EXPERIMENT NO: 1B

1. Pandas

```
import pandas as pd
# Create a DataFrame
data = {'Name': ['Alice', 'Bob', 'Charlie'], 'Age': [25, 30, 35]}
df = pd.DataFrame(data)
# Filter rows where Age > 25
filtered_df = df[df['Age'] > 25]
print(filtered_df)
```

OUTPUT:

```
      Name  Age
1      Bob   30
2  Charlie   35
```

2. NumPy

```
import numpy as np
# Create a NumPy array
arr = np.array([1, 2, 3, 4, 5])# Calculate the mean of the array
mean_value = np.mean(arr)
print(mean_value)
```

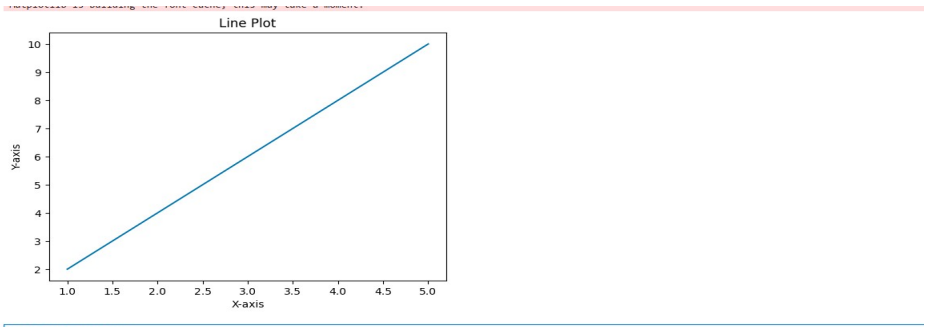
OUTPUT:

```
3.0
```

3. Matplotlib

```
import matplotlib.pyplot as plt
# Create a simple line plot
x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]
plt.plot(x, y)
plt.title("Line Plot")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

OUTPUT:

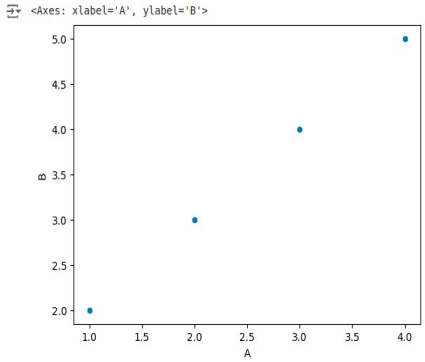


4. Seaborn

```
import seaborn as sns
```

```
import pandas as pd
# Create a DataFrame
data = {'A': [1, 2, 3, 4], 'B': [2, 3, 4, 5]}
df = pd.DataFrame(data)# Create a scatter plot
sns.scatterplot(x='A', y='B', data=df)
```

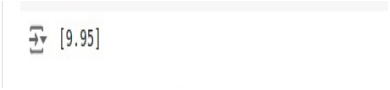
OUTPUT:



5. Scikit-learn

```
from sklearn.linear_model import LinearRegression
import numpy as np
# Sample data
X = np.array([[1], [2], [3], [4]])
y = np.array([2.2, 4.3, 6.1, 8.0])
# Train a linear regression model
model = LinearRegression()
model.fit(X, y)
# Predict
prediction = model.predict([[5]])
print(prediction)
```

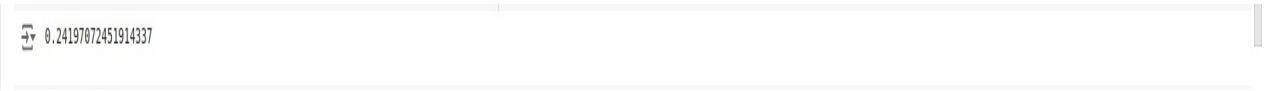
OUTPUT:



6. SciPy

```
from scipy.stats import norm# Calculate the probability density of a normal distribution
x = 1
prob_density = norm.pdf(x, loc=0, scale=1)
print(prob_density)
```

OUTPUT:

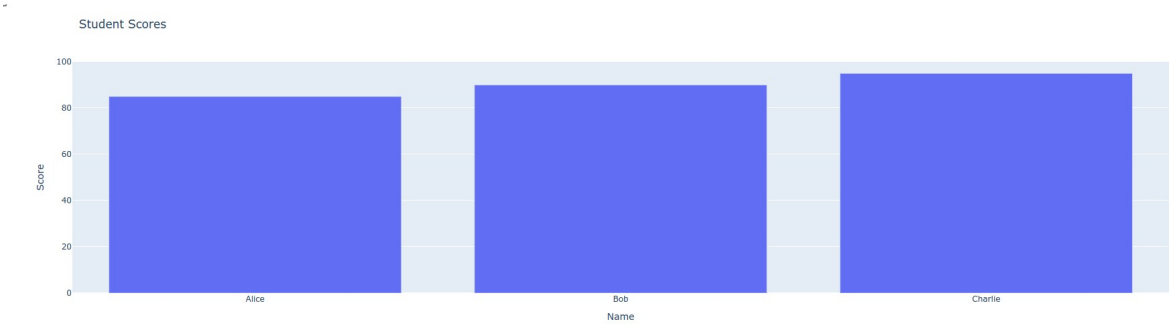


7. Plotly

```
import plotly.express as px
# Create a bar chart
data = {'Name': ['Alice', 'Bob', 'Charlie'], 'Score': [85, 90, 95]}
```

fig = px.bar(data, x='Name', y='Score', title='Student Scores')
fig.show()

OUTPUT:



8. Statsmodels

```
import statsmodels.api as sm

# Sample data
X = [1, 2, 3, 4]
y = [2.2, 4.3, 6.1, 8.0]

# Add a constant term for the intercept
X = sm.add_constant(X)

# Fit an Ordinary Least Squares (OLS) model
model = sm.OLS(y, X).fit()

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

OUTPUT:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.999
Model:                            OLS   Adj. R-squared:                  0.999
Method:                 Least Squares   F-statistic:                     2048.
Date:                Wed, 15 Jan 2025   Prob (F-statistic):           0.000488
Time:                        09:46:00   Log-Likelihood:                 5.1316
No. Observations:                   4   AIC:                            -6.263
Df Residuals:                       2   BIC:                            -7.491
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.3500      0.116      3.012      0.095      -0.150       0.850
x1             1.9200      0.042     45.255      0.000       1.737       2.103
==============================================================================
Omnibus:                        nan   Durbin-Watson:                   2.622
Prob(Omnibus):                  nan   Jarque-Bera (JB):                0.552
Skew:                         0.795   Prob(JB):                        0.759
Kurtosis:                     2.114   Cond. No.                         7.47
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
/usr/local/lib/python3.10/dist-packages/statsmodels/stats/stattools.py:74: ValueWarning:

omni_normtest is not valid with less than 8 observations; 4 samples were given.
```

9. TensorFlow and PyTorch

```
import tensorflow as tf
# Create a simple tensor
tensor = tf.const
```

OUTPUT:

```
tf.Tensor(
[[1 2]
 [3 4]], shape=(2, 2), dtype=int32)
```

10. OpenCV

```
from google.colab.patches import cv2_imshow
!curl -o logo.png https://encrypted-tbn0.gstatic.com/images?
q=tbn:ANd9GcRW9h4fU35Nb2lGTCcT-GS1pVkWd5gqV0yegw&simport cv2
img = cv2.imread('logo.png', cv2.IMREAD_UNCHANGED)
cv2_imshow(img)
```

OUTPUT:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 76909  100 76909    0     0   519k      0 --:--:-- --:--:-- --:--:--  517k
```