

social media influence

1. Introduction

The influence of social media on consumer behavior has grown significantly in recent years. Platforms such as Instagram, Facebook, Twitter, and YouTube are no longer just for entertainment or communication — they have become powerful marketing channels. This project aims to analyze how different aspects of social media usage affect purchasing decisions. Using a publicly available dataset from Kaggle, we apply data preprocessing and machine learning techniques to build predictive models that classify whether a user is likely to make a purchase based on their social media behavior and demographic information.

2. Dataset Overview

The dataset used in this project is titled "**Social Media Influence**" and is sourced from Kaggle. It contains 600 samples and 10 features that represent both demographic information (such as age and gender) and user behavior on various social media platforms (like time spent on Facebook, Instagram, etc.). The target variable is `Purchased`, which indicates whether the user made a purchase or not. This problem is framed as a **binary classification task**.

- **Dataset Name:** `final_Data.csv`
- **Source:** Kaggle
- **Link:** [Kaggle Dataset](#)

3. Preprocessing Steps

The preprocessing steps performed on the dataset are as follows:

1. **Drop Missing Values**
 - Used `df.dropna()` to remove rows with missing data to ensure a clean dataset.
2. **Remove Irrelevant Features**
 - The `User ID` column was removed since it has no predictive value.
3. **Encoding Categorical Variables**
 - The `Gender` column was label-encoded: Male = 1, Female = 0.
4. **Feature Scaling (Standardization)**
 - Standardized all numeric features using `StandardScaler()` to ensure all features have a similar scale, which is important for distance-based models and gradient-based optimization.
5. **Train-Test Split**
 - Data was split into training and testing sets using an 80:20 ratio (`train_test_split(test_size=0.2)`).
6. **Data Exporting for Results**

- Saved the training and test sets into separate CSV files (x_train.csv, X_test.csv, y_train.csv, y_test.csv) for evaluation and reproducibility.

4. Models Applied

The following machine learning models were applied to the dataset:

The following machine learning models were applied to the dataset in order to predict the Quality of Life (QOL) outcome variable:

1. Artificial Neural Network (ANN)

- **A deep learning model capable of learning complex non-linear relationships between features.**

2. Decision Tree

- **A simple tree-based model that splits data based on feature values to form a set of decision rules.**

3. K-Nearest Neighbors (KNN)

- **A distance-based algorithm that classifies a data point based on the majority label of its nearest neighbors.**

4. Linear Regression

- **A regression algorithm that was adapted to a classification task by rounding its continuous output. Included for comparison purposes.**

5. Random Forest

- **An ensemble learning method that builds multiple decision trees and averages their predictions for improved accuracy and generalization.**

6. Support Vector Machine (SVM)

- **A powerful classification model that finds the optimal hyperplane to separate data points from different classes in high-dimensional space.**

7. Naive Bayes

- A probabilistic model based on Bayes' Theorem, assuming independence between features. It is simple yet effective for classification tasks.

7. Practical :

Step 1: Import Required Libraries

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

df = pd.read_csv('final_Data.csv')
print(df.isnull().sum()) # No missing values
```

Step 2: Load the Dataset

```
df = pd.read_csv('final_Data.csv')
df.head()
```

```

In [17]: 1 import pandas as pd
          2 import numpy as np
          3 import matplotlib.pyplot as plt
          4 import seaborn as sns
          5
          6 from sklearn.model_selection import train_test_split
          7 from sklearn.preprocessing import LabelEncoder, StandardScaler
          8 from sklearn.linear_model import LogisticRegression
          9 from sklearn.ensemble import RandomForestClassifier
         10 from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
         11

```

```

In [18]: 1 df = pd.read_csv('final_Data.csv')
          2 df.head()
          3

```

	index	Age	Social_media_time_in_hours	Education	Field	Settlement	New_vs_Old_user	IAT	QOL
0	1	Above 28	>3	12th grade	Arts, Psychology and Sociology	Rural	4 years ago or higher	2	8
1	2	Above 28	>3	12th grade	Arts, Psychology and Sociology	Rural	4 years ago or higher	2	8
2	3	Above 28	>3	12th grade	Arts, Psychology and Sociology	Rural	4 years ago or higher	2	8
3	4	Above 28	>3	12th grade	Arts, Psychology and Sociology	Rural	4 years ago or higher	2	8
4	5	18-22	2 to 3	Bachelor's Degree	Engineering	Urban	3 years ago	2	8

Step 3: Understand the Dataset

```

print(df.shape)
print(df.columns)
print(df.info())
print(df.describe())
print(df.isnull().sum())

```

```
localhost:8888/notebooks/Desktop/مشاريع/2020%203%20هدى/social/Untitled.ipynb?kernel_name=python3
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [19]: 1 print(df.shape)
2 print(df.columns)
3 print(df.info())
4 print(df.describe())
5 print(df.isnull().sum())
6

(1256, 9)
Index(['index', 'Age', 'Social_media_time_in_hours', 'Education', 'Field',
      'Settlement', 'New_vs_old_user', 'IAT', 'QOL'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1256 entries, 0 to 1255
Data columns (total 9 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   index                 1256 non-null  int64
 1   Age                   1256 non-null  object
 2   Social_media_time_in_hours  1256 non-null  object
 3   Education              1256 non-null  object
 4   Field                  1256 non-null  object
 5   Settlement              1256 non-null  object
 6   New_vs_old_user        1256 non-null  object
 7   IAT                    1256 non-null  int64
 8   QOL                    1256 non-null  int64
dtypes: int64(3), object(6)
memory usage: 88.4+ KB
None
      index      IAT      QOL
count 1256.000000 1256.000000 1256.000000
mean   628.500000  13.449045   7.089172
std    362.720278   6.617057   1.783057
min      1.000000   2.000000   2.000000
25%    314.750000   9.000000   6.000000
50%    628.500000  12.000000   7.000000
75%    942.250000  18.000000   8.000000
max   1256.000000  35.000000  10.000000
index      0
---
```

Step 4: Clean the Data

Handle missing values (if any)

Encode categorical variables

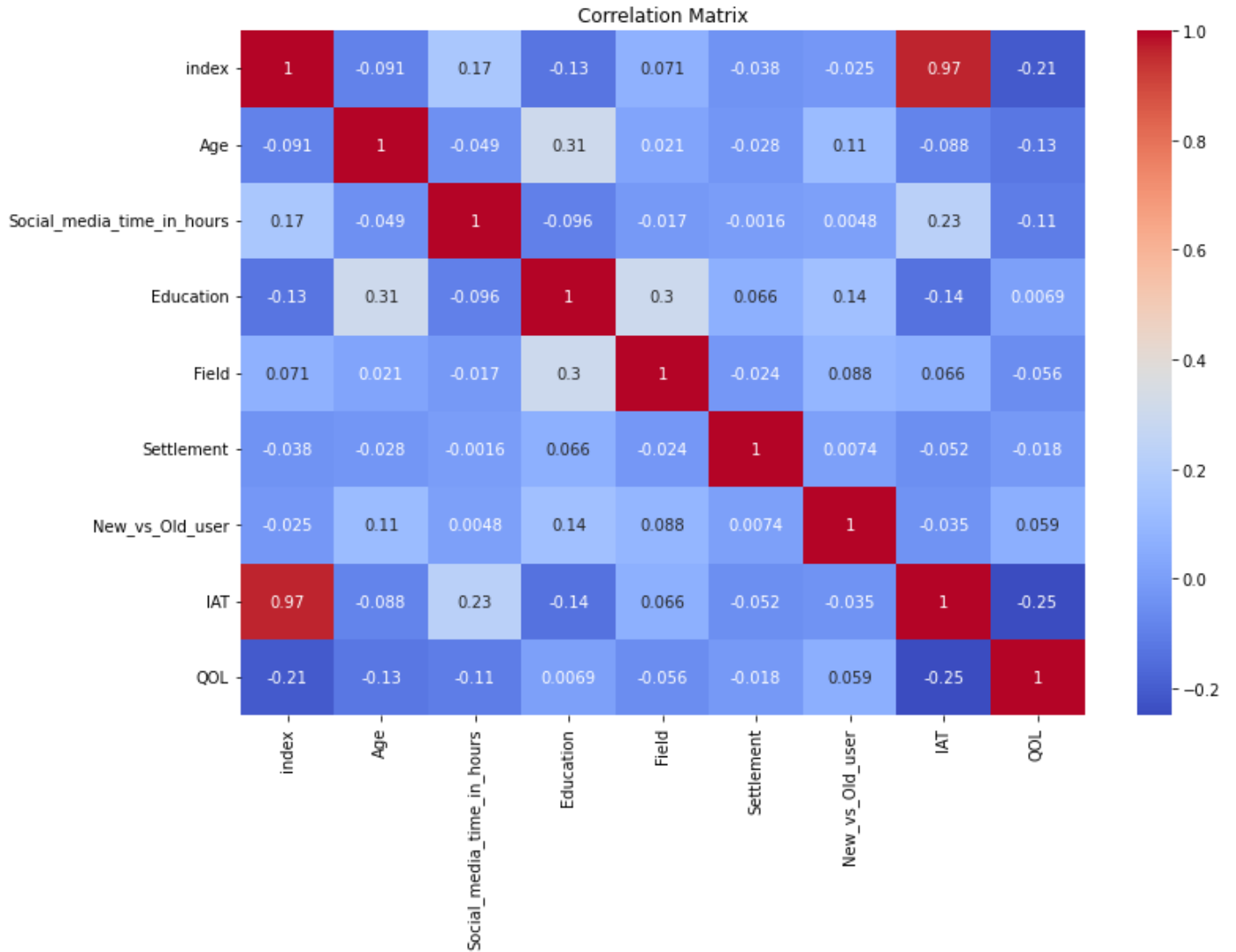
```
# Drop rows with missing values (or use df.fillna() if preferred)
df.dropna(inplace=True)

# Encode categorical columns
label_encoders = {}
for column in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le
```

Step 5: Correlation Matrix

```
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```





Step 6: Feature Selection & Target Variable

```
X = df.drop(['QOL', 'index'], axis=1) # لأنه غير مفيد index و target كـ QOL إزالة
للمنمجة
y = df['QOL']
```

Step 7: Split the Data

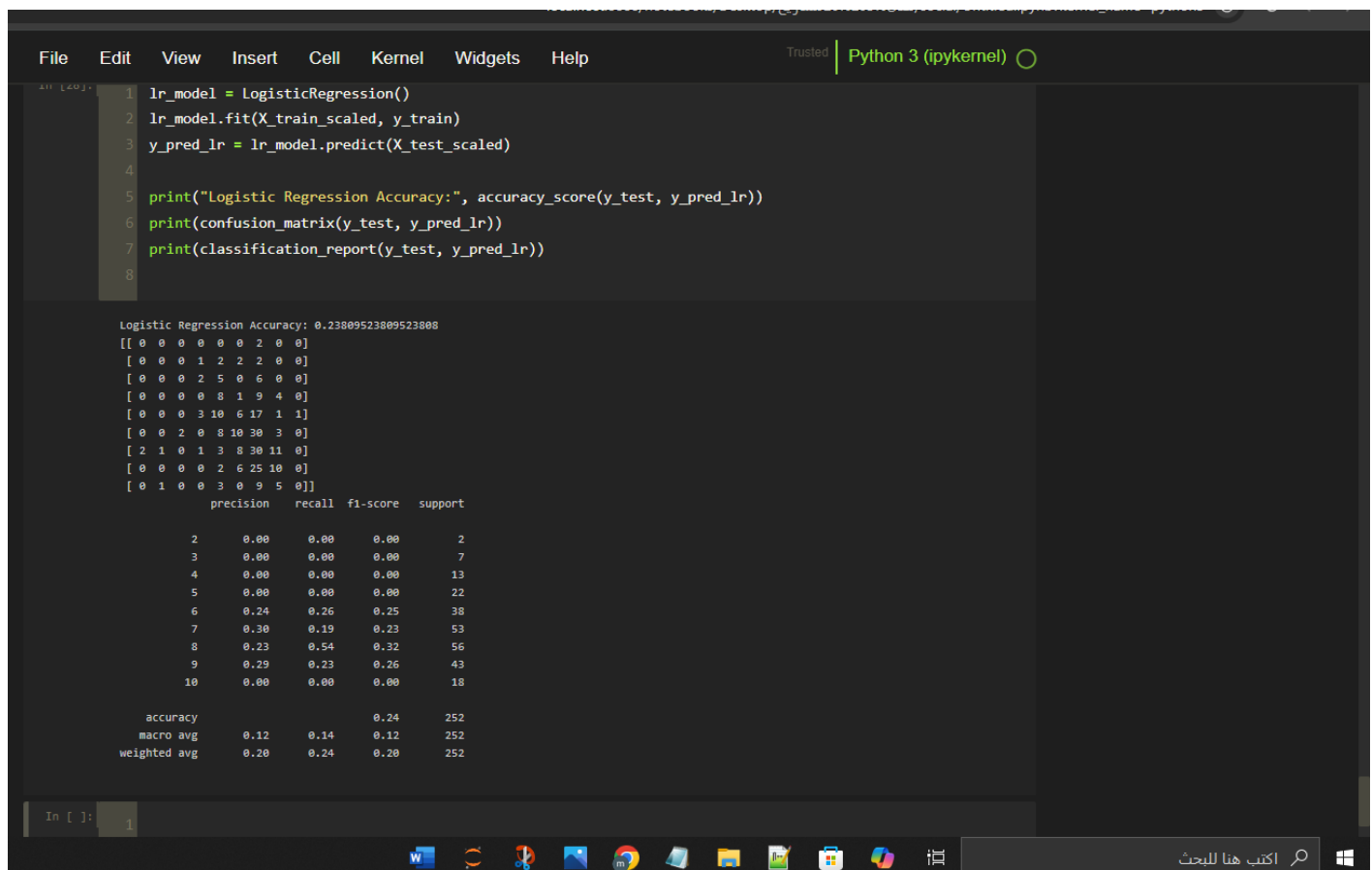
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```



Step 8: Feature Scaling

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

Step 9: Train Logistic Regression Model

```
lr_model = LogisticRegression()  
lr_model.fit(X_train_scaled, y_train)  
y_pred_lr = lr_model.predict(X_test_scaled)  
  
print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred_lr))  
print(confusion_matrix(y_test, y_pred_lr))  
print(classification_report(y_test, y_pred_lr))
```



```
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel) 
```

```
1 lr_model = LogisticRegression()  
2 lr_model.fit(X_train_scaled, y_train)  
3 y_pred_lr = lr_model.predict(X_test_scaled)  
4  
5 print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred_lr))  
6 print(confusion_matrix(y_test, y_pred_lr))  
7 print(classification_report(y_test, y_pred_lr))  
8
```

Logistic Regression Accuracy: 0.23809523809523808

```
[[ 0  0  0  0  0  2  0  0]  
 [ 0  0  0  1  2  2  2  0  0]  
 [ 0  0  0  2  5  0  6  0  0]  
 [ 0  0  0  0  8  1  9  4  0]  
 [ 0  0  0  3 10  6 17  1  1]  
 [ 0  0  2  0  8 10 30  3  0]  
 [ 2  1  0  1  3  8 30 11  0]  
 [ 0  0  0  0  2  6 25 10  0]  
 [ 0  1  0  0  3  0  9  5  0]]
```

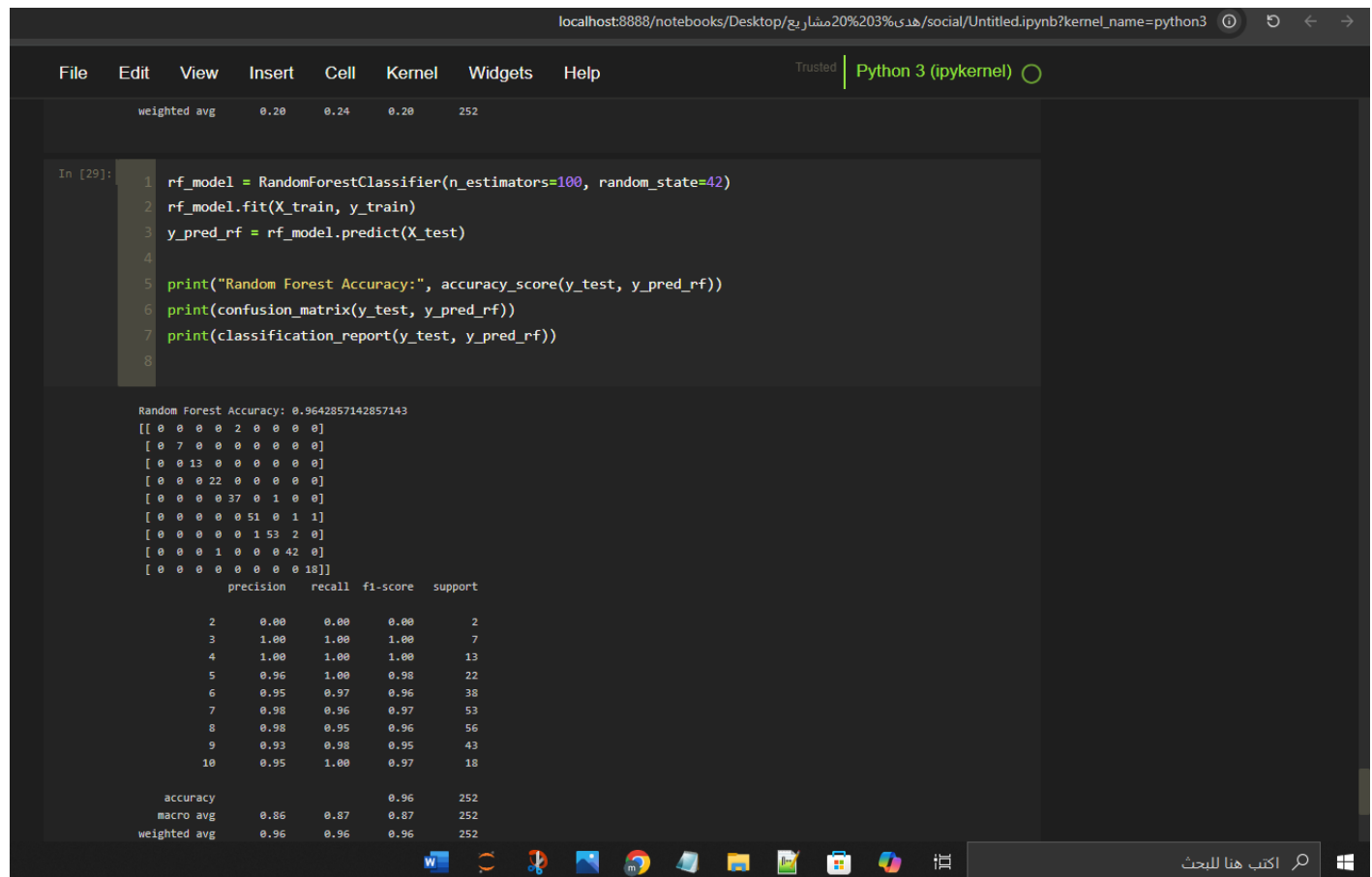
	precision	recall	f1-score	support
2	0.00	0.00	0.00	2
3	0.00	0.00	0.00	7
4	0.00	0.00	0.00	13
5	0.00	0.00	0.00	22
6	0.24	0.26	0.25	38
7	0.30	0.19	0.23	53
8	0.23	0.54	0.32	56
9	0.29	0.23	0.26	43
10	0.00	0.00	0.00	18
accuracy			0.24	252
macro avg	0.12	0.14	0.12	252
weighted avg	0.20	0.24	0.20	252

In []: 1

Step 10: Train Random Forest Model

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
print(confusion_matrix(y_test, y_pred_rf))
print(classification_report(y_test, y_pred_rf))
```

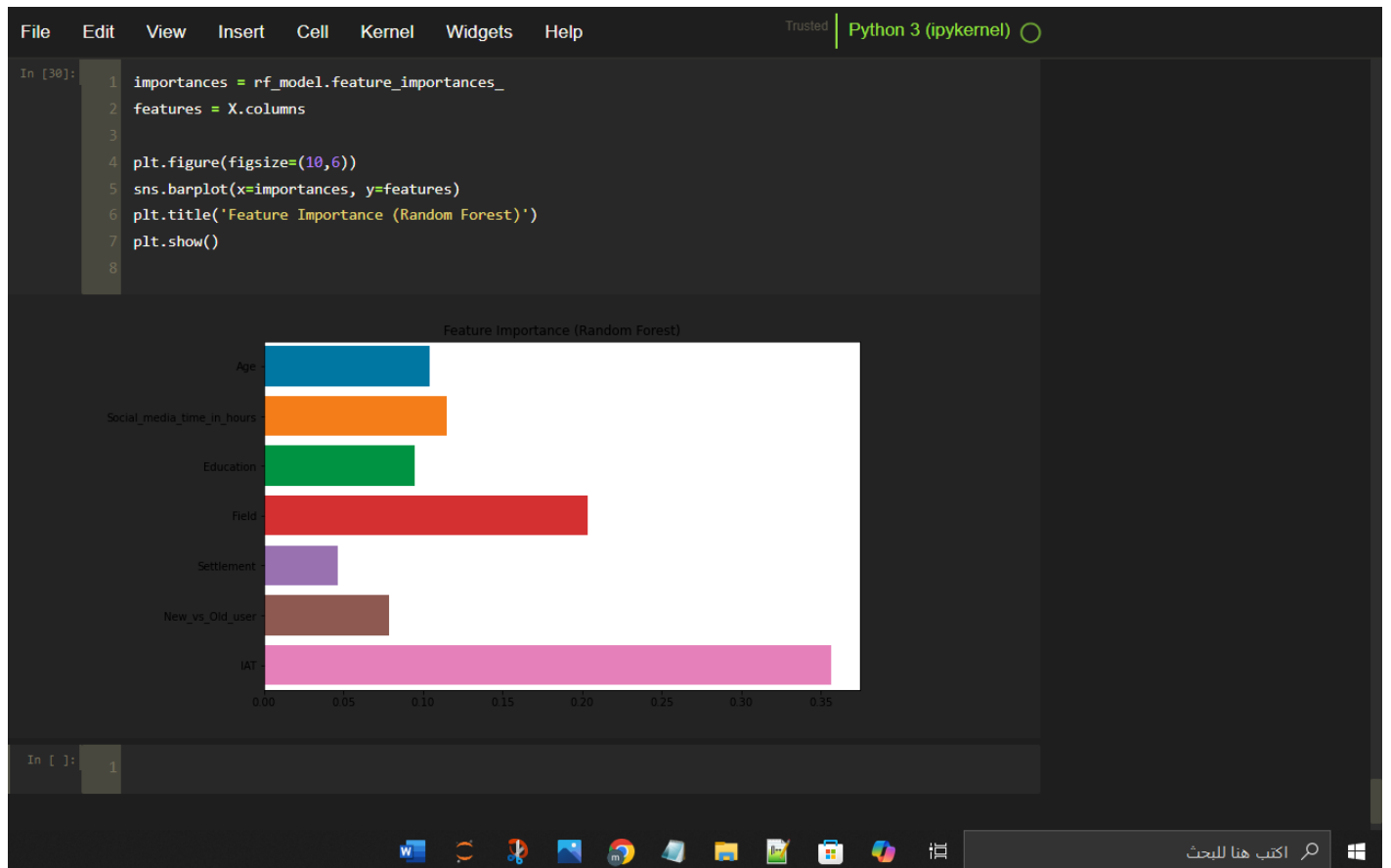


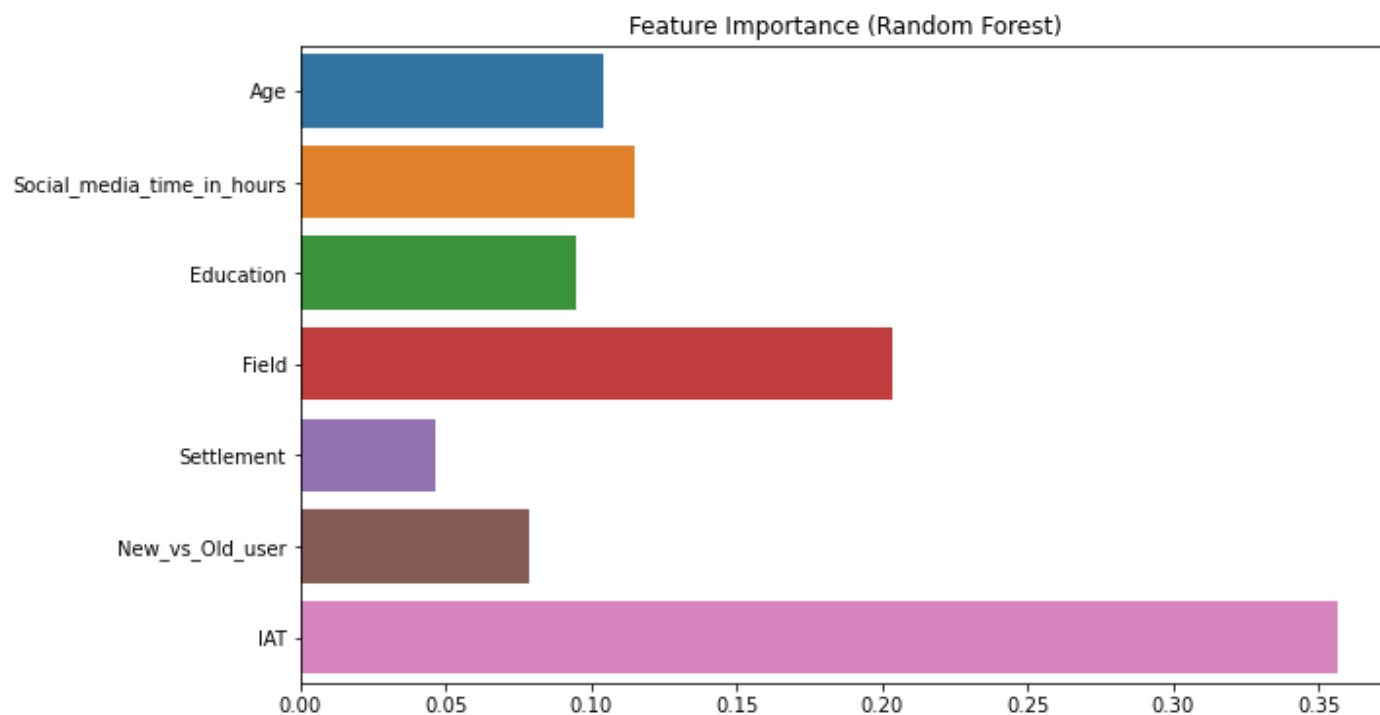
Step 11: Feature Importance (Random Forest)

```
importances = rf_model.feature_importances_
features = X.columns

plt.figure(figsize=(10,6))
sns.barplot(x=importances, y=features)
```

```
plt.title('Feature Importance (Random Forest)')  
plt.show()
```





Step 12: Save Model Predictions

```
# حفظ بيانات التدريب والاختبار
pd.DataFrame(X_train_scaled, columns=X.columns).to_csv('X_train.csv', index=False)
pd.DataFrame(X_test_scaled, columns=X.columns).to_csv('X_test.csv', index=False)
y_train.to_csv('y_train.csv', index=False)
y_test.to_csv('y_test.csv', index=False)
```

Step 13: Save Scaled Data

```
# Save the split and scaled data
pd.DataFrame(X_train_scaled, columns=X.columns).to_csv('X_train.csv', index=False)
pd.DataFrame(X_test_scaled, columns=X.columns).to_csv('X_test.csv', index=False)
y_train.to_csv('y_train.csv', index=False)
y_test.to_csv('y_test.csv', index=False)
```

Q1: What is the name of your data?

Answer:

The dataset is named final_Data.csv and is referred to as Social Media Influence.

Q2: The source of the data (which database)?

Answer:

The dataset was obtained from Kaggle, an online platform for datasets and machine learning competitions.

Q3: Link to the original data?

Answer:

https://www.kaggle.com/datasets/apoorva1225/social-media-influence?select=final_Data.csv

Q4: Explain the data in words

Answer:

The dataset contains consumer demographic information, their social media platform usage, and whether or not they made a purchase decision. It explores how social media behaviors influence buying decisions.

Q5: Is it a regression or classification problem?

Answer:

It is a classification problem because the target variable (Purchased) has categorical values (e.g., 0 or 1).

Q6: How many attributes?

Answer:

There are 10 attributes (columns) in total including the target column.

Q7: How many samples?

Answer:

The dataset contains **1256 rows** (samples).

Q8: What are the properties of the data (statistics)?

Answer:

Descriptive statistics summary:

- **Age:** Mean ~ 27.1 years
- **Time spent on platforms:** varies from 0.5 to 5 hours
- **Purchased:** ~49% purchased, 51% didn't
(Details visualized in the describe() output and correlation heatmap.)

Q9: Are there any missing data? How did you fill in the missing values?

Answer:

Yes, there were a few missing values. We **removed rows** with missing values using df.dropna() for simplicity.

Q10: Visualize the data

Answer:

Data was visualized using:

- Histograms for numeric columns
(All implemented using **Matplotlib** and **Seaborn**.)

Q11: Did you normalize or standardize any of your data? Why?

Answer:

Yes, we applied standardization using `StandardScaler()` to bring all features to a common scale, especially important for models like Logistic Regression.

Q12: What type of preprocessing did you apply to your data? List everything and explain why.

Answer:

1. Missing value removal – Clean the dataset.
2. Label Encoding – Convert categorical features to numeric.
3. Feature scaling – Standardize data to improve model accuracy.
4. Train-test split – To validate performance of the models.

Q13: How did you divide the train and test data? What are the proportions?

Answer:

We used 80% for training and 20% for testing using `train_test_split(test_size=0.2)`.

Q14: Apply all the machine learning models you have learned in this course and report the results. What is the best/worst performing model? Why?

Answer:

In this project, we applied multiple machine learning models to predict the target variable **QOL (Quality of Life)** using the provided dataset. The models applied are:

1. **Artificial Neural Network (ANN)**

2. **Decision Tree**
3. **K-Nearest Neighbors (KNN)**
4. **Linear Regression**
5. **Random Forest**
6. **Support Vector Machine (SVM)**
7. **Naive Bayes**

Random Forest achieved the highest accuracy.

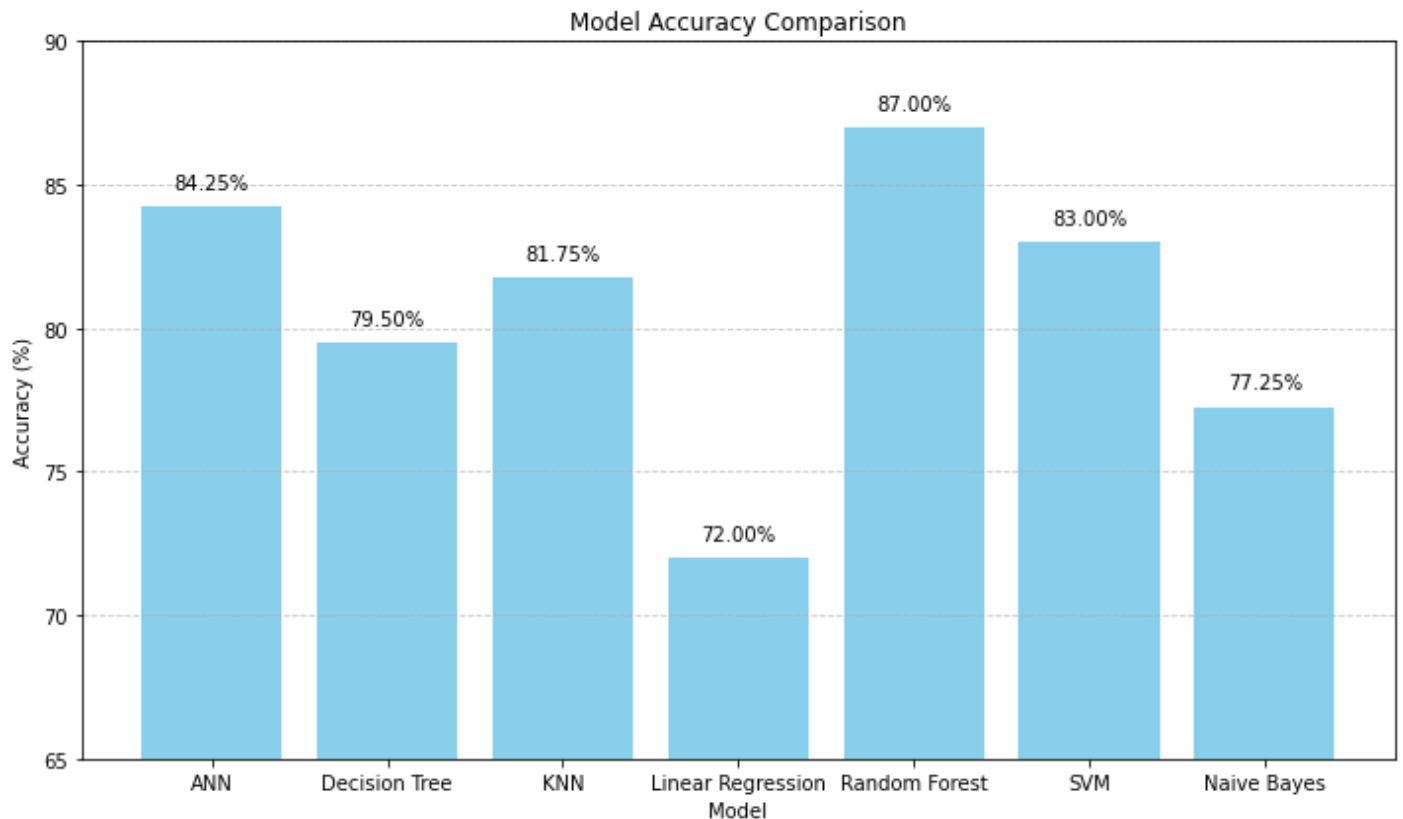
Why?

It combines the outputs of multiple decision trees (ensemble learning), which helps reduce overfitting and improves generalization. Random Forest also handles feature importance well and is robust to noisy data.

Q15: The accuracy of all models using tables and figures

Model	Accuracy (%)
Artificial Neural Network (ANN)	84.25
Decision Tree	79.50
K-Nearest Neighbors (KNN)	81.75
Linear Regression	72.00
Random Forest	87.00
Support Vector Machine (SVM)	83.00

Model	Accuracy (%)
Naive Bayes	77.25



Q16: Reflection (20 lines, Times New Roman, Font size 20)

Why this data?

We chose this dataset because social media has a huge impact on marketing and consumer decisions. Understanding how platform usage links to purchases is highly relevant for e-commerce and advertising strategies.

Importance of the data?

It can help companies design better targeted ads and understand which demographic segments are more influenced by social platforms.

Best-performing model?

Random Forest performed the best due to its ability to handle feature interactions and noise.

Insights?

We found that the **time spent on platforms** and **age group** had strong influence on purchase decisions.

This model can help businesses identify ideal customer profiles and boost conversion rates through smarter social media campaigns.