Somaiya Vidyavihar University

(A Constituent College of Somaiya Vidyavihar University)

Batch: A1	**Roll No.: 16010121045**
Experiment / Assignment / Tutorial No. 7	

Title: Implementation of input modeling steps for simulation

Objective: To understand and implement the process of developing input models for simulation by following the four essential steps: data collection, identifying probability distributions, choosing parameters for the distributions, and evaluating the goodness of fit. The experiment aims to highlight the significance of high-quality input data in producing reliable simulation outputs.

Expected Outcome of Experiment:

CO4: Analyze the systems for input modeling and validation.

CO5: Estimate the different parameters of absolute and relative performance of different simulation systems.

Books/ Journals/ Websites referred:

1. "Discrete-Event System Simulation" by Jerry Banks, John S. Carson II, Barry L. Nelson, David M. Nicol.
2. SimPy Documentation: https://simpy.readthedocs.io/en/latest/
3. SciPy Documentation: https://docs.scipy.org/doc/scipy/

Background:

(Explain in brief steps for input modeling development in simulation.)

Problem Statement:

Perform the following steps of Input Model Development:

1. **Collect Data from the Real System:**
   - Identify the key input processes in the system (e.g., arrival times, service times).
   - Collect data through direct observation, historical records, or sensors.
   - Ensure data is recorded accurately and is sufficient in quantity to perform statistical analysis.

2. **Identify a Probability Distribution to Represent the Input Process:**
   - Plot the collected data to visually inspect its distribution (e.g., using histograms).
   - Use statistical software or tools to fit different probability distributions to the data.

- Compare the fit of different distributions using statistical measures (e.g., likelihood, AIC).

3. **Choose Parameters for the Distribution:**
   - Use statistical software or tools to estimate the parameters of the chosen distribution.
   - Ensure that the estimated parameters make sense in the context of the real system.

4. **Evaluate the Chosen Distribution and Parameters for Goodness of Fit:**
   - Perform goodness-of-fit tests such as the Kolmogorov-Smirnov test, Chi-square test.
   - Compare the observed data with the expected values from the chosen distribution.

<u>Implementation Steps with Screen shots:</u>

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

df = pd.read_csv('tsla_2014_2023.csv')

# 2. Plot the volume column
plt.hist(df['volume'], bins=100, density=True, alpha=0.7, color='g')
plt.title("Histogram of Tesla Stock volumes")
plt.xlabel("volume")
plt.ylabel("Density")
plt.show()

# 3. Fit Probability Distributions: Fit Normal and Exponential distributions to the volume data
fit_expon = stats.expon.fit(df['volume'])
fit_lognorm = stats.lognorm.fit(df['volume'])
```

```python
# Plot the fitted distributions
x = np.linspace(min(df['volume']), max(df['volume']), 100)
pdf_expon = stats.expon.pdf(x, *fit_expon)
pdf_lognorm = stats.lognorm.pdf(x, *fit_lognorm)

plt.hist(df['volume'], bins=100, density=True, alpha=0.6, color='gray')
plt.plot(x, pdf_expon, 'r-', label="Exponential Fit")
plt.plot(x, pdf_lognorm, 'b-', label="Log Normal Fit")
plt.title("Fitting Distributions to volumes")
plt.xlabel("volume")
plt.ylabel("Density")
plt.legend()
plt.show()

# 4. Extract Parameters for the Fitted Distributions
print(f"Exponential Fit Parameters (loc, scale): {fit_expon}")
print(f"Log-Normal Fit Parameters (shape, loc, scale): {fit_lognorm}")

# 5. Goodness-of-Fit Testing using Kolmogorov-Smirnov (KS) test
ks_expon = stats.kstest(df['volume'], 'expon', args=fit_expon)
print(f"Kolmogorov-Smirnov test for Exponential: {ks_expon}")

ks_lognorm = stats.kstest(df['volume'], 'lognorm', args=fit_lognorm)
print(f"Kolmogorov-Smirnov test for Log Normal: {ks_lognorm}")

# 6. Chi-square goodness-of-fit test

# Observed frequencies (histogram data)
observed_freq, bins = np.histogram(df['volume'], bins=100)
```

```
# Expected frequencies (Exponential and Log-Normal)
expected_freq_expon = stats.expon.cdf(bins[1:], *fit_expon) -
stats.expon.cdf(bins[:-1], *fit_expon)
expected_freq_lognorm = stats.lognorm.cdf(bins[1:], *fit_lognorm) -
stats.lognorm.cdf(bins[:-1], *fit_lognorm)

# Normalize expected frequencies to match the observed sum
expected_freq_expon *= len(df['volume'])
expected_freq_lognorm *= len(df['volume'])

# Scale expected frequencies to match the total observed sum
expected_freq_expon *= observed_freq.sum() / expected_freq_expon.sum()
expected_freq_lognorm *= observed_freq.sum() / expected_freq_lognorm.sum()

# Perform the chi-square test
chi_square_expon = stats.chisquare(f_obs=observed_freq,
f_exp=expected_freq_expon)
print(f"Chi-square test for Exponential: {chi_square_expon}")

chi_square_lognorm = stats.chisquare(f_obs=observed_freq,
f_exp=expected_freq_lognorm)
print(f"Chi-square test for Log-Normal: {chi_square_lognorm}")
```

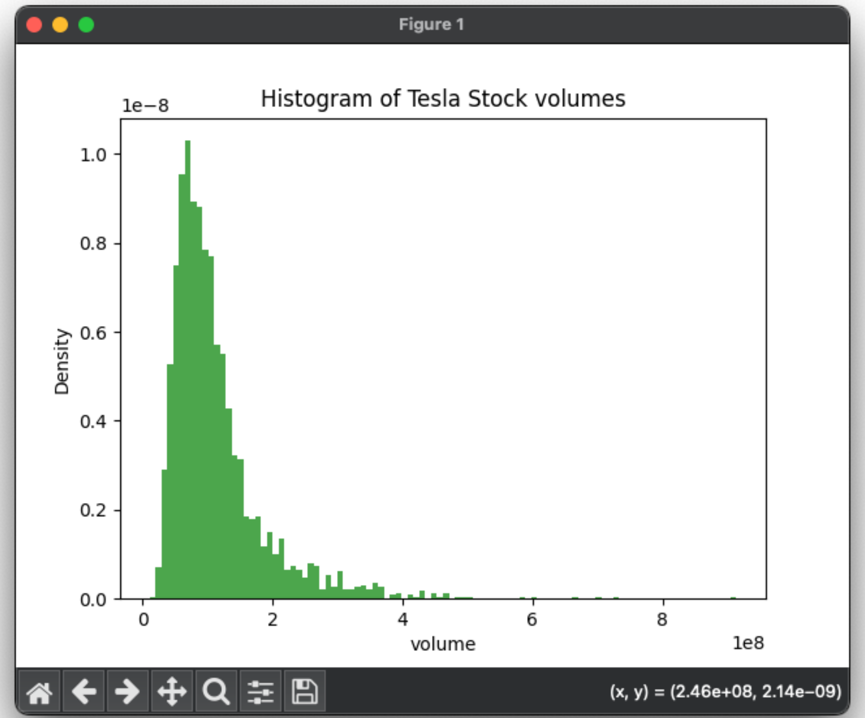
Figure 1
Histogram of Tesla Stock volumes
1e−8
Density
volume
1e8
(x, y) = (2.46e+08, 2.14e−09)

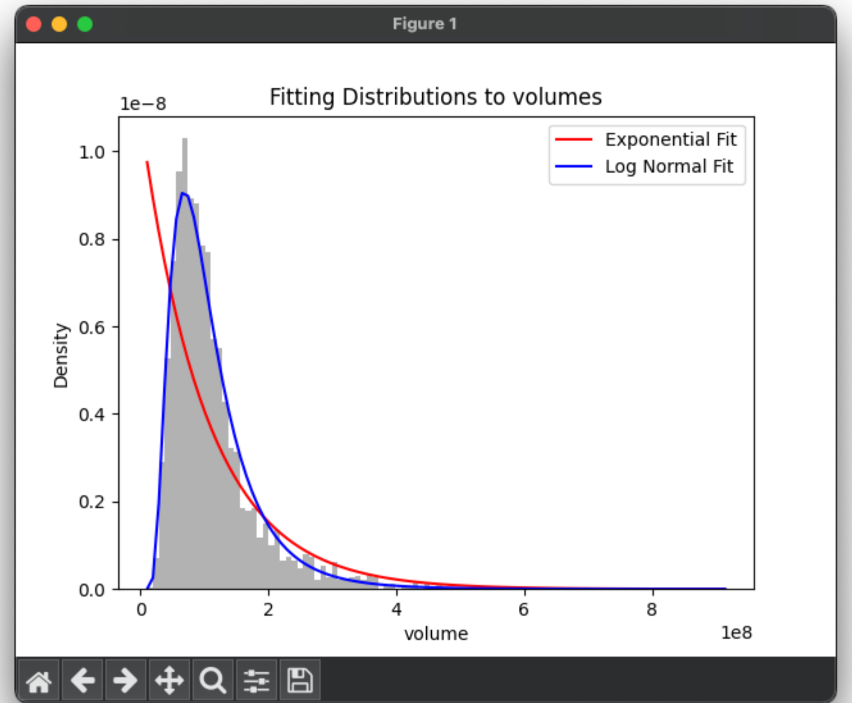
Figure 1
Fitting Distributions to volumes
1e−8
Exponential Fit
Log Normal Fit
Density
volume
1e8

```
❯ python3 -u "/Users/pargatsinghdhanjal/Sem 7/CSM/Code/Exp7.py"
2024-10-11 02:00:28.057 Python[68996:3056922] +[IMKClient subclass]: chose IMKClient_Legacy
2024-10-11 02:00:28.057 Python[68996:3056922] +[IMKInputSession subclass]: chose IMKInputSession_Legacy
Exponential Fit Parameters (loc, scale): (10620000.0, 102578599.20508744)
Log-Normal Fit Parameters (shape, loc, scale): (np.float64(0.5986118999925562), 7327288.557954236, np.float64(87775141.30054902))
Kolmogorov-Smirnov test for Exponential: KstestResult(statistic=np.float64(0.21853297505643754), pvalue=np.float64(5.540307966133598e-106), statistic_location=np.int64(49744500), statistic_sign=np.int8(-1))
Kolmogorov-Smirnov test for Log Normal: KstestResult(statistic=np.float64(0.02963700103775457З), pvalue=np.float64(0.023589481479335293), statistic_location=np.int64(122659500), statistic_sign=np.int8(1))
Chi-square test for Exponential: Power_divergenceResult(statistic=np.float64(1123.743858306693), pvalue=np.float64(2.7590469624684115e-173))
Chi-square test for Log-Normal: Power_divergenceResult(statistic=np.float64(279.4565318763716), pvalue=np.float64(4.0540048923564866e-19))
```

Conclusion:

The **log-normal distribution** provides a much better fit for the Tesla stock volumes than the **exponential distribution**, as evidenced by lower Kolmogorov-Smirnov (KS) statistics and chi-square statistics.
However, the **log-normal fit** is a reasonable fit but not perfect. The small p-values from the tests indicate that even the log-normal distribution does not perfectly describe the stock volumes, but it's closer to the actual data than the exponential distribution.
The large differences found in the **exponential distribution** suggest that it is **not a good fit** for modeling the volume of Tesla stock trades.

Post lab Questions:

1. **Explore the concept of multivariate input models. How would you approach input modeling if the input data involved multiple correlated variables?**

Multivariate input modeling involves capturing the relationships between multiple correlated variables. Key steps include exploring correlations (e.g., using a correlation matrix), visualizing relationships (e.g., scatterplot matrix), and selecting appropriate multivariate distributions. Techniques like Principal Component Analysis (PCA) can help reduce dimensionality while preserving the data's structure.