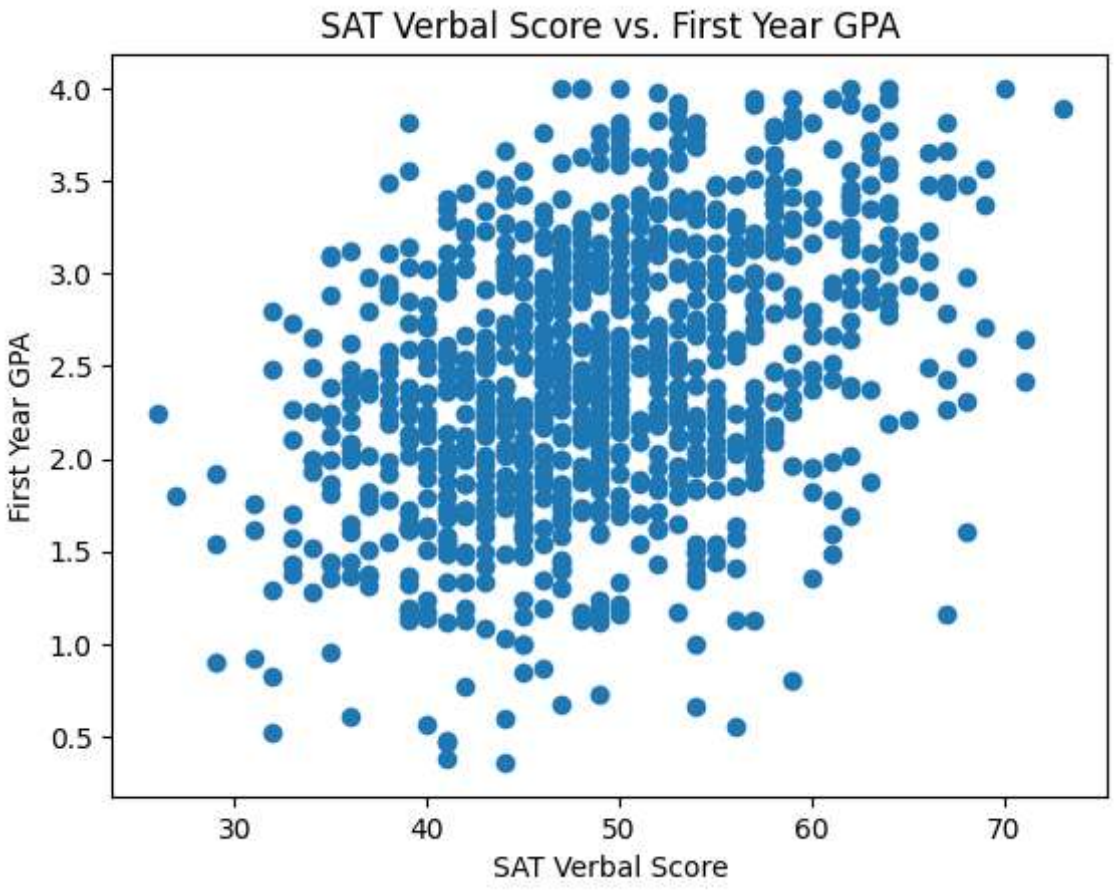
In [5]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# compute CDF for hs_gpa
cdf = np.cumsum(np.histogram(df['hs_gpa'], bins=10, density=True)[0])

# plot CDF for hs_gpa
fig, ax = plt.subplots()
ax.plot(np.histogram(df['hs_gpa'], bins=10, density=True)[1][1:], cdf)
ax.set_xlabel('GPA')
ax.set_ylabel('Cumulative Probability')
ax.set_title('CDF Analysis of High School GPA')
plt.show()
```



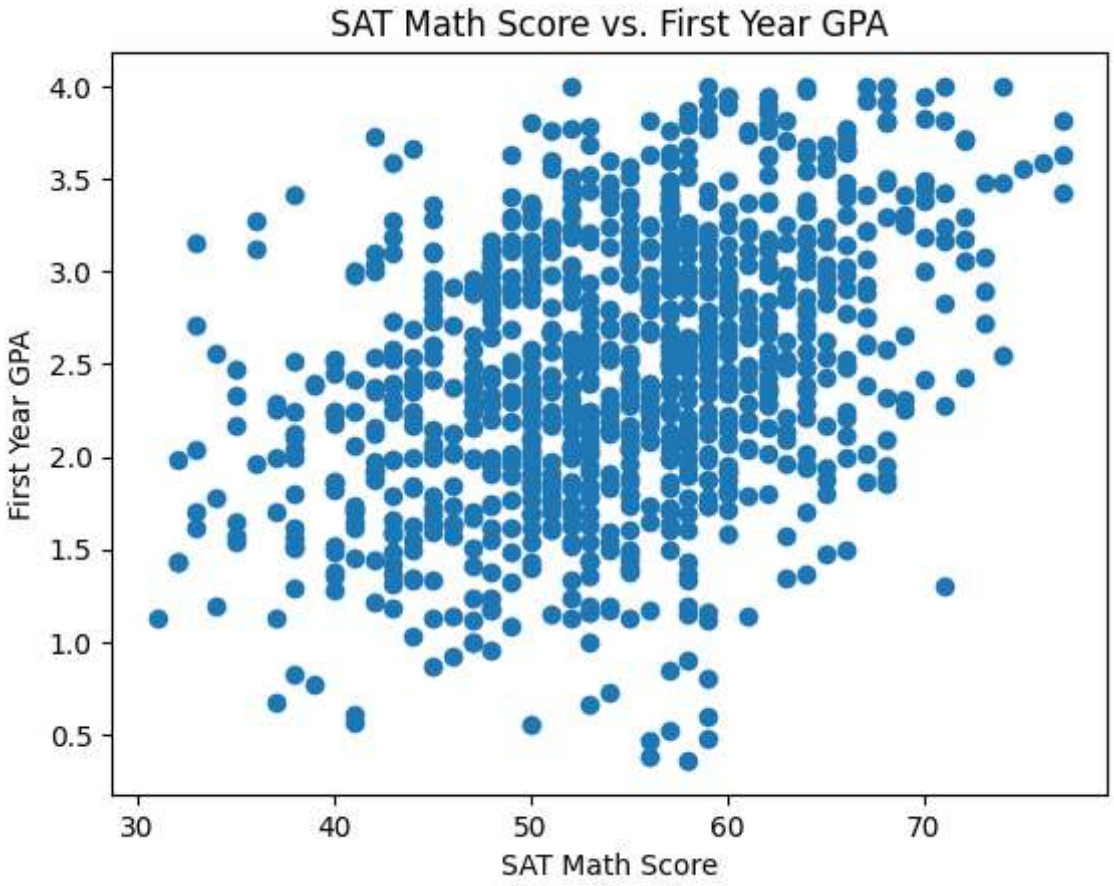
In [6]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# calculate mean and standard deviation
mu, std = norm.fit(df['sat_sum'])

# create a normal distribution plot
plt.hist(df['sat_sum'], bins=6, density=True, alpha=0.6, color='b')

xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
plt.plot(x, p, 'k', linewidth=2)

plt.title('Normal Distribution of SAT Scores')
plt.xlabel('SAT Score')
plt.ylabel('Probability Density')
plt.show()
```



In [7]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# Create the scatter plot
plt.scatter(df['sat_v'], df['fy_gpa'])

# Set the plot title and axis labels
plt.title('SAT Verbal Score vs. First Year GPA')
plt.xlabel('SAT Verbal Score')
plt.ylabel('First Year GPA')

# Show the plot
plt.show()
```



In [8]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# Create the scatter plot
plt.scatter(df['sat_m'], df['fy_gpa'])

# Set the plot title and axis labels
plt.title('SAT Math Score vs. First Year GPA')
plt.xlabel('SAT Math Score')
plt.ylabel('First Year GPA')

# Show the plot
plt.show()
```



In [9]:
```python
import pandas as pd

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# Calculate the correlation between 'fy_gpa' and other columns
corr_matrix = df.corr()['fy_gpa']

# Print the correlation coefficients
print(corr_matrix)
```

```
sex         0.102209
sat_v       0.393295
sat_m       0.384160
sat_sum     0.454933
hs_gpa      0.535207
fy_gpa      1.000000
Name: fy_gpa, dtype: float64
```

In [10]:
```python
import pandas as pd
import statsmodels.api as sm

# Read CSV file into a DataFrame
df = pd.read_csv('cleaned_data.csv')

# Set the predictor variables (X) and response variable (y)
X = df[['sex', 'sat_v', 'sat_m', 'sat_sum', 'hs_gpa']]
X = sm.add_constant(X)  # add a constant term
```

```
y = df['fy_gpa']

# Fit the linear regression model
model = sm.OLS(y, X).fit()

# Print the model summary
print()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 fy_gpa   R-squared:                       0.360
Model:                            OLS   Adj. R-squared:                  0.357
Method:                 Least Squares   F-statistic:                     139.0
Date:                Mon, 08 May 2023   Prob (F-statistic):           2.86e-94
Time:                        22:18:57   Log-Likelihood:                -872.69
No. Observations:                 994   AIC:                             1755.
Df Residuals:                     989   BIC:                             1780.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.0837      0.165     -6.563      0.000      -1.408      -0.760
sex            0.1516      0.040      3.814      0.000       0.074       0.230
sat_v          0.0053      0.002      2.354      0.019       0.001       0.010
sat_m          0.0053      0.002      2.334      0.020       0.001       0.010
sat_sum        0.0106      0.001     10.588      0.000       0.009       0.013
hs_gpa         0.5270      0.039     13.437      0.000       0.450       0.604
==============================================================================
Omnibus:                       24.862   Durbin-Watson:                   2.023
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               26.507
Skew:                          -0.363   Prob(JB):                     1.75e-06
Kurtosis:                       3.334   Cond. No.                     3.12e+15
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
[2] The smallest eigenvalue is 1.67e-24. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

In [ ]: