21.Point Estimation & Confidence Interval for Rare Elements Using NumPy

Program:-

```
import numpy as np
import pandas as pd
from scipy.stats import norm
import csv
# Create a dummy rare_elements.csv file for demonstration
with open('rare_elements.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(['concentration'])
    writer.writerow([10.5])
    writer.writerow([12.1])
    writer.writerow([9.8])
    writer.writerow([11.3])
    writer.writerow([13.0])
    writer.writerow([10.9])
    writer.writerow([11.7])
    writer.writerow([12.5])
    writer.writerow([10.1])
    writer.writerow([11.8])
data = pd.read_csv("rare_elements.csv")
values = data['concentration'].values
n = int(input("Enter sample size: "))
confidence = float(input("Enter confidence level (e.g., 0.95): "))
precision = float(input("Enter desired precision: "))
sample = np.random.choice(values, size=n, replace=False)
mean = np.mean(sample)
std = np.std(sample, ddof=1)
```

```
z = norm.ppf((1 + confidence) / 2)
margin = z * (std / np.sqrt(n))
lower = mean - margin
upper = mean + margin
print("\nEstimated Mean:", mean)
print("Confidence Interval:", (lower, upper))
print("Desired Precision:", precision)
```

Output:-

```
Enter sample size: 5
Enter confidence level (e.g., 0.95): 0.98
Enter desired precision: 4
Estimated Mean: 11.72
Confidence Interval: (np.float64(11.060366531885496), np.float64(12.379633468114505))
Desired Precision: 4.0
```

22.Customer Rating Confidence Interval Using Pandas

Program:-

```
import pandas as pd
from scipy.stats import t
import numpy as np
import csv
# Create a dummy customer_reviews.csv file for demonstration
with open('customer_reviews.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(['rating'])
    writer.writerow([4.5])
    writer.writerow([3.0])
    writer.writerow([5.0])
    writer.writerow([2.5])
    writer.writerow([4.0])
    writer.writerow([3.5])
    writer.writerow([4.5])
    writer.writerow([5.0])
    writer.writerow([3.0])
    writer.writerow([4.0])
data = pd.read_csv("customer_reviews.csv")
ratings = data['rating']
mean = ratings.mean()
std = ratings.std(ddof=1)
n = len(ratings)

confidence = 0.95
t_val = t.ppf((1 + confidence) / 2, df=n-1)
margin = t_val * (std / np.sqrt(n))
```

```
lower = mean - margin

upper = mean + margin

print("Average Rating:", mean)

print("95% Confidence Interval:", (lower, upper))
```

Output:-

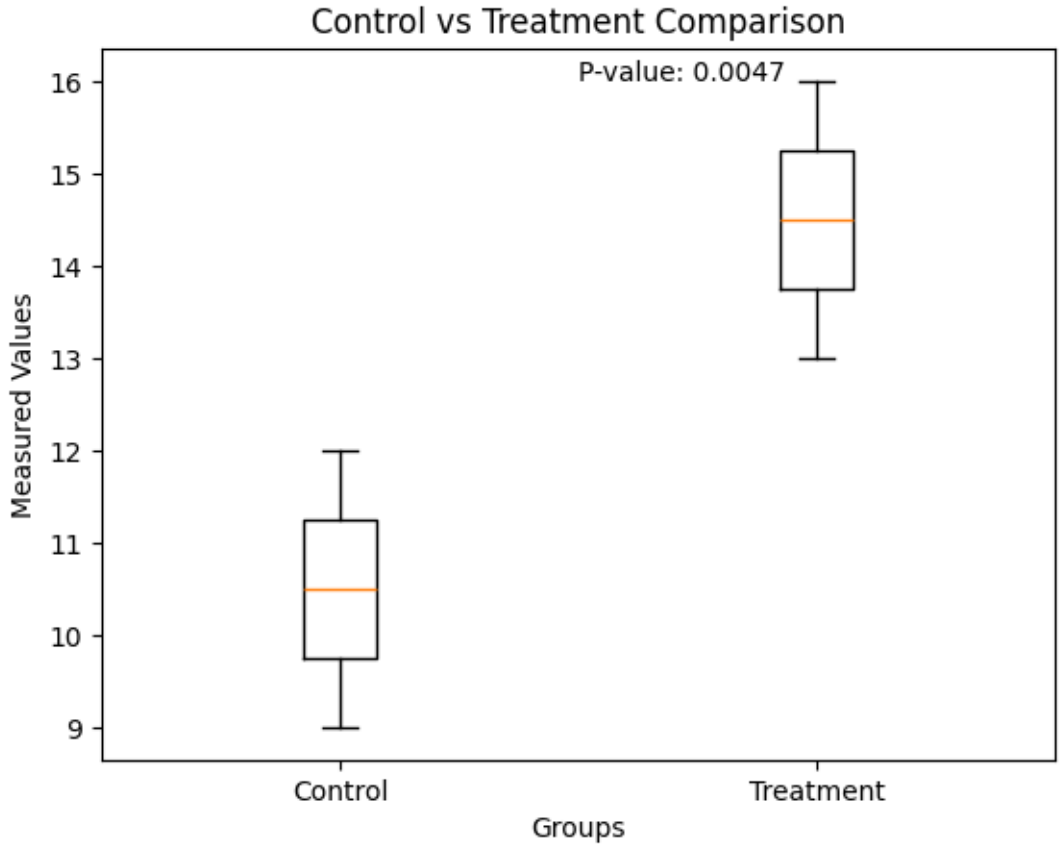
```
Average Rating: 3.9

95% Confidence Interval: (np.float64(3.2736370443121694),
np.float64(4.52636295568783))
```

23.Hypothesis Testing For New Medical Treatment With Visualization

Program:-

```
import pandas as pd
from scipy.stats import ttest_ind
import matplotlib.pyplot as plt
import csv
# Create a dummy clinical_trial.csv file for demonstration
with open('clinical_trial.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(['group', 'value'])
    writer.writerow(['control', 10])
    writer.writerow(['control', 12])
    writer.writerow(['control', 11])
    writer.writerow(['control', 9])
    writer.writerow(['treatment', 15])
    writer.writerow(['treatment', 14])
    writer.writerow(['treatment', 16])
    writer.writerow(['treatment', 13])
data = pd.read_csv("clinical_trial.csv")
control = data[data['group']=="control"]['value']
treatment = data[data['group']=="treatment"]['value']
t_stat, p_val = ttest_ind(control, treatment)
plt.boxplot([control, treatment], tick_labels=["Control", "Treatment"])
plt.title("Control vs Treatment Comparison")
plt.xlabel("Groups")
plt.ylabel("Measured Values")
plt.text(1.5, max(max(control), max(treatment)), f"P-value: {p_val:.4f}")
plt.show()
print("P-value:", p_val)
```

Output:-



P-value: 0.004659214943993936

Result: Statistically Significant Difference

24.K-Nearest Neighbors Classifier for Medical Condition Prediction

Program:-

```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
import csv
# Create a dummy patients_data.csv file for demonstration
with open('patients_data.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(['symptom1', 'symptom2', 'symptom3', 'condition'])
    writer.writerow([10, 20, 30, 0])
    writer.writerow([15, 25, 35, 0])
    writer.writerow([50, 60, 70, 1])
    writer.writerow([55, 65, 75, 1])
    writer.writerow([12, 22, 32, 0])
    writer.writerow([48, 58, 68, 1])
data = pd.read_csv("patients_data.csv")
X = data.drop("condition", axis=1)
y = data["condition"]
k = int(input("Enter value of k: "))
model = KNeighborsClassifier(n_neighbors=k)
model.fit(X, y)
print("Enter new patient symptom values:")
new_patient = []
for col in X.columns:
    val = float(input(f"{col}: "))
    new_patient.append(val)
```

Output:-

Enter value of k: 3

Enter new patient symptom values:

symptom1: 50

symptom2: 60

symptom3: 70

Prediction: Patient HAS the medical condition.

25.Decision Tree Classifier for Iris Flower Species Prediction

Program:-

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
iris = load_iris()
X = iris.data
y = iris.target
model = DecisionTreeClassifier()
model.fit(X, y)
print("Enter measurements:")
sl = float(input("Sepal length: "))
sw = float(input("Sepal width: "))
pl = float(input("Petal length: "))
pw = float(input("Petal width: "))
prediction = model.predict([[sl, sw, pl, pw]])
species = iris.target_names[prediction[0]]
print("Predicted Species:", species)
```

Output:-

```
Enter measurements:
Sepal length: 4
Sepal width: 5
Petal length: 6
Petal width: 3
Predicted Species: virginica
```