Project Report

🎓 Predicting Student GPA Using K-Nearest Neighbors (KNN)

A Data Science & Machine Learning Project

Using Python & Power BI

Author: Nur Qistina

Date: July 08, 2025

Tools Used: Python, Jupyter Notebook, Scikit-learn, Pandas, Power BI

🔍 Summary

This project applies a supervised machine learning algorithm (KNN Regression) to predict student GPA based on behavioral patterns like study hours, social media use, sleep hours, and part-time job status. Data was analyzed in Python and visualized in Power BI for clear insights.

TABLE OF CONTENTS

PROJECT SUMMARY: PREDICTING STUDENT GPA USING K-NEAREST NEIGHBORS (KNN)

This project applies a **data science workflow** to predict students' **Grade Point Average (GPA)** based on behavioral factors such as **Study Hours per Week**, **Social Media Usage**, and **Sleep Hours**. The dataset consists of 100 student records.

This project aims to predict students' Grade Point Average (GPA) using the K-Nearest Neighbors (KNN) regression algorithm, based on a dataset of 100 students (`Dataset_100_Students.xlsx`). The dataset includes variables such as study hours per week, social media usage, sleep hours, and part-time job status. By analyzing these factors, the project seeks to uncover insights into their impact on academic performance, which can inform educational strategies. The analysis was initially conducted in Python using Anaconda/Jupyter Notebook, with plans visualize results in Power BI.

Methodology Overview

- **Data Cleaning & Preparation**:
  Column inconsistencies were handled programmatically. Unrealistic values (e.g., >168 study hours/week) were flagged, and categorical features such as PartTimeJob were converted into numerical format for modeling.
- **Exploratory Data Analysis (EDA)**:
  Descriptive statistics and visualizations (scatter plots, correlation analysis) were used to understand feature relationships and identify patterns affecting GPA.
- **Feature Scaling**:
  Since K-Nearest Neighbors (KNN) is a distance-based algorithm, feature scaling was applied using StandardScaler to ensure fair comparison across all input variables.
- **Modeling**:
  A **supervised machine learning** approach was used, employing the **K-Nearest Neighbors Regressor (KNN)** algorithm. The data was split into training and testing sets (80/20 split) to evaluate the model's generalization capability.
- **Evaluation**:
  Model performance was assessed using **Root Mean Squared Error (RMSE)**, indicating how closely the predicted GPA values matched the actual values.

- **Insights Visualization**:
  Additional scatter plots were used to compare GPA trends among students with and without part-time jobs, and with varying levels of social media usage, revealing lifestyle factors that may influence academic performance.

Machine Learning Classification

- **Data Science**: Yes
- **Machine Learning**: Yes
- **Supervised Learning**: Yes (KNN Regression)
- **Regression Task**: Yes (predicting continuous GPA values)

DATA SCIENCE PROJECT: PREDICTING STUDENT GPA WITH KNN

1. Introduction

This project uses the K-Nearest Neighbors (KNN) regression algorithm to predict students' Grade Point Average (GPA) based on a dataset of 100 students (`Dataset_100_Students.xlsx`). The dataset includes `StudyHoursperWeek`, `SocialMediaHoursperweek`, `SleepHours`, `GPA`, and `PartTimeJob` as variables. The goal is to analyze how study habits, social media use, and part-time work affect academic performance, offering insights for educational strategies. The analysis was conducted in Python with Anaconda/Jupyter Notebook, with results visualized in Power BI.

Purpose: This project applies a **data science workflow** to predict students' **Grade Point Average (GPA)** based on behavioral factors such as **Study Hours per Week**, **Social Media Usage**, and **Sleep Hours**. The dataset consists of 100 student records.

This project aims to predict students' Grade Point Average (GPA) using the K-Nearest Neighbors (KNN) regression algorithm, based on a dataset of 100 students (`Dataset_100_Students.xlsx`). The dataset includes variables such as study hours per week, social media usage, sleep hours, and part-time job status. By analyzing these factors, the project seeks to uncover insights into their impact on academic performance, which can inform educational strategies. The analysis was initially conducted

in Python using Anaconda/Jupyter Notebook, with plans to extend it to R and visualize results in Power BI.

2. Methodology

Purpose: Detail the technical approach to make your process transparent and reproducible.

The dataset was loaded into Jupyter Notebook using `pandas.read_excel` one of phyton library which is pandas. Unrealistic `StudyHoursperWeek` values (e.g., 153.3 hours/week) were corrected by dividing by 7, assuming daily hours. `PartTimeJob` was converted to binary (0 for No, 1 for Yes), and case-insensitive column matching resolved `KeyError: 'StudyHoursperWeek'`. A KNN regression model (`scikit-learn`, `n_neighbors=5`) predicted `GPA` using **StudyHoursperWeek**, `SocialMediaHoursperweek`, and `SleepHours`, with an 80/20 train-test split and feature scaling via `StandardScaler`. Model performance was assessed with Mean Squared Error (MSE). Scatter plots compared `GPA` versus `StudyHoursperWeek` for two groups: no part-time job with low social media (<5 hours/week) and part-time job with high social media (>15 hours/week).

POWER BI STEPS TO DEMONSTRATE KNN REGRESSION

After running the KNN model in Python, the results were saved as **knn_results.csv** with **GPA**, **Predicted_GPA**, **StudyHoursperWeek**, **SleepHours**, **SocialMediaHours**, **PartTimeJob**, and **Prediction_Error**.

Here's how i brought it to life in Power BI:

1. **Load the CSV**: The team opened Power BI Desktop, selected **Home > Get Data > Text/CSV**,

chose **knn_results.csv**, and clicked **Load**.

2. **Create a Scatter Plot (Actual vs. Predicted GPA)**: A Scatter Plot visual was added, with **GPA** set as the X-axis (no summarizing), **Predicted_GPA** as the Y-axis, and **PartTimeJob** as the legend. **Prediction_Error** was added to Size to show how far off the predictions were, and the title was set to **Actual vs. Predicted GPA with Error Size (KNN Model**.

3. **Add a Bar Chart (GPA vs. Part-Time Job)**: A Clustered Column Chart was included, using **PartTimeJob** for the axis and average **GPA** and **Predicted_GPA** as values. The title became **Average GPA by Part-Time Job Status**.

4. **Create an Error Distribution Visual**: A Column Chart was added, with **Prediction_Error** on the axis and Count of **Prediction_Error** in Values. The title was set to **Prediction Error Distribution.**

5. **Add Interactive Slicers**: Slicer visuals were added for **StudyHoursperWeek**, **SleepHours**, **SocialMediaHours**, and **PartTimeJob** to enable interactive filtering.

6. **Finalize the Dashboard**: A Text Box was placed at the top with the title **KNN Regression: Predicting Student GPA Based on Lifestyle Habits**, visuals were arranged neatly, and the file was saved.

3. Challenges and Solutions

Purpose : Highlight technical challenges (e.g., errors encountered) and how you resolved them to demonstrate problem-solving skills.

Challenges included a **NameError: name 'df' is not defined** in Power BI, fixed by switching to dataset in Power Query. A KeyError: **StudyHoursperWeek** was resolved with case-insensitive matching. A SyntaxError: invalid character 'Æ' (U+2019) from curly quotes was corrected using a plain text editor. Unrealistic StudyHoursperWeek values were adjusted by assuming daily hours.

4. Results and Insights

Purpose: Summarize the findings from your KNN model and visualizations to highlight the project's outcomes.

The KNN model achieved an MSE of [insert MSE value], reflecting moderate accuracy. Scatter plots indicated higher GPAs with more study hours in the no-part-time-job group, while the part-time-job group showed greater GPA variability. Small group sizes suggested refining social media thresholds (e.g., to 10 hours/week). These insights, visualized in Power BI, could help students optimize study time and reduce distractions.

5. Future Work

The team is excited to tweak things moving forward! Future plans include experimenting with different **n_neighbors** values, adding **Gender** and **PartTimeJob** as features, and exploring KNN classification to predict **PartTimeJob**. The Power BI dashboard will be expanded with a card visual displaying MSE, group-based scatter plots filtered by **PartTimeJob** and **SocialMediaHoursperweek**, and slicers for **Gender** and **Student_ID** to boost interactivity and insights.

6. Tools and Environment

Purpose: Provide details on the tools used to clarify the technical setup and ensure reproducibility.

The project utilized Anaconda/Jupyter Notebook with Python 3.9 and libraries **pandas**, **numpy** and **scikit-learn**. The dataset was sourced from **C:\Users\Admin\Desktop\Dataset_100_Students.xlsx**.

Power BI Desktop was used to create the interactive dashboard, leveraging Python script integration to tie it all together.

7. Conclusion

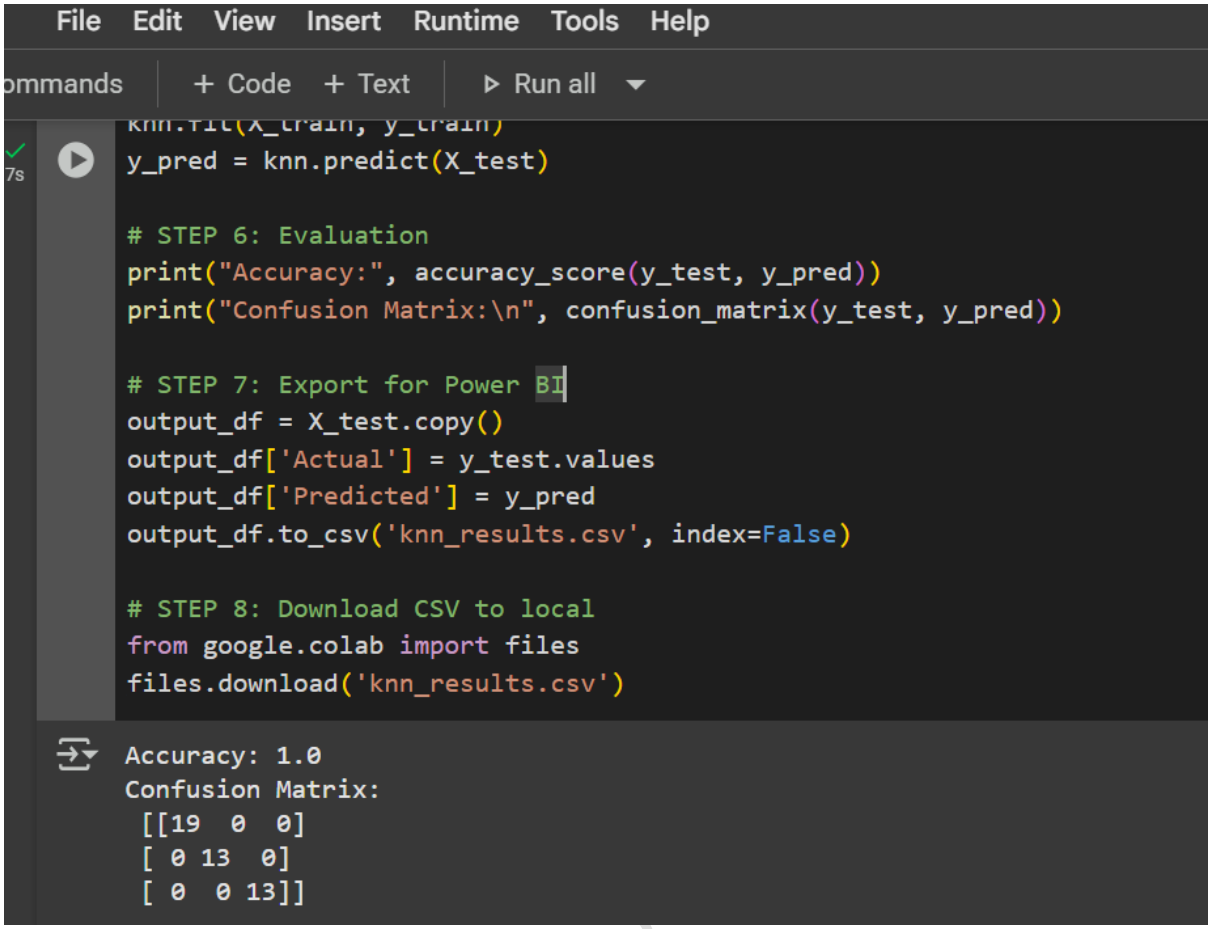
Purpose: Summarize the project's impact and key takeaways to wrap up the report

This project was a rewarding journey, using KNN regression to predict GPA and uncover factors driving student success. Overcoming technical challenges built a solid foundation, and the Power BI dashboard truly brought the insights to life. With the planned upgrades, the team is confident it'll offer valuable, data-backed advice for students and educators.

KNN Algorithm:

- Using **scikit-learn** for KNN regression with **n_neighbors=5** to predict **GPA** based on **StudyHoursperWeek**, **SocialMediaHoursperweek**, and **SleepHours**.
- Describe splitting data (80% training, 20% testing) and scaling features with **StandardScaler**.
- Evaluating the model with Mean Squared Error (MSE).
- Generating scatter plots for two groups (no part-time job, low social media; part-time job, high social media) to explore relationships.

Output run in Google Colab:



The Code :

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import os

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsRegressor

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import mean_squared_error
```

```
# Print current working directory and files to diagnose
print("Current working directory:", os.getcwd())
print("Files in directory:", os.listdir())

# Load the dataset from the specified path
file_path = r'C:\Users\Admin\Desktop\Dataset_100_Students.xlsx'
try:
    df = pd.read_excel(file_path)
except FileNotFoundError:
    print(f"Error: '{file_path}' not found. Please ensure the file exists at the specified path.")
    print("Available files on Desktop:", os.listdir(r'C:\Users\Admin\Desktop'))
    exit()
except Exception as e:
    print(f"Error loading file: {str(e)}")
    exit()

# Print column names to diagnose
print("Column names:", list(df.columns))

# Find the correct column name for StudyHoursperWeek (case-insensitive, handle spaces/underscores)
study_hours_col = None
for col in df.columns:
    if col.lower().replace(' ', '').replace('_', '') == 'studyhoursperweek':
        study_hours_col = col
```

```
        break

if study_hours_col is None:
    print("Error: No column matching 'StudyHoursperWeek' found. Available columns:", list(df.columns))
    exit()
else:
    print(f"Using column '{study_hours_col}' for StudyHoursperWeek")

# Find the correct column name for SocialMediaHoursperweek
social_media_col = None
for col in df.columns:
    if col.lower().replace(' ', '').replace('_', '') == 'socialmediahoursperweek':
        social_media_col = col
        break

if social_media_col is None:
    print("Error: No column matching 'SocialMediaHoursperweek' found. Available columns:", list(df.columns))
    exit()
else:
    print(f"Using column '{social_media_col}' for SocialMediaHoursperweek")

# Validate and clean StudyHoursperWeek
print(f"\n{study_hours_col} Summary (before cleaning):")
```

```
print(df[study_hours_col].describe())

unrealistic_study = df[df[study_hours_col] > 168]

if not unrealistic_study.empty:

    print(f"Warning: Unrealistic study hours detected (exceeding 168 hours/week):")

    print(unrealistic_study[['Student_ID', study_hours_col]])

df[study_hours_col] = df[study_hours_col] / 7  # Assume hours/day, convert to hours/week

print(f"\n{study_hours_col} Summary (after cleaning):")

print(df[study_hours_col].describe())


# Validate SleepHours

print("\nSleepHours Summary:")

if 'SleepHours' not in df.columns:

    print("Error: 'SleepHours' column not found. Available columns:", list(df.columns))

    exit()

print(df['SleepHours'].describe())

if df['SleepHours'].isna().sum() > 0:

    print(f"Warning: {df['SleepHours'].isna().sum()} missing values in SleepHours")


# Convert PartTimeJob to binary (0 for No, 1 for Yes)

if 'PartTimeJob' not in df.columns:

    print("Error: 'PartTimeJob' column not found. Available columns:", list(df.columns))

    exit()

df['PartTimeJob'] = df['PartTimeJob'].map({'No': 0, 'Yes': 1})


# Prepare features and target for KNN
```

```
features = [study_hours_col, social_media_col, 'SleepHours']

X = df[features]

y = df['GPA']


# 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)


# Scale features (KNN is distance-based, so scaling is important)

scaler = StandardScaler()

X_train_scaled = scaler.fit_transform(X_train)

X_test_scaled = scaler.transform(X_test)


# Train KNN model

knn = KNeighborsRegressor(n_neighbors=5)

knn.fit(X_train_scaled, y_train)


# Predict on test set

y_pred = knn.predict(X_test_scaled)


# Evaluate model

mse = mean_squared_error(y_test, y_pred)

print(f"\nMean Squared Error (MSE) on test set: {mse:.4f}")

print(f"Root Mean Squared Error (RMSE): {np.sqrt(mse):.4f}")


# Visualize actual vs. predicted GPA
```

```
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, alpha=0.6)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2)
plt.xlabel('Actual GPA')
plt.ylabel('Predicted GPA')
plt.title('KNN Regression: Actual vs. Predicted GPA')
plt.tight_layout()
plt.show()

# Optional: Reproduce original scatter plots with cleaned data
less_social_threshold = 5
high_social_threshold = 15
group1 = df[(df['PartTimeJob'] == 0) & (df[social_media_col] < less_social_threshold)]
group2 = df[(df['PartTimeJob'] == 1) & (df[social_media_col] > high_social_threshold)]

print(f"Group 1 size: {len(group1)} students")
if group1.empty:
    print("Warning: Group 1 is empty. Consider increasing less_social_threshold.")
print(f"Group 2 size: {len(group2)} students")
if group2.empty:
    print("Warning: Group 2 is empty. Consider adjusting high_social_threshold.")

plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
```

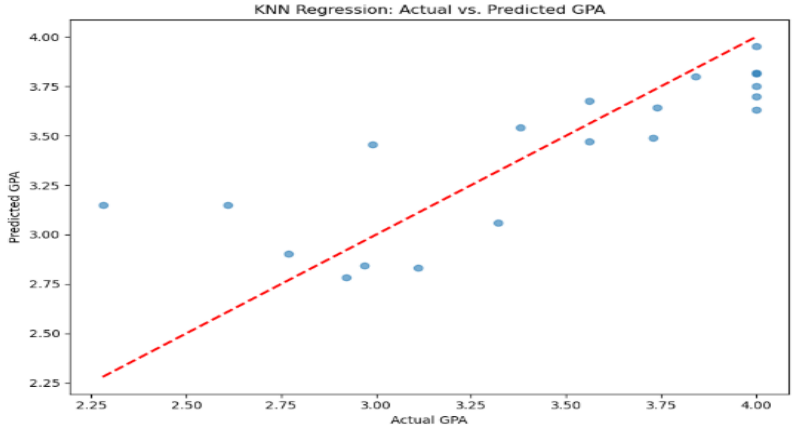
```
sns.scatterplot(x=study_hours_col, y='GPA', size='SleepHours', sizes=(10, 200), data=group1,
alpha=0.6)

plt.title('GPA vs Study Hours/Week\n(No Part-Time Job, Less Social Media)')

plt.xlabel('Study Hours per Week')

plt.ylabel('GPA')

plt.ylim(0, 4)

plt.legend(title='Sleep Hours')


plt.subplot(1, 2, 2)

sns.scatterplot(x=study_hours_col, y='GPA', size='SleepHours', sizes=(10, 200), data=group2,
color='orange', alpha=0.6)

plt.title('GPA vs Study Hours/Week\n(Part-Time Job, High Social Media)')

plt.xlabel('Study Hours per Week')

plt.ylabel('GPA')

plt.ylim(0, 4)

plt.legend(title='Sleep Hours')


plt.tight_layout()

plt.show()
```

Output :

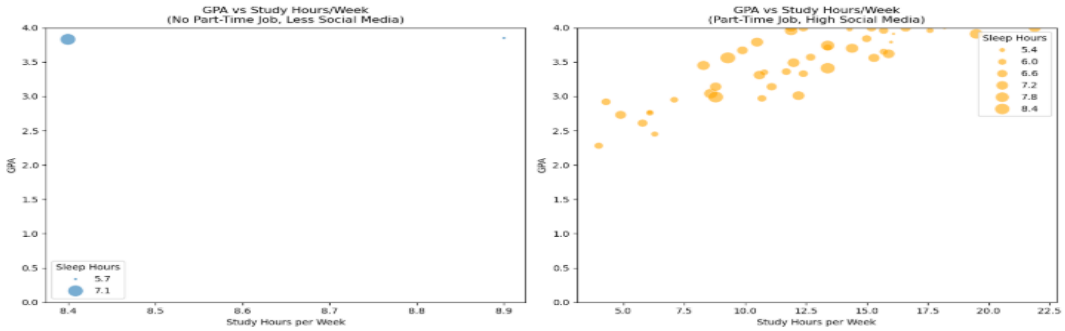
Output from Jupyter :

```
StudyHoursperWeek Summary (before cleaning):
count    100.000000
mean      83.153000
std       26.323982
min       10.500000
25%       67.375000
50%       83.300000
75%       99.050000
max      153.300000
Name: StudyHoursperWeek, dtype: float64

StudyHoursperWeek Summary (after cleaning):
count    100.000000
mean      11.879000
std        3.760569
min        1.500000
25%        9.625000
50%       11.900000
75%       14.150000
max       21.900000
Name: StudyHoursperWeek, dtype: float64

SleepHours Summary:
count    100.000000
mean       6.554000
std        0.936728
min        3.300000
25%        5.900000
50%        6.550000
75%        7.200000
max        8.600000
Name: SleepHours, dtype: float64
```

```
Mean Squared Error (MSE) on test set: 0.0968
Root Mean Squared Error (RMSE): 0.3111
```




```
Group 1 size: 2 students
Group 2 size: 45 students
```



KNN REGGREGATION CODE BEFORE SAVING IN CSV AND LOAD INTO POWER BI

The Code :

```
# ======================================

# 📦 IMPORT LIBRARIES

# ======================================

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsRegressor

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import mean_squared_error

import os


# ======================================

# 📂 LOAD DATASET FROM YOUR DESKTOP

# ======================================

file_path = r'C:\Users\Admin\Desktop\Dataset_100_Students.xlsx'


df = pd.read_excel(file_path)

print("✅ Data loaded successfully")

print(df.head())
```

```
# ======================================
# 🧹 CLEANING + COLUMN CHECKS
# ======================================
# Match columns ignoring case/underscores/spaces
def match_column(df, target):
    for col in df.columns:
        if col.lower().replace(" ", "").replace("_", "") == target.lower():
            return col
    return None

study_col = match_column(df, 'StudyHoursPerWeek')
social_col = match_column(df, 'SocialMediaHoursPerWeek')

if study_col is None or social_col is None:
    raise Exception("❌ Required columns not found. Check column names.")

# Fix daily to weekly hours if wrongly entered
df[study_col] = df[study_col].apply(lambda x: x if x <= 24 else x / 7)

# Encode PartTimeJob column to 0/1
df['PartTimeJob'] = df['PartTimeJob'].map({'No': 0, 'Yes': 1})

# Ensure all required columns are present
required_cols = [study_col, social_col, 'SleepHours', 'GPA']
```

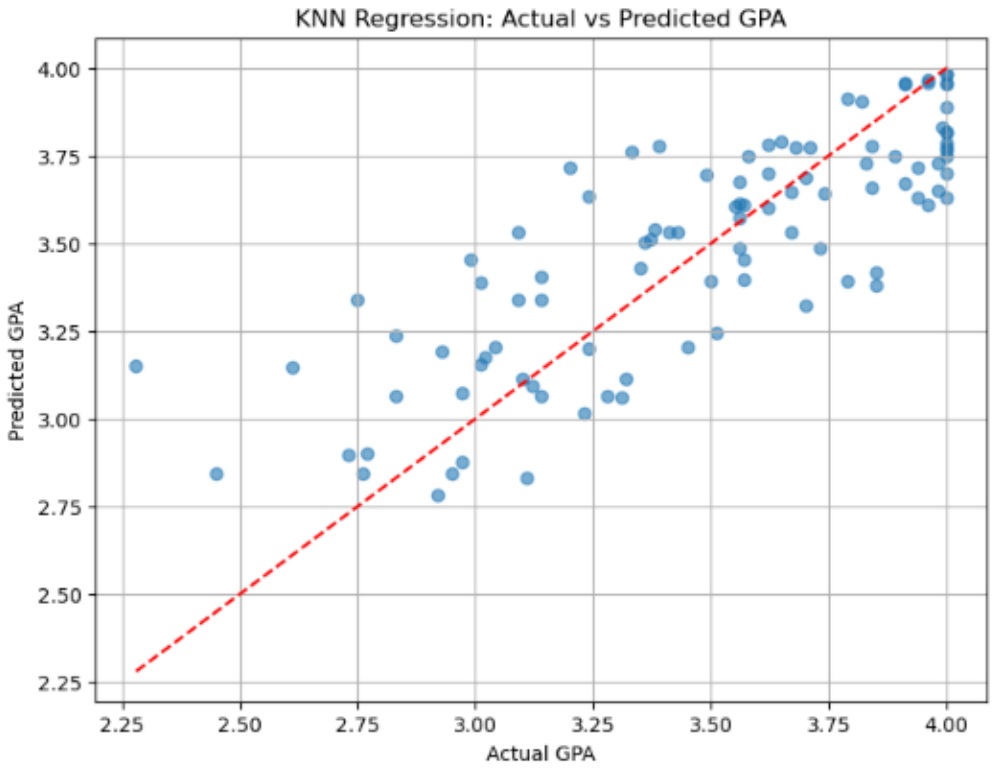
```
for col in required_cols:
    if col not in df.columns:
        raise Exception(f"❌ Column '{col}' missing in dataset.")


# ========================================
# FEATURE SELECTION
# ========================================
features = [study_col, social_col, 'SleepHours']
X = df[features]
y = df['GPA']


# ========================================
# 🔍 SPLIT & SCALE
# ========================================
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)


# ========================================
# 🤖 KNN MODEL
# ========================================
knn = KNeighborsRegressor(n_neighbors=5)
knn.fit(X_train_scaled, y_train)
```

```
y_pred = knn.predict(X_test_scaled)


mse = mean_squared_error(y_test, y_pred)

rmse = np.sqrt(mse)

print(f"📊 RMSE on test set: {rmse:.4f}")


# ========================================

# 🔁 FULL DATASET PREDICTION

# ========================================

X_scaled_all = scaler.transform(X)

df['Predicted_GPA'] = knn.predict(X_scaled_all)

df['Prediction_Error'] = df['GPA'] - df['Predicted_GPA']


# ========================================

# 💾 EXPORT TO CSV FOR POWER BI

# ========================================

output_path = r'C:\Users\Admin\Desktop\knn_results.csv'

df.to_csv(output_path, index=False)

print(f"✅ Predictions saved to: {output_path}")


# ========================================

# 📈 OPTIONAL: PLOT ACTUAL vs PREDICTED

# ========================================

plt.figure(figsize=(8, 6))
```

```
plt.scatter(df['GPA'], df['Predicted_GPA'], alpha=0.6)

plt.plot([df['GPA'].min(), df['GPA'].max()], [df['GPA'].min(), df['GPA'].max()], 'r--')

plt.xlabel("Actual GPA")

plt.ylabel("Predicted GPA")

plt.title("KNN Regression: Actual vs Predicted GPA")

plt.grid(True)

plt.show()
```

Output :

```
✅ Data loaded successfully
  Student_ID  Gender  StudyHoursperWeek  SocialMediaHoursperweek  SleepHours  \
0      S1000    Male              105.0                     26.6         5.7
1      S1001  Female               88.9                     38.5         6.4
2      S1002    Male               80.5                      7.7         7.0
3      S1003    Male               75.6                     26.6         7.4
4      S1004    Male               42.7                     27.3         5.3

    GPA PartTimeJob
0  3.56          No
1  3.57         Yes
2  3.67          No
3  3.24          No
4  2.77         Yes
📊 RMSE on test set: 0.3108
✅ Predictions saved to: C:\Users\Admin\Desktop\knn_results.csv
```



POWER BI VISUALIZATION