# Name : Sandeep Sahebrao Solanke Division: CM Batch: CM4 Roll no: 87 PRN: 202401090099

**Problem statements for a given dataset using Numpy and Pandas and Apply Numpy and pandas methods:  
  
1. Sales Dataset**

**Problem Statemen**t: Calculate the monthly sales trends and visualize them.  
 **Solution**:

import pandas as pd

import matplotlib.pyplot as plt

# Load the dataset

df = pd.read\_csv("sales\_dataset.csv")

# Group sales by month

monthly\_sales = df.groupby("Month")["Sales"].sum()

# Plot the trend

plt.figure(figsize=(10, 6))

monthly\_sales.plot(kind="line", marker="o")

plt.title("Monthly Sales Trend")

plt.xlabel("Month")

plt.ylabel("Total Sales")

plt.grid()

plt.show()

**2. COVID-19 Dataset**

**Problem Statement:** Find the country with the highest total number of confirmed cases.  
 **Solution:**

# Group by country and sum confirmed cases

total\_cases = df.groupby("Country")["Confirmed"].sum()

# Find the country with the maximum cases

highest\_cases\_country = total\_cases.idxmax()

print("Country with highest confirmed cases:", highest\_cases\_country)

**3. FIFA Dataset**

**Problem Statement:** Find the top 5 players with the highest overall ratings.  
 **Solution:**# Sort players by Overall Ratings in descending order

top\_players = df.sort\_values(by="Overall", ascending=False).head(5)

print("Top 5 Players:\n", top\_players[["Name", "Overall"]])

**4. Cricket World Cup Dataset**

**Problem Statement:** Calculate the batting average for each player.  
 **Solution:**

# Calculate Batting Average = Total Runs / Total Innings

df["Batting Average"] = df["Total Runs"] / df["Total Innings"]

print(df[["Player", "Batting Average"]])

**5. IPL Dataset**

**Problem Statement:** Identify the team with the most wins.  
 **Solution:**

# Group by Team and count Wins

team\_wins = df.groupby("Team")["Wins"].sum()

# Find the team with the most wins

top\_team = team\_wins.idxmax()

print("Team with most wins:", top\_team)

**6. Kaggle Text Classification Dataset**

**Problem Statement:** Analyze the frequency of each word in the dataset.  
 **Solution:**

from collections import Counter

# Combine all text

all\_text = " ".join(df["Text"])

# Split and count frequency

word\_freq = Counter(all\_text.split())

print("Word Frequencies:\n", word\_freq.most\_common(10))

**Explanation:** Word frequencies are counted using Python's collections.Counter

**7. Movie Review Dataset**

**Problem Statement:** Calculate the average review sentiment score for each genre.  
 **Solution:**

# Group by Genre and calculate average sentiment score

avg\_sentiment = df.groupby("Genre")["Sentiment Score"].mean()

print("Average Sentiment Score by Genre:\n", avg\_sentiment)

**8. Amazon Product Dataset**

**Problem Statement:** Find the top-rated product in each category.  
 **Solution:**

# Sort within each category by Rating

top\_products = df.sort\_values(by=["Category", "Rating"], ascending=[True, False]).groupby("Category").head(1)

print("Top-rated Products:\n", top\_products[["Category", "Product Name", "Rating"]])

**9. Grocery Dataset**

**Problem Statement:** Find the most purchased item across all stores.  
 **Solution:**

# Group by Item and sum purchases

total\_purchases = df.groupby("Item")["Quantity"].sum()

# Identify the most purchased item

most\_purchased\_item = total\_purchases.idxmax()

print("Most Purchased Item:", most\_purchased\_item)

**10. House Price Dataset**

**Problem Statement:** Analyze the correlation between house size and price.  
 **Solution:**

# Calculate correlation

correlation = df["Size"].corr(df["Price"])

print("Correlation between Size and Price:", correlation)

**Explanation:** The correlation coefficient measures the relationship between size and price.

**11. Paper Review Dataset**

**Problem Statement:** Identify the average number of citations per research paper.  
 **Solution:**

# Calculate the average citations

avg\_citations = df["Citations"].mean()

print("Average Citations per Paper:", avg\_citations)

**12. Twitter US Airline Sentiment**

**Problem Statement:** Count the number of positive, negative, and neutral tweets for each airline.  
 **Solution:**

# Group by Airline and Sentiment, and count occurrences sentiment\_count = df.groupby(["Airline", "Sentiment"])["Tweet"].count() print("Sentiment Counts by Airline:\n", sentiment\_count)

**13. Goodreads Book Reviews**

**Problem Statement:** Find the book with the highest average review score and at least 500 reviews.  
 **Solution:**

# Filter books with at least 500 reviews filtered\_books = df[df["Review Count"] >= 500] # Find the book with the highest average review score top\_book = filtered\_books.loc[filtered\_books["Average Score"].idxmax()] print("Top-rated Book:\n", top\_book[["Title", "Average Score", "Review Count"]])

**14. IMDB Dataset**

**Problem Statement:** Analyze the most frequently occurring genres in movies. **Solution:**

# Split Genres column and count frequencies

from collections import Counter

genre\_list = df["Genres"].str.split(",").sum()

genre\_count = Counter(genre\_list)

print("Most Frequent Genres:\n", genre\_count.most\_common(5))

**15. MovieLens Latest Datasets**

**Problem Statement:** Calculate the average rating for each movie.  
 **Solution:**

# Group by Movie and calculate the average rating avg\_rating = df.groupby("Movie")["Rating"].mean() print("Average Rating by Movie:\n", avg\_rating)

# Group by Movie and calculate the average rating

avg\_rating = df.groupby("Movie")["Rating"].mean()

print("Average Rating by Movie:\n", avg\_rating)

**16. SMS Spam Collection**

**Problem Statement:** Calculate the ratio of spam to ham messages.  
 **Solution:**

# Count spam and ham messages

spam\_count = df[df["Label"] == "spam"].shape[0]

ham\_count = df[df["Label"] == "ham"].shape[0]

spam\_ratio = spam\_count / ham\_count

print("Spam-to-Ham Ratio:", spam\_ratio)

**17. WordNet**

**Problem Statement:** Find the word with the highest number of synonyms.  
 **Solution:**

# Identify the word with the most synonyms

most\_synonyms\_word = df.loc[df["Synonym Count"].idxmax()]

print("Word with Most Synonyms:\n", most\_synonyms\_word[["Word", "Synonym Count"]])

**18. Yelp Reviews**

**Problem Statement:** Analyze the average review score for each business category.  
 **Solution:**

# Group by Business Category and calculate average score

avg\_review\_score = df.groupby("Category")["Review Score"].mean()

print("Average Review Score by Business Category:\n", avg\_review\_score)

**19. The Blog Authorship Corpus**

**Problem Statement:** Identify the author with the highest number of blog posts.  
 **Solution:**

# Count blog posts for each author author\_posts = df.groupby("Author")["Blog Post"].count() # Find the author with the most blog posts top\_author = author\_posts.idxmax() print("Author with Most Blog Posts:", top\_author)

**20. Weather Dataset**

**Problem Statement:** Identify the hottest day and its temperature.  
 **Solution:**

# Find the row with the maximum temperature

hottest\_day = df.loc[df["Temperature"].idxmax()]

print("Hottest Day:\n", hottest\_day[["Date", "Temperature"]])