Statistical Analysis in Python with Real-World Datasets

Statistical analysis is a fundamental aspect of data science and machine learning. By understanding and applying statistical methods, you can derive meaningful insights from datasets, make predictions, and inform decision-making. In this session, we will explore the basics of statistical analysis using Python, leveraging libraries like pandas, numpy, and scipy. We will use various datasets, including the built-in Diabetes dataset from the sklearn library, and explore how to apply statistical methods to datasets from sources like Kaggle and the UCI Machine Learning Repository.

Required libraries

Before we dive into coding, make sure you have Python installed along with the necessary libraries. You can install the required libraries using the following commands:

In [84]:
```
pip install numpy
```
```
Requirement already satisfied: numpy in c:\users\p.rahitya\anaconda3\lib\site-packages (1.26.4)
Note: you may need to restart the kernel to use updated packages.
```

In [85]:
```
pip install pandas
```
```
Requirement already satisfied: pandas in c:\users\p.rahitya\anaconda3\lib\site-packages (2.1.4)
Requirement already satisfied: numpy<2,>=1.23.2 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\p.rahitya\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [86]:
```
pip install scikit-learn
```
```
Requirement already satisfied: scikit-learn in c:\users\p.rahitya\anaconda3\lib\site-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in c:\users\p.rahitya\anaconda3\lib\site-packages (from scikit-learn) (1.26.4)
Requirement already satisfied: scipy>=1.3.2 in c:\users\p.rahitya\anaconda3\lib\site-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in c:\users\p.rahitya\anaconda3\lib\site-packages (from scikit-learn) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\p.rahitya\anaconda3\lib\site-packages (from scikit-learn) (2.2.0)
Note: you may need to restart the kernel to use updated packages.
```

In [87]:
```
pip install statsmodels
```
```
Requirement already satisfied: statsmodels in c:\users\p.rahitya\anaconda3\lib\site-packages (0.14.0)
Requirement already satisfied: numpy>=1.18 in c:\users\p.rahitya\anaconda3\lib\site-packages (from statsmodels) (1.26.4)
Requirement already satisfied: scipy!=1.9.2,>=1.4 in c:\users\p.rahitya\anaconda3\lib\site-packages (from statsmodels) (1.11.4)
Requirement already satisfied: pandas>=1.0 in c:\users\p.rahitya\anaconda3\lib\site-packages (from statsmodels) (2.1.4)
Requirement already satisfied: patsy>=0.5.2 in c:\users\p.rahitya\anaconda3\lib\site-packages (from statsmodels) (0.5.3)
Requirement already satisfied: packaging>=21.3 in c:\users\p.rahitya\anaconda3\lib\site-packages (from statsmodels) (23.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas>=1.0->statsmodels) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas>=1.0->statsmodels) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\p.rahitya\anaconda3\lib\site-packages (from pandas>=1.0->statsmodels) (2023.3)
Requirement already satisfied: six in c:\users\p.rahitya\anaconda3\lib\site-packages (from patsy>=0.5.2->statsmodels) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

Applying Statistical Methods

Datasets are available in Kaggle or the UCI Machine Learning Repository.

Let us choose a Health-related dataset like Heart Disease dataset from kaggle.

let us import the dataset using Pandas as dataframe.

In [88]:
```python
import pandas as pd
```

In [89]:
```python
data=pd.read_csv('heart.csv')
```

In [90]:
```python
data.head()
```

Out[90]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

let us consider the above dataset for Stastical Snalysis using Python

Descriptive Statistics

Descriptive statistics aim to summarize and organize data to make it easily understandable. This involves describing the central tendency, dispersion, and shape of the data distribution. Descriptive statistics provide a simple summary about the sample and the measures, without making inferences about the population from which the data was drawn.

1. Measures of Central Tendency: Describe the central point in a dataset.

a. Mean: The average value for a column

In [91]:
```python
print('Mean of the dataset:\n',data.mean())
```
```
Mean of the dataset:
 age          54.434146
sex           0.695610
cp            0.942439
trestbps    131.611707
chol        246.000000
fbs           0.149268
restecg       0.529756
thalach     149.114146
exang         0.336585
oldpeak       1.071512
slope         1.385366
ca            0.754146
thal          2.323902
target        0.513171
dtype: float64
```

b. Median: The middle value in a sorted dataset.

In [92]:
```python
print('Median of the dataset:\n',data.median())
```
```
Median of the dataset:
 age          56.0
sex           1.0
cp            1.0
trestbps    130.0
chol        240.0
fbs           0.0
restecg       1.0
thalach     152.0
exang         0.0
oldpeak       0.8
slope         1.0
ca            0.0
thal          2.0
target        1.0
dtype: float64
```

c. Mode: The most frequent value in the dataset.

In [93]:
```python
print('Mode of the dataset:\n',data.mode().iloc[0])
```
```
Mode of the dataset:
 age          58.0
sex           1.0
cp            0.0
trestbps    120.0
chol        204.0
fbs           0.0
restecg       1.0
thalach     162.0
exang         0.0
oldpeak       0.0
slope         1.0
ca            0.0
thal          2.0
target        1.0
Name: 0, dtype: float64
```

2. Measures of Dispersion: Describe the spread of the data.

a. Variance: A measure of how spread out the values are from the mean.

In [94]:
```python
print('Variance of the dataset:\n',data.var())
```
```
Variance of the dataset:
 age           82.306450
sex            0.211944
cp             1.060160
trestbps     306.835410
chol        2661.787109
fbs            0.127111
restecg        0.278655
thalach      529.263325
exang          0.223514
oldpeak        1.380750
slope          0.381622
ca             1.062544
thal           0.385219
target         0.250071
dtype: float64
```

b. Standard Deviation: The square root of the variance; gives insight into the spread of data.

In [95]:
```python
print('Standard Deviation of the dataset:\n',data.std())
```
```
Standard Deviation of the dataset:
 age          9.072290
sex          0.460373
cp           1.029641
trestbps    17.516718
chol        51.592510
fbs          0.356527
restecg      0.527878
thalach     23.005724
exang        0.472772
oldpeak      1.175053
slope        0.617755
ca           1.030798
thal         0.620660
target       0.500070
dtype: float64
```

c. Range: The difference between the maximum and minimum values.

In [96]:
```python
print('Range of the dataset:\n',data.max()-data.min())
```
```
Range of the dataset:
 age          48.0
sex           1.0
cp            3.0
trestbps    106.0
chol        438.0
fbs           1.0
restecg       2.0
thalach     131.0
exang         1.0
oldpeak       6.2
slope         2.0
ca            4.0
thal          3.0
target        1.0
dtype: float64
```

3.Skewness: It measures the asymmetry of the distribution of values in a dataset.

If the skewness is 0, the data is perfectly symmetric (like a normal distribution).

If the skewness is positive, the distribution is skewed to the right (longer tail on the right side).

If the skewness is negative, the distribution is skewed to the left (longer tail on the left side).

In [97]:
```python
print('Skewness of the dataset:\n',data.skew())
```
```
Skewness of the dataset:
 age        -0.248866
sex        -0.851449
cp          0.529455
trestbps    0.739768
chol        1.074073
fbs         1.971339
restecg     0.180440
thalach    -0.513777
exang       0.692655
oldpeak     1.210899
slope      -0.479134
ca          1.261189
thal       -0.524390
target     -0.052778
dtype: float64
```

4. Kurtosis: It measures the "tailedness" of the distribution, or how much data is concentrated in the tails compared to the normal distribution.

Mesokurtic: A kurtosis of 0 (after subtracting 3) indicates that the dataset has a normal tail distribution (same as a normal distribution).

Leptokurtic: A kurtosis greater than 0 indicates that the dataset has heavy tails (more outliers).

Platykurtic: A kurtosis less than 0 indicates light tails (fewer outliers).

In [98]:
```python
print('Kurtosis of the dataset:\n',data.kurt())
```
```
Kurtosis of the dataset:
 age        -0.525618
sex        -1.277531
cp         -1.149500
trestbps    0.991221
chol        3.996803
fbs         1.889859
restecg    -1.309614
thalach    -0.088822
exang      -1.523205
oldpeak     1.314471
slope      -0.647129
ca          0.701123
thal        0.250827
target     -2.001123
dtype: float64
```

Inferential Statistics:

Inferential statistics take data from a sample and make inferences or predictions about a population. The main goal of inferential statistics is to determine if the patterns observed in the sample are significant or merely due to chance.

1. Population vs Sample:

A population refers to the entire set of items or individuals of interest.

A sample is a subset of the population used for analysis. The idea is that if the sample is representative of the population, conclusions drawn from the sample can be generalized to the population.

2. Estimation:

Point Estimation: A single value (like a mean) is calculated from the sample and used to estimate the population parameter.

Confidence Interval: A range of values derived from the sample that is likely to contain the population parameter (e.g., 95% confidence intervals).

3. Hypothesis Testing:

Null Hypothesis ($H0$): The assumption that there is no significant difference or effect. Alternative Hypothesis ($H1$): The assumption that there is a significant difference or effect. P-value: The probability of observing the data if the null hypothesis is true. A low p-value (commonly less than 0.05) suggests rejecting the null hypothesis.

Types of Tests:

T-test: Compares the means of two groups.

Chi-square test: Tests for independence between categorical variables.

ANOVA: Tests the difference between means of three or more groups.

let us perforn T-test for our heart disease details dataset

T-test:

A t-test for the trestbps (resting blood pressure) column in the heart disease dataset can help determine whether the sample mean resting blood pressure is significantly different from a given population mean.

let us import stats module from scipy package

In [99]:
```python
from scipy import stats
```

Now, We want to test whether the mean resting blood pressure (trestbps) in the dataset is significantly different from a hypothetical population mean. Let's assume the population mean for normal resting blood pressure is 120 mm Hg.

Hypothesis:

Null Hypothesis ($H0$): The mean resting blood pressure (trestbps) of the patients is equal to 120 mm Hg.

Alternative Hypothesis ($H1$): The mean resting blood pressure is not equal to 120 mm Hg.

In [100…
```python
# Extract the 'trestbps' (resting blood pressure) column from the dataset
trestbps_values=data['trestbps']
```

In [101…
```python
# Hypothetical population mean for resting blood pressure (normal is around 120 mm Hg)
population_mean=120.0
```

In [102…
```python
# Perform one-sample t-test
t_stat,p_value=stats.ttest_1samp(trestbps_values,population_mean)
```

In [103…
```python
print(f'T-Stastics: {t_stat}')
print(f'P-Value: {p_value}')
```
```
T-Stastics: 21.22292673122529
P-Value: 3.925192364196875e-83
```

In [104…
```python
# let us take 95% confidence interval for the trestbps
alpha=0.05
if p_value<alpha:
    print('Reject Null Hypothesis(H0): The mean resting blood pressure (trestbps) of the patients is equal to 120 mm Hg.')
else:
    print('Fail to reject Null Hypothesis(H0): The mean resting blood pressure (trestbps) of the patients is equal to 120 mm Hg.')
```
```
Reject Null Hypothesis(H0): The mean resting blood pressure (trestbps) of the patients is equal to 120 mm Hg.
```

4. Confidence Interval:

A confidence interval is a range of values, derived from the sample data, that is likely to contain the true population parameter. Confidence intervals provide an estimate of the uncertainty associated with a sample statistic, allowing researchers to gauge the precision of their estimates.

Required libraies are pandas and scipy

Sample Mean: The average trestbps value from the dataset.

In [105…
```python
sample_mean=data['trestbps'].mean()
```

Standard Error: Measures the standard deviation of the sample mean estimate. It's calculated as the sample standard deviation divided by the square root of the sample size.

In [106…
```python
standard_error=stats.sem(trestbps_values)
```

The stats.norm.interval function returns the interval within which the true population mean is expected to lie with 95% confidence.

In [107…
```python
confidence_interval=stats.norm.interval(0.95,loc=sample_mean,scale=standard_error)
```

In [108…
```python
print(f'95% Condfidence Interval for mean Resting Blood Pressure(trestbps): {confidence_interval}')
```
```
95% Condfidence Interval for mean Resting Blood Pressure(trestbps): (130.5393515374625, 132.68406309668387)
```

5. Regression Analysis:

Linear Regression: Models the relationship between a dependent variable and one or more independent variables by fitting a linear equation.

Logistic Regression: Used when the dependent variable is binary (e.g., yes/no).

let us do Linear Regression Analysis for our data set

let us use cp (chest pain type) as a predictor in a linear regression model with the heart disease dataset, you'll follow a similar approach to the one previously outlined. However, note that cp is typically a categorical variable (often encoded as integers representing different types of chest pain).

Used Library: The statsmodels library in Python provides tools for statistical modeling and hypothesis testing. To use statsmodels, you typically import it using import statsmodels.api as sm. This library includes functions for linear regression, logistic regression, and various other statistical tests.

In [109…
```python
import statsmodels.api as sm
```

In [110…
```python
# Check the unique values in 'cp' to understand how it's encoded
print(data['cp'].unique())
```
```
[0 1 2 3]
```

In [111…
```python
# Define independent variable (add constant for intercept)
x=sm.add_constant(data['cp'])
```

In [112…
```python
# Define dependent variable
y=data['target']
```

In [113…
```python
# Fit linear regression model
model=sm.OLS(y,x).fit()
```

In [114…
```python
print(model)
```
```
<statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x0000025959B44A90>
```

In [115…
```python
print(model.summary())
```
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 target   R-squared:                       0.189
Model:                            OLS   Adj. R-squared:                  0.188
Method:                 Least Squares   F-statistic:                     238.6
Date:                Sun, 08 Sep 2024   Prob (F-statistic):           1.56e-48
Time:                        16:35:54   Log-Likelihood:                -636.16
No. Observations:                1025   AIC:                             1276.
Df Residuals:                    1023   BIC:                             1286.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.3141      0.019     16.463      0.000       0.277       0.352
cp             0.2112      0.014     15.445      0.000       0.184       0.238
==============================================================================
Omnibus:                      323.801   Durbin-Watson:                   1.934
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               49.943
Skew:                          -0.073   Prob(JB):                     1.43e-11
Kurtosis:                       1.928   Cond. No.                         2.46
==============================================================================

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

This analysis helps understand how different types of chest pain (as encoded by cp) influence the likelihood of heart disease. If cp is a categorical variable with multiple values, consider using dummy coding (one-hot encoding) for more detailed analysis.

Visualization: Scatter Plot with Regression Line

Creating visualizations can help illustrate the relationships between variables and the regression line effectively. let do it using Python libraries like Matplotlib and Seaborn

In [116…
```python
import matplotlib.pyplot as plt
```

In [117…
```python
import seaborn as sns
```

In [118…
```python
#setting figure size
plt.figure(figsize=(10,6))
```
Out[118…
```
<Figure size 1000x600 with 0 Axes>
<Figure size 1000x600 with 0 Axes>
```

In [119…
```python
sns.regplot(x='cp', y='target', data=data)
plt.title('Scatter Plot with Regression Line for CP vs. Target')
plt.xlabel('CP')
plt.ylabel('Target')
plt.show()
```