In [1]:

```
pip install pandas numpy scipy scikit-learn statsmodels
```

```
Requirement already satisfied: pandas in c:\users\pavan\appdata\local\programs\pytho
n\python312\lib\site-packages (2.2.2)
Requirement already satisfied: numpy in c:\users\pavan\appdata\local\programs\python
\python312\lib\site-packages (2.1.1)
Requirement already satisfied: scipy in c:\users\pavan\appdata\local\programs\python
\python312\lib\site-packages (1.14.1)
Requirement already satisfied: scikit-learn in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (1.5.1)
Requirement already satisfied: statsmodels in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (0.14.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\pavan\appdata\loca
l\programs\python\python312\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\pavan\appdata\local\progra
ms\python\python312\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: joblib>=1.2.0 in c:\users\pavan\appdata\local\program
s\python\python312\lib\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\pavan\appdata\local
\programs\python\python312\lib\site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: patsy>=0.5.6 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from statsmodels) (0.5.6)
Requirement already satisfied: packaging>=21.3 in c:\users\pavan\appdata\local\progr
ams\python\python312\lib\site-packages (from statsmodels) (24.1)
Requirement already satisfied: six in c:\users\pavan\appdata\local\programs\python\p
ython312\lib\site-packages (from patsy>=0.5.6->statsmodels) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:

```python
import pandas as pd
from sklearn.datasets import load_diabetes

# Load the dataset
diabetes = load_diabetes()
df = pd.DataFrame(data=diabetes.data, columns=diabetes.feature_names)
df['target'] = diabetes.target

# Display the first few rows
print(df.head())
```

```
        age       sex       bmi        bp        s1        s2        s3  \
0  0.038076  0.050680  0.061696  0.021872 -0.044223 -0.034821 -0.043401   
1 -0.001882 -0.044642 -0.051474 -0.026328 -0.008449 -0.019163  0.074412   
2  0.085299  0.050680  0.044451 -0.005670 -0.045599 -0.034194 -0.032356   
3 -0.089063 -0.044642 -0.011595 -0.036656  0.012191  0.024991 -0.036038   
4  0.005383 -0.044642 -0.036385  0.021872  0.003935  0.015596  0.008142   

         s4        s5        s6  target  
0 -0.002592  0.019907 -0.017646   151.0  
1 -0.039493 -0.068332 -0.092204    75.0  
2 -0.002592  0.002861 -0.025930   141.0  
3  0.034309  0.022688 -0.009362   206.0  
4 -0.002592 -0.031988 -0.046641   135.0  
```

In [3]:

```
# Calculate basic descriptive statistics
print("Mean:\n", df.mean())
print("\nMedian:\n", df.median())
print("\nMode:\n", df.mode().iloc[0])
print("\nStandard Deviation:\n", df.std())
print("\nVariance:\n", df.var())

# Additional descriptive statistics
print("\nRange:\n", df.max() - df.min())
print("\nSkewness:\n", df.skew())
print("\nKurtosis:\n", df.kurt())
```

```
Mean:
 age      -1.444295e-18
sex       2.543215e-18
bmi      -2.255925e-16
bp       -4.854086e-17
s1       -1.428596e-17
s2        3.898811e-17
s3       -6.028360e-18
s4       -1.788100e-17
s5        9.243486e-17
s6        1.351770e-17
target    1.521335e+02
dtype: float64

Median:
 age         0.005383
sex        -0.044642
bmi        -0.007284
bp         -0.005670
s1         -0.004321
s2         -0.003819
s3         -0.006584
s4         -0.002592
s5         -0.001947
s6         -0.001078
target    140.500000
dtype: float64

Mode:
 age        0.016281
sex       -0.044642
bmi       -0.030996
bp        -0.040099
s1        -0.037344
s2        -0.001001
s3        -0.013948
s4        -0.039493
s5        -0.018114
s6         0.003064
target    72.000000
Name: 0, dtype: float64

Standard Deviation:
 age        0.047619
sex        0.047619
bmi        0.047619
bp         0.047619
s1         0.047619
s2         0.047619
s3         0.047619
s4         0.047619
s5         0.047619
s6         0.047619
target    77.093005
dtype: float64
```

```
Variance:
 age          0.002268
sex          0.002268
bmi          0.002268
bp           0.002268
s1           0.002268
s2           0.002268
s3           0.002268
s4           0.002268
s5           0.002268
s6           0.002268
target    5943.331348
dtype: float64

Range:
 age         0.217952
sex         0.095322
bmi         0.260831
bp          0.244442
s1          0.280694
s2          0.314401
s3          0.283486
s4          0.261629
s5          0.259694
s6          0.273379
target    321.000000
dtype: float64

Skewness:
 age      -0.231382
sex       0.127385
bmi       0.598148
bp        0.290658
s1        0.378108
s2        0.436592
s3        0.799255
s4        0.735374
s5        0.291754
s6        0.207917
target    0.440563
dtype: float64

Kurtosis:
 age      -0.671224
sex      -1.992811
bmi       0.095094
bp       -0.532797
s1        0.232948
s2        0.601381
s3        0.981507
s4        0.444402
s5       -0.134367
s6        0.236917
target   -0.883057
dtype: float64
```

In [4]:

```
#Performing Inferential Statistics

from scipy import stats

# Example data: BMI values
bmi_values = df['bmi']

# Hypothetical population mean for BMI
population_mean = 0.05

# Perform one-sample t-test
t_stat, p_value = stats.ttest_1samp(bmi_values, population_mean)

print(f"T-Statistic: {t_stat}")
print(f"P-Value: {p_value}")
```

```
T-Statistic: -22.074985843710174
P-Value: 2.7634312235044638e-73
```

In [5]:

```
#Confidence Intervals

import numpy as np
from scipy import stats

# Sample mean and standard error for BMI
sample_mean = np.mean(bmi_values)
standard_error = stats.sem(bmi_values)

# Compute 95% confidence interval for BMI
confidence_interval = stats.norm.interval(0.95, loc=sample_mean, scale=standard_err

print(f"95% Confidence Interval for BMI: {confidence_interval}")
```

```
95% Confidence Interval for BMI: (np.float64(-0.004439332370169141), np.float64(0.00
44393323701686915))
```

In [6]:

```
#Regression Analysis

import statsmodels.api as sm

# Define independent variable (add constant for intercept)
X = sm.add_constant(df['bmi'])

# Define dependent variable
y = df['target']

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

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

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 target   R-squared:                       0.344
Model:                            OLS   Adj. R-squared:                  0.342
Method:                 Least Squares   F-statistic:                     230.7
Date:                Sat, 07 Sep 2024   Prob (F-statistic):           3.47e-42
Time:                        19:23:41   Log-Likelihood:                -2454.0
No. Observations:                 442   AIC:                             4912.
Df Residuals:                     440   BIC:                             4920.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        152.1335      2.974     51.162      0.000     146.289     157.978
bmi          949.4353     62.515     15.187      0.000     826.570    1072.301
==============================================================================
Omnibus:                       11.674   Durbin-Watson:                   1.848
Prob(Omnibus):                  0.003   Jarque-Bera (JB):                7.310
Skew:                           0.156   Prob(JB):                       0.0259
Kurtosis:                       2.453   Cond. No.                         21.0
==============================================================================

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

cancer data set

In [26]:

```python
import pandas as pd

# Load dataset (replace 'your_dataset.csv' with the actual file path)
df = pd.read_csv(r"C:\Users\pavan\Downloads\archive (5)\cancer.csv")

# Display the first few rows
df.head()
```

Out[26]:

	Class	age	menopause	tumor-size	inv-nodes	node-caps	deg-malig	breast	breast-quad	irradiat
0	0	5	1	1	1	2	1	3	1	1
1	0	5	4	4	5	7	10	3	2	1
2	0	3	1	1	1	2	2	3	1	1
3	0	6	8	8	1	3	4	3	7	1
4	0	4	1	1	3	2	1	3	1	1

Calculate Statistical Measures

In [27]:

```python
# Calculate mean, median, mode, standard deviation, and variance for relevant featu
mean_values = df.mean()
median_values = df.median()
mode_values = df.mode().iloc[0]
std_values = df.std()
variance_values = df.var()
```

```python

# Display calculated statistics
print("Mean Values:\n", mean_values)
print("Median Values:\n", median_values)
print("Mode Values:\n", mode_values)
print("Standard Deviation Values:\n", std_values)
print("Variance Values:\n", variance_values)
```

```
Mean Values:
 Class          0.348974
age            4.442815
menopause      3.143695
tumor-size     3.208211
inv-nodes      2.826979
node-caps      3.233138
deg-malig      3.542522
breast         3.435484
breast-quad    2.868035
irradiat       1.604106
dtype: float64
Median Values:
 Class          0.0
age            4.0
menopause      1.0
tumor-size     1.0
inv-nodes      1.0
node-caps      2.0
deg-malig      1.0
breast         3.0
breast-quad    1.0
irradiat       1.0
dtype: float64
Mode Values:
 Class          0
age            1
menopause      1
tumor-size     1
inv-nodes      1
node-caps      2
deg-malig      1
breast         3
breast-quad    1
irradiat       1
Name: 0, dtype: int64
Standard Deviation Values:
 Class          0.476995
age            2.822781
menopause      3.061753
tumor-size     2.985140
inv-nodes      2.865457
node-caps      2.224523
deg-malig      3.646104
breast         2.438573
breast-quad    3.054599
irradiat       1.733792
dtype: float64
Variance Values:
 Class           0.227525
age             7.968091
menopause       9.374329
tumor-size      8.911063
inv-nodes       8.210842
node-caps       4.948504
deg-malig      13.294078
```

```
breast          5.946639
breast-quad     9.330577
irradiat        3.006033
dtype: float64
```

Hypothesis Testing For this example, let's conduct a hypothesis test to determine if the mean tumor size is significantly different from a chosen value (e.g., 5).

In [32]:

```
from scipy import stats

# Set the hypothesized mean
hypothesized_mean = 5

# Perform a one-sample t-test
t_statistic, p_value = stats.ttest_1samp(df['tumor-size'], hypothesized_mean)

# Display results
print(f'T-statistic: {t_statistic}, P-value: {p_value}')

# Determine significance
alpha = 0.05
if p_value < alpha:
    print("Reject the null hypothesis: The mean tumor size is significantly differe
else:
    print("Fail to reject the null hypothesis: The mean tumor size is not significa
```

```
T-statistic: -15.675242111482317, P-value: 1.6346262019129213e-47
Reject the null hypothesis: The mean tumor size is significantly different from 5.
```

Compute a 95% Confidence Interval Finally, we will compute a 95% confidence interval for the mean of the tumor size.

In [36]:

```
import numpy as np

# Calculate the mean and standard error
mean_tumor_size = df['tumor-size'].mean()
sem = stats.sem(df['tumor-size'])

# Calculate the confidence interval
ci = stats.t.interval(0.95, len(df['tumor-size']) - 1, loc=mean_tumor_size, scale=s

print(f'95% Confidence Interval for the mean tumor size: {ci}')
```

```
95% Confidence Interval for the mean tumor size: (np.float64(2.9837747849078164), n
p.float64(3.4326475024822125))
```

This code will help you perform the required analysis on the cancer dataset. Ensure that the column names used in the code match those in your dataset (like tumor-size). Adjust the hypotheses and features as needed based on your specific interests in the dataset. Exploring Regression Analysis on a New DatasetData Preprocessing We need to identify the dependent variable (target) and independent variables (features). For this example, let's say we want to analyze how age affects tumor-size.

In [39]:

```
# Check for missing values
df.isnull().sum()

# Display the data types
df.dtypes
```

Out[39]:

```
Class          int64
age            int64
menopause      int64
tumor-size     int64
inv-nodes      int64
node-caps      int64
deg-malig      int64
breast         int64
breast-quad    int64
irradiat       int64
dtype: object
```

Linear Regression Model We'll fit a linear regression model using age as the independent variable and tumor-size as the dependent variable.

In [41]:

```
import statsmodels.api as sm

# Define the independent variable (X) and dependent variable (y)
X = df['age']
y = df['tumor-size']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

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

# Display the model summary
model_summary = model.summary()
model_summary
```

Out[41]:

OLS Regression Results

Dep. Variable:	tumor-size	**R-squared:**	0.429
Model:	OLS	**Adj. R-squared:**	0.428
Method:	Least Squares	**F-statistic:**	511.2
Date:	Sun, 08 Sep 2024	**Prob (F-statistic):**	7.76e-85
Time:	12:59:31	**Log-Likelihood:**	-1521.9
No. Observations:	682	**AIC:**	3048.
Df Residuals:	680	**BIC:**	3057.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.1304	0.161	0.809	0.419	-0.186	0.447
age	0.6928	0.031	22.609	0.000	0.633	0.753

Omnibus:	87.179	**Durbin-Watson:**	1.894
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	121.385
Skew:	0.929	**Prob(JB):**	4.38e-27
Kurtosis:	3.907	**Cond. No.**	10.1

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

Model Summary Interpretation Coefficients: Indicate the change in the dependent variable for a one-unit change in the independent variable. P-values: Help determine the significance of each coefficient (typically, a p-value < 0.05 indicates significance). R-squared: Represents the proportion of variance in the dependent variable that can be explained by the independent variable(s). Visualization Let's create visualizations to illustrate the relationship between age and tumor-size along with the regression line.

In [50]:

```
pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\pavan\appdata\local\programs\p
ython\python312\lib\site-packages (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\pavan\appdata\local\prog
rams\python\python312\lib\site-packages (from matplotlib) (1.3.0)
Requirement already satisfied: cycler>=0.10 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\pavan\appdata\local\pro
grams\python\python312\lib\site-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\pavan\appdata\local\pro
grams\python\python312\lib\site-packages (from matplotlib) (1.4.7)
Requirement already satisfied: numpy>=1.23 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from matplotlib) (2.1.1)
Requirement already satisfied: packaging>=20.0 in c:\users\pavan\appdata\local\progr
ams\python\python312\lib\site-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\pavan\appdata\local\programs\py
thon\python312\lib\site-packages (from matplotlib) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\pavan\appdata\local\prog
rams\python\python312\lib\site-packages (from matplotlib) (3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\pavan\appdata\local
\programs\python\python312\lib\site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\pavan\appdata\local\programs\pyt
hon\python312\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [53]:

```
pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\pavan\appdata\local\programs\pyth
on\python312\lib\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\pavan\appdata\local
\programs\python\python312\lib\site-packages (from seaborn) (2.1.1)
Requirement already satisfied: pandas>=1.2 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from seaborn) (2.2.2)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\pavan\appdata\loc
al\programs\python\python312\lib\site-packages (from seaborn) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\pavan\appdata\local\prog
rams\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.
3.0)
Requirement already satisfied: cycler>=0.10 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\pavan\appdata\local\pro
grams\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.
53.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\pavan\appdata\local\pro
grams\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.
4.7)
Requirement already satisfied: packaging>=20.0 in c:\users\pavan\appdata\local\progr
ams\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (24.
1)
Requirement already satisfied: pillow>=8 in c:\users\pavan\appdata\local\programs\py
thon\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\pavan\appdata\local\prog
rams\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.
1.4)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\pavan\appdata\local
\programs\python\python312\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\pavan\appdata\local\programs
\python\python312\lib\site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\pavan\appdata\local\progra
ms\python\python312\lib\site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: six>=1.5 in c:\users\pavan\appdata\local\programs\pyt
hon\python312\lib\site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4-
>seaborn) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [56]:

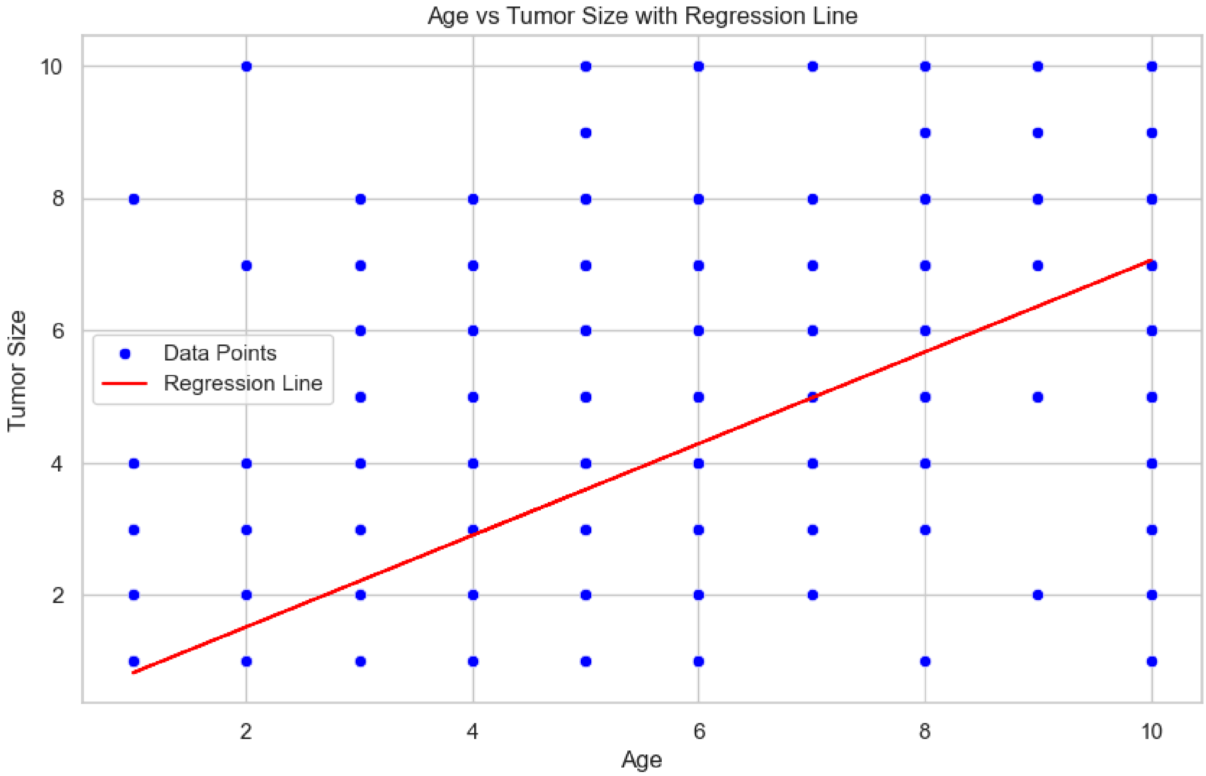
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Set the style
sns.set(style="whitegrid")

# Create a scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x='age', y='tumor-size', data=df, color='blue', label='Data Points'

# Plot the regression line
plt.plot(df['age'], model.predict(X), color='red', label='Regression Line')

# Add titles and labels
plt.title('Age vs Tumor Size with Regression Line')
plt.xlabel('Age')
plt.ylabel('Tumor Size')
plt.legend()
plt.show()
```



This analysis will provide insights into how age impacts tumor size in the dataset. You can interpret the model summary to understand the significance of the relationship and visualize it with the scatter plot and regression line.