Assignment – 3

```
import pandas as pd

from collections import defaultdict


# Step 1: Load dataset

df = pd.read_csv("forestfires.csv")

df.head()


# Step 2: Mapper – emit (month, temperature) pairs

mapped = []

for _, row in df.iterrows():

    mapped.append((row["Month"], row["Temperature_Celsius"]))  # month, temperature


print("Sample mapped data:", mapped[:5])


# Step 3: Shuffle & Sort – group by month

grouped = defaultdict(list)

for month, temp in mapped:

    grouped[month].append(temp)


# Step 4: Reducer – calculate average temperature per month

results = {month: sum(temps)/len(temps) for month, temps in grouped.items()}


# Step 5: Display output

print("Average Temperature by Month:")

for m, avg in results.items():

    print(f"{m}: {avg:.2f}")
```

```
# Simulating Hive query using pandas groupby

hive_result = df.groupby("Month")[["Temperature_Celsius", "Burned_Area_hectares"]].mean().reset_index()

hive_result.columns = ["Month", "Avg_Temp", "Avg_Burned_Area"]

hive_result
```

Assignment-4

```
import seaborn as sns

import matplotlib.pyplot as plt

import pandas as pd

from mpl_toolkits import mplot3d

import networkx as nx

import squarify


iris = sns.load_dataset("iris")


# 1D Visualization

sns.countplot(x="species", data=iris)

plt.title("1D - Count of Iris Species")

plt.show()


# 2D Visualization

sns.scatterplot(x="petal_length", y="petal_width", hue="species", data=iris)

plt.title("2D - Petal Length vs Petal Width")

plt.show()


# 3D Visualization

fig = plt.figure()
```

```
ax = fig.add_subplot(111, projection='3d')
ax.scatter(iris['sepal_length'], iris['sepal_width'], iris['petal_length'])
plt.title("3D - Iris Data")
plt.show()

# Temporal Visualization (dummy data)
time = pd.DataFrame({
    "Month": pd.date_range("2024-01-01", periods=6, freq="M"),
    "Value": [10, 15, 20, 18, 25, 30]
})
plt.plot(time["Month"], time["Value"], marker="o")
plt.title("Temporal - Value Over Time")
plt.show()

# Multidimensional Visualization
sns.scatterplot(x="sepal_length", y="petal_length",
                hue="species", size="sepal_width", data=iris)
plt.title("Multidimensional - Iris Features")
plt.show()

# Tree / Hierarchical Visualization
data = pd.DataFrame({
    "Category": ["A", "B", "C", "D"],
    "Value": [30, 15, 25, 10]
})
squarify.plot(sizes=data["Value"], label=data["Category"], alpha=0.7)
plt.title("Hierarchical - Treemap Example")
plt.axis("off")
plt.show()
```

```
# Network Visualization
G = nx.Graph()
G.add_edges_from([("A", "B"), ("A", "C"), ("B", "D"), ("C", "E")])
nx.draw(G, with_labels=True, node_color='lightblue', node_size=800)
plt.title("Network Visualization")
plt.show()
```

Experiement-5

```
# Import libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# Step 1: Load dataset
data = pd.read_csv("Admission.csv")   # your CSV file name
data.head()

# Step 2: Select features (inputs) and target (output)
X = data[["GRE Score", "TOEFL Score", "University Rating", "SOP", "LOR", "CGPA", "Research"]]
y = data["Chance of Admit"]

# Step 3: Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 4: Create and train the model
model = LinearRegression()
model.fit(X_train, y_train)
```

```
# Step 5: Make predictions
y_pred = model.predict(X_test)

# Step 6: Evaluate model
score = r2_score(y_test, y_pred)
print("Model Accuracy (R² Score):", round(score, 3))

# Step 7: Test with your own values
sample = [[320, 110, 4, 4.0, 4.5, 9.2, 1]]  # Example student
predicted = model.predict(sample)
print("Predicted Chance of Admit:", round(predicted[0], 3))
```