1. Average GPA by Class: What is the average GPA for each class?

import pandas as pd

import numpy as np

import os

import glob

import matplotlib.pyplot as plt

import seaborn as sns

from plotly import \_\_version\_\_

from plotly.offline import download\_plotlyjs, init\_notebook\_mode, plot, iplot

import cufflinks as cf

data=pd.read\_csv("project 4 data.csv")

data.head(10) #for checking values are exists or not

data.info() #checking information

data.isnull().any().sum() #Is there any null value or not

data.isnull().any()

data.dropna(inplace=True) #deleting all duplicate row with values

duplicates=data[data.duplicated()]

duplicates

#display duplicates rows

if not duplicates.empty:

    print("Duplicates not found")

    print(duplicates)

else:

    print("Not found.")

data.head() #check 5 rows from the beginning

import pandas as pd

# Assuming 'data' is in our DataFrame loaded from a CSV file

try:

    print(type(data))

except NameError:

    print("The variable 'data' is not defined.")

data.head()  # again display the first few rows of the DataFrame

print(data.columns) #seeing columns

# Group the data by class and calculate the average GPA for each class

average\_gpa\_by\_class = data.groupby('class')['GPA'].mean().reset\_index()

# Display the results

print(average\_gpa\_by\_class)

Answer is: class GPA

0 A 82.844605

1 B 83.251724

1. Gender Performance Comparison: How do male and female students compare in terms of average GPA?

# Calculate average GPA for male and female students

avg\_gpa\_by\_gender = data.groupby('gender')['GPA'].mean()

# Print the results

print("Average GPA for Male Students:", avg\_gpa\_by\_gender['male'])

print("Average GPA for Female Students:", avg\_gpa\_by\_gender['female'])

Answer is:

Average GPA for Male Students: 81.85216216216217

Average GPA for Female Students: 83.55823529411765

1. Race Performance Analysis: What is the average GPA for students of different races?

# Calculate the average GPA for students of different races

avg\_gpa\_by\_race = data.groupby('race')['GPA'].mean()

print(avg\_gpa\_by\_race)

Answer is:

race

1 83.909012

2 80.156667

3 81.388333

4 78.897500

5 73.400000

6 83.955000

7 81.592000

Name: GPA, dtype: float64

1. Course Performance: How do students perform in different courses (Algebra, Calculus1, Calculus2, Statistics, Probability, Measure, Functional\_analysis)?

# Calculate statistics for each course

statistics = {}

courses = ["Algebra", "Calculus1", "Calculus2", "Statistics", "Probability", "Measure", "Functional\_analysis"]

for course in courses:

    statistics[course] = {

        'mean': data[course].mean(),

        'median': data[course].median(),

        'std': data[course].std()

    }

# Display the statistics

for course, stats in statistics.items():

    print(f"{course} - Mean: {stats['mean']}, Median: {stats['median']}, Standard Deviation: {stats['std']}")

Answer is:

Algebra - Mean: 76.05714285714286, Median: 76.0,

Standard Deviation: 11.722618317585036

Calculus1 - Mean: 71.96190476190476, Median: 73.0,

Standard Deviation: 12.197039213176515

Calculus2 - Mean: 78.94285714285714, Median: 83.0,

Standard Deviation: 14.997325768942282

Statistics - Mean: 85.13333333333334, Median: 87.0,

Standard Deviation: 10.269509263485793

Probability - Mean: 83.87619047619047, Median: 85.0,

Standard Deviation: 10.51436310490709

Measure - Mean: 80.76190476190476, Median: 81.0,

Standard Deviation: 10.296119330120122

Functional\_analysis - Mean: 75.32380952380953, Median: 76.0, Standard Deviation: 13.003324455170949

1. Correlation with GPA: What is the correlation between GPA and performance in individual courses?

import pandas as pd

# Load the data

data = pd.read\_csv("project 4 data.csv")

# Select only numeric columns

numeric\_data = data.select\_dtypes(include=['number'])

# Calculate the correlation matrix

correlation\_matrix = numeric\_data.corr()

print(correlation\_matrix)

Answer is:

ID race GPA Algebra Calculus1

ID 1.000000 -0.199560 0.688738 0.751647 0.502151

race -0.199560 1.000000 -0.246571 -0.249298 -0.192550

GPA 0.688738 -0.246571 1.000000 0.761590 0.660560

Algebra 0.751647 -0.249298 0.761590 1.000000 0.608083

Calculus1 0.502151 -0.192550 0.660560 0.608083 1.000000

Calculus2 0.610048 -0.173228 0.803865 0.640578 0.588929

Statistics 0.925148 -0.179594 0.740903 0.759595 0.547605

Probability 0.627886 -0.229312 0.732898 0.752480 0.484766

Measure 0.635982 -0.186478 0.758676 0.755499 0.471347

Functional\_analysis 0.464508 -0.102435 0.809451 0.572575 0.430825

from4 -0.112189 -0.154352 -0.052361 -0.078426 0.034589

y 0.319634 0.011883 0.292536 0.257051 0.125429

GPA.1 0.688738 -0.246571 1.000000 0.761590 0.660560

ID.1 1.000000 -0.199560 0.688738 0.751647 0.502151

Calculus2 Statistics Probability Measure \

ID 0.610048 0.925148 0.627886 0.635982

race -0.173228 -0.179594 -0.229312 -0.186478

GPA 0.803865 0.740903 0.732898 0.758676

Algebra 0.640578 0.759595 0.752480 0.755499

Calculus1 0.588929 0.547605 0.484766 0.471347

Calculus2 1.000000 0.640908 0.638694 0.658728

Statistics 0.640908 1.000000 0.716918 0.644505

Probability 0.638694 0.716918 1.000000 0.704155

...

from4 -0.112189

y 0.319634

GPA.1 0.688738

ID.1 1.000000

1. Performance Distribution: What is the distribution of GPAs across all students?

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# Create a histogram of GPAs

plt.figure(figsize=(12, 6))

sns.histplot(data['GPA'], kde=True, bins=20, color='b')

plt.title('Distribution of GPAs')

plt.xlabel('GPA')

plt.ylabel('Frequency')

plt.show()

# Create a density plot of GPAs

plt.figure(figsize=(12, 6))

sns.kdeplot(data['GPA'], shade=True, color='y')

plt.title('Density Plot of GPAs')

plt.xlabel('GPA')

plt.ylabel('Density')

plt.show()

7. Top Performers: Who are the top 10 students in terms of GPA?

# Top Performers:  Who are the top 10 students in terms of GPA?

# Sort the data by GPA in descending order

top\_performers = data.sort\_values(by='GPA', ascending=False)

# Select the top 10 students

top\_10\_students = top\_performers.head(10)

# Print the top 10 students

print("Top 10 Students in terms of GPA:")

print(top\_10\_students)

Answer is: Top 10 Students in terms of GPA:

ID class gender race GPA Algebra Calculus1 Calculus2 \

104 1245 A male 1 93.71 93 97 99

68 1209 A female 1 93.06 82 90 93

75 1216 A female 1 91.90 95 86 96

96 1237 A male 2 91.74 94 100 96

93 1234 A female 1 91.61 93 82 95

97 1238 A male 1 91.14 98 90 98

41 1182 B female 1 90.66 88 78 95

84 1225 A female 1 90.62 94 81 90

98 1239 A male 1 90.31 84 82 99

79 1220 A female 3 89.95 87 87 92

Statistics Probability Measure Functional\_analysis from1 from2 from3 \

104 100 97 90 90 K B A

68 91 93 99 94 M B S

75 92 95 90 95 Z B A

96 97 95 94 93 C B S

93 95 94 100 93 Y B S

97 97 83 93 89 AA B A

41 85 97 97 99 B B A

84 93 91 86 94 S A A

98 97 89 96 85 P B A

79 92 90 92 84 P B A

...

41 0 1 B 90.66 1182

84 0 1 A 90.62 1225

98 0 2 A 90.31 1239

79 2 1 A 89.95 1220

8. Course Difficulty: Which courses have the lowest average scores, indicating higher difficulty?

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# List of course score columns

course\_columns = ['Algebra', 'Calculus1', 'Calculus2', 'Statistics', 'Probability', 'Measure', 'Functional\_analysis']

# Calculate the average scores for each course

average\_scores = data[course\_columns].mean()

# Print the average scores

print("Average scores for each course:")

print(average\_scores)

# Find the courses with the lowest average scores

lowest\_avg\_scores = average\_scores.nsmallest(3)  # Adjust the number if you want more or fewer courses

# Print the courses with the lowest average scores

print("\nCourses with the lowest average scores:")

print(lowest\_avg\_scores)

# Create a bar chart of the average scores

plt.figure(figsize=(12, 6))

sns.barplot(x=average\_scores.index, y=average\_scores.values, palette='viridis')

# Highlight the courses with the lowest average scores

for index, value in enumerate(average\_scores):

    if value in lowest\_avg\_scores.values:

        plt.text(index, value + 1, 'Challenging', color='red', ha='center')

plt.title('Average Scores of Each Course')

plt.xlabel('Course')

plt.ylabel('Average Score')

plt.xticks(rotation=45)

plt.show()

9. Impact of From Columns: How do the 'from1', 'from2', 'from3', and 'from4' columns impact the GPA?

import statsmodels.api as sm

from statsmodels.formula.api import ols

# Regression analysis for 'from4'

X = data[['from4']]

y = data['GPA']

X = sm.add\_constant(X)

reg\_model = sm.OLS(y, X).fit()

reg\_summary = reg\_model.summary()

# ANOVA for 'from1', 'from2', 'from3'

anova\_results = {}

for col in ['from1', 'from2', 'from3']:

    model = ols(f'GPA ~ C({col})', data=data).fit()

    anova\_table = sm.stats.anova\_lm(model, typ=2)

    anova\_results[col] = anova\_table

reg\_summary, anova\_results

10. Predictive Analysis: Can we predict the 'y' value based on GPA and course scores?

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

# Load the dataset

df = pd.read\_csv('project 4 data.csv')

# Prepare the feature matrix X (GPA and course scores) and target vector y ('y')

X = df[['GPA', 'Algebra', 'Calculus1', 'Calculus2', 'Statistics', 'Probability', 'Measure', 'Functional\_analysis']]

y = df['y']

# Split the dataset 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)

# Train the logistic regression model with increased maximum iterations

model = LogisticRegression(max\_iter=1000)

model.fit(X\_train, y\_train)

# Make predictions on the test set

y\_pred = model.predict(X\_test)

# Evaluate the model

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy}')

**Visualization Questions**

1. GPA Distribution: Create a histogram showing the distribution of GPAs.

import pandas as pd

import matplotlib.pyplot as plt

# Data including

data = pd.read\_csv("project 4 data.csv")

# Create DataFrame

df = pd.DataFrame(data)

# Plot histogram

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

plt.hist(df['GPA'], bins=20, color='blue', edgecolor='black')

plt.title('Distribution of GPAs')

plt.xlabel('GPA')

plt.ylabel('Frequency')

plt.grid(True)

plt.show()

Answer is: histogram

1. Average GPA by Gender: Create a bar chart comparing average GPA between male and female students.

# Plotting

avg\_gpa\_by\_gender.plot(kind='bar', color=['blue', 'green'])

plt.title('Average GPA by Gender')

plt.xlabel('Gender')

plt.ylabel('Average GPA')

plt.xticks(rotation=0)  # Rotate x-axis labels if needed

plt.show()

Answer is:

1. Average GPA by Race: Create a bar chart comparing average GPA across different races.

 # Plotting

avg\_gpa\_by\_race = data.groupby('race')['GPA'].mean()

avg\_gpa\_by\_race.plot(kind='bar', color='m')

plt.title('Average GPA by Race')

plt.xlabel('Race')

plt.ylabel('Average GPA')

plt.xticks(rotation=45)  # Rotate x-axis labels for better readability if needed

plt.show()

4. Course Score Trends: Create line charts for the average scores in Algebra, Calculus1, Calculus2, Statistics, Probability, Measure, and Functional\_analysis.

# Importing necessary libraries

import matplotlib.pyplot as plt

statistics = {}

courses = ["Algebra", "Calculus1", "Calculus2", "Statistics", "Probability", "Measure", "Functional\_analysis"]

for course in courses:

    statistics[course] = {

        'mean': data[course].mean(),

        'median': data[course].median(),

        'std': data[course].std()

    }

# Display the statistics

for course, stats in statistics.items():

    print(f"{course} - Mean: {stats['mean']}, Median: {stats['median']}, Standard Deviation: {stats['std']}")

# Calculate average scores for each course

average\_scores = data[courses].mean()

# Plotting

average\_scores.plot(kind='line', color='m',marker='\*',mfc='g',mec='b',linewidth='2')

plt.title('Average Scores in Different Courses')

plt.xlabel('Courses')

plt.ylabel('Average Score')

plt.xticks(rotation=45)  # Rotate x-axis labels for better readability if needed

plt.grid(True)

plt.show()

5.Correlation Heatmap: Create a heatmap to visualize the correlation matrix of GPA and course scores.

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# Select columns related to GPA and course scores

columns\_of\_interest = ['GPA', 'Algebra', 'Calculus1', 'Calculus2', 'Statistics', 'Probability', 'Measure', 'Functional\_analysis']

data\_selected = data[columns\_of\_interest]

# Calculate the correlation matrix

correlation\_matrix = data\_selected.corr()

# Create a heatmap

plt.figure(figsize=(10, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', linewidths=0.5)

plt.title('Correlation Heatmap of GPA and Course Scores')

plt.show()

6. Box Plot of GPA by Class: Create a box plot to show the distribution of GPAs for each class.

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# Create a box plot with custom colors

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

sns.boxplot(x='class', y='GPA', data=data, palette=['g', 'b'])

plt.title('Box Plot of GPA by Class')

plt.xlabel('Class')

plt.ylabel('GPA')

plt.show()

1. Scatter Plot of GPA vs. Course Scores: Create scatter plots to show the relationship between GPA and each course score.

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# List of course score columns

course\_columns = ['Algebra', 'Calculus1', 'Calculus2', 'Statistics', 'Probability', 'Measure', 'Functional\_analysis']

# List of colors to use

colors = ['r', 'g', 'b', 'k', 'm', 'c', 'y']

# Create scatter plots for each course score vs GPA

for course, color in zip(course\_columns, colors):

    plt.figure(figsize=(8, 6))

    sns.scatterplot(x=course, y='GPA', data=data, color=color)

    plt.title(f'Scatter Plot of GPA vs. {course}')

    plt.xlabel(course)

    plt.ylabel('GPA')

  plt.show()

1. Course Difficulty: Create a bar chart to show the average scores of each course, highlighting the most challenging courses.

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# List of course score columns

course\_columns = ['Algebra', 'Calculus1', 'Calculus2', 'Statistics', 'Probability', 'Measure', 'Functional\_analysis']

# Calculate the average scores for each course

average\_scores = data[course\_columns].mean()

# Create a bar chart

plt.figure(figsize=(12, 8))

sns.barplot(x=average\_scores.index, y=average\_scores.values, palette='viridis')

# Highlight the most challenging courses

# Assuming the most challenging courses are those with the lowest average scores

threshold = average\_scores.mean()

for index, value in enumerate(average\_scores):

    if value < threshold:

        plt.text(index, value + 1, 'Challenging', color='red', ha='center')

plt.title('Average Scores of Each Course')

plt.xlabel('Course')

plt.ylabel('Average Score')

plt.xticks(rotation=45)

plt.show()

1. Impact of 'From' Columns: Create a series of box plots to show the impact of 'from1', 'from2', 'from3', and 'from4' on GPA.

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv("project 4 data.csv")

# List of 'from' columns

from\_columns = ['from1', 'from2', 'from3', 'from4']

# List of colors to use for the box plots

colors = ['Blues', 'Greens', 'Reds', 'Purples']

# Create a box plot for each 'from' column with different colors

plt.figure(figsize=(16, 12))

for i, (from\_col, color) in enumerate(zip(from\_columns, colors), start=1):

    plt.subplot(2, 2, i)  # Create a 2x2 grid of plots

    sns.boxplot(x=from\_col, y='GPA', data=data, palette=color)

plt.title(f'Impact of {from\_col} on GPA')

plt.xlabel(from\_col)

plt.ylabel('GPA')

plt.tight\_layout()

plt.show()

1. correlation matrix of the data set.

import pandas as pd

# Load the data

data = pd.read\_csv("project 4 data.csv")

# Select only numeric columns

numeric\_data = data.select\_dtypes(include=['number'])

# Calculate the correlation matrix correlation\_matrix = numeric\_data.corr()

print(correlation\_matrix)