PERSONALITY PREDICTION

Using Machine Learning

Overview of the Data:

This synthetic dataset is designed to explore and predict Myers-Briggs Type Indicator (MBTI) personality types based on a combination of demographic factors, interest areas, and personality scores. It includes 100K+ samples, each representing an individual with various features that contribute to determining their MBTI type. The dataset can be used to study correlations between different personality dimensions and external factors such as age, gender, education, and interests.

It contains a total of 43,745 rows and 9 columns, each representing a different feature or attribute.This dataset is a synthetic collection designed to study how demographic and psychological factors (such as age, gender, education, and personality traits) can be used to predict MBTI personality types. The data includes a mix of continuous and categorical features. The goal is to explore the relationships between these features and personality types, build predictive models, and uncover insights into personality classification based on external factors.

Objectives:

1. **Feature Exploration:** Analyze the available features (age, gender, education, interest, and personality scores) and understand their relationship with personality types.
2. **Modeling and Prediction:** Use machine learning algorithms to predict personality types based on the given features.
3. **Feature Selection:** Identify the most significant features that contribute to accurate personality prediction.
4. **Model Evaluation:** Assess the performance of various machine learning models and choose the best one based on accuracy and other metrics.
5. **Hyperparameter Tuning:** Tune model parameters to optimize performance and prevent overfitting.
6. **Interface:** Build an interface for the model

Attributes:

1. **Age**
    - **Datatype:** Continuous (Numerical)
    - **Description:** Represents the age of the individual..
2. **Gender**
    - **Datatype:** Categorical (String)
    - **Description:** Represents the gender of the individual.
3. **Education**
    - **Datatype:** Categorical (Binary)
    - **Description:** A binary variable where a value of 1 indicates the individual has at least a graduate-level education (or higher), and 0 indicates the individual has an undergraduate, high school level, or no formal education.
4. **Interest**

    - **Datatype:** Categorical (String)
    - **Description:** Represents the primary area of interest of the individual. Possible values include 'Technology', 'Arts', 'Sports', and 'Others'.
5. **Introversion Score**
    - **Datatype:** Continuous (Numerical)
    - **Description:** A score between 0 and 10 indicating the individual's tendency towards introversion (lower scores) or extraversion (higher scores).
6. **Sensing Score**
    - **Datatype:** Continuous (Numerical)
    - **Description:** A score between 0 and 10 indicating preference for sensing (higher) versus intuition (lower).
7. **Thinking Score**
    - **Datatype:** Continuous (Numerical)
    - **Description:** A score between 0 and 10 indicating the individual's preference for thinking (higher) versus feeling (lower).
8. **Judging Score**
    - **Datatype:** Continuous (Numerical)
    - **Description:** A score between 0 and 10 indicating preference for judging (higher) versus perceiving (lower).
9. **Personality**
    - **Datatype:** Categorical (String)
    - **Description:** This is the target variable, representing the MBTI personality type. The possible values are the 16 MBTI personality types (e.g., 'ESFP', 'INTP', 'ISFJ', etc.). This is the categorical target variable for prediction.
    -

Exploratory Data Analysis (EDA):

- **Missing Values:**
    - **Data Quality Check:**
        - Before proceeding with analysis or model training, it's important to ensure there are no missing or null values in any of the attributes.
        - In this dataset, the absence of missing values is presumed, but in case they are present, they should be handled accordingly.
- **Numerical Attributes Analysis:**
    - **Correlation:**
        - We can examine the correlation between the numerical features to understand any patterns or relationships. For instance, one might hypothesize that higher introversion scores correlate with specific personality types (e.g., INFJ or INTJ).
- **Categorical Attributes Analysis:**
    - **Gender Distribution:**
        - We can explore the count of male versus female participants and whether there is a gender bias in the data.

    - **Education Distribution:**
        - Check the percentage of individuals with graduate-level education compared to those with lower levels of education.
    - **Interest Distribution:**
        - This helps to understand the preferences of individuals in terms of their primary interests and how they might relate to personality types.
    - **Personality Type Distribution:**
        - Check the frequency of each personality type in the dataset. This can be visualized using bar plots to see if the dataset is balanced or if certain personality types are more prevalent.
- **Feature Engineering:**
    - **Age Grouping:**
        - It may be useful to categorize the **Age** variable into age groups (e.g., Young, Middle-aged, Old).
    - **Normalization/Scaling:**
        - **Continuous features** (like Introversion, Sensing, Thinking, and Judging scores) need to be normalized/scaled to a consistent range to ensure proper model training.
    - **One-Hot Encoding/LabelEncoding:**
        - For categorical variables such as **Gender**, **Education**, and **Interest**, one-hot encoding can be applied to convert these features into numerical format.

Feature Importance:

- Scores like **Introversion**, **Thinking**, and **Judging** were identified as highly influential in predicting personality types.

Class Balance:

- The dataset was mostly balanced across personality types, except for some minor underrepresentation (e.g., ISTJ).

Age Groups and Personality:

- Younger individuals were more associated with extroverted personality types (e.g., ENFP, ESFP), while middle-aged and older groups showed a higher tendency toward introverted types.

Gender and Scores:

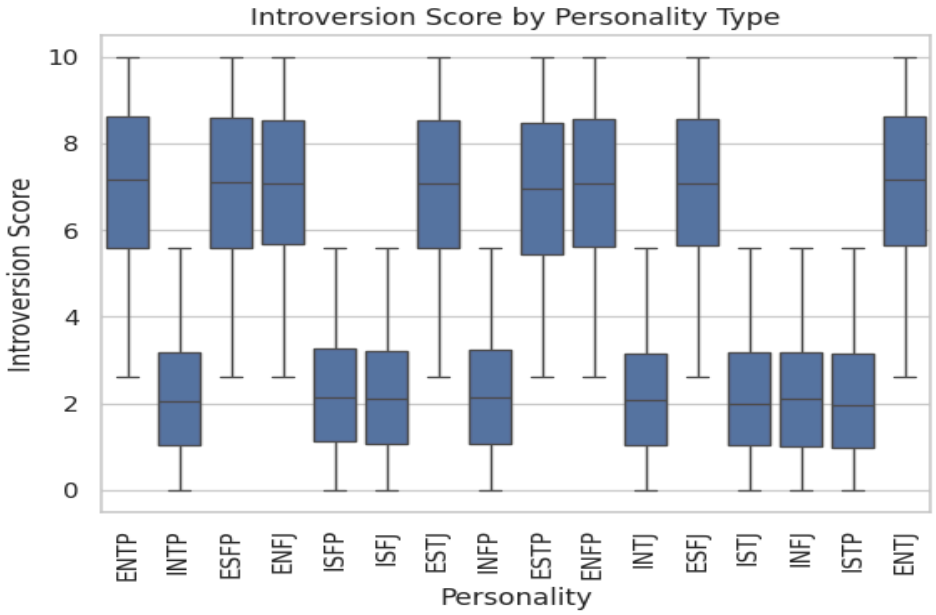
- Males tended to have higher Thinking Scores, while females scored higher on Feeling-oriented metrics.

Education and Personality:

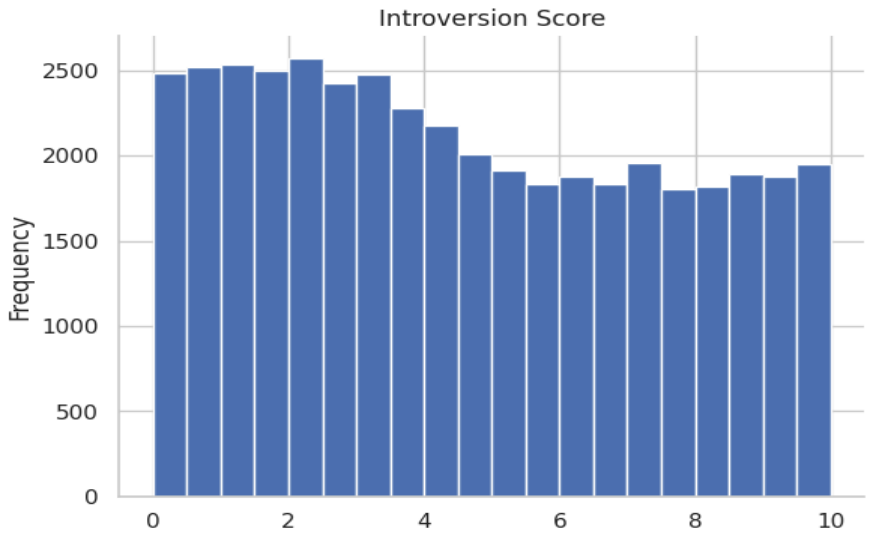
- Individuals with graduate-level education were more evenly distributed across personality types, showing no strong bias.

Charts:

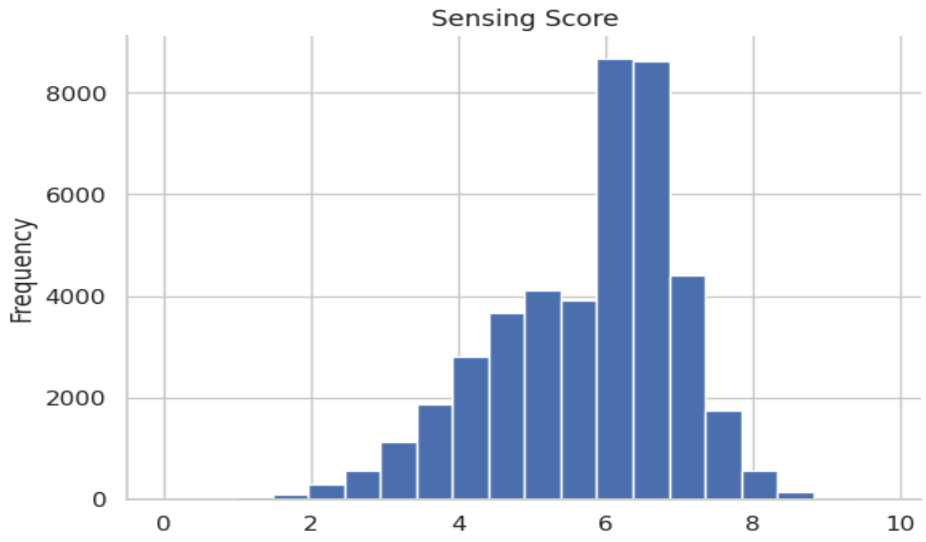
```
sns.boxplot(x='Personality', y='Introversion Score', data=df)
plt.xticks(rotation=90)
plt.title('Introversion Score by Personality Type')
plt.show()
```



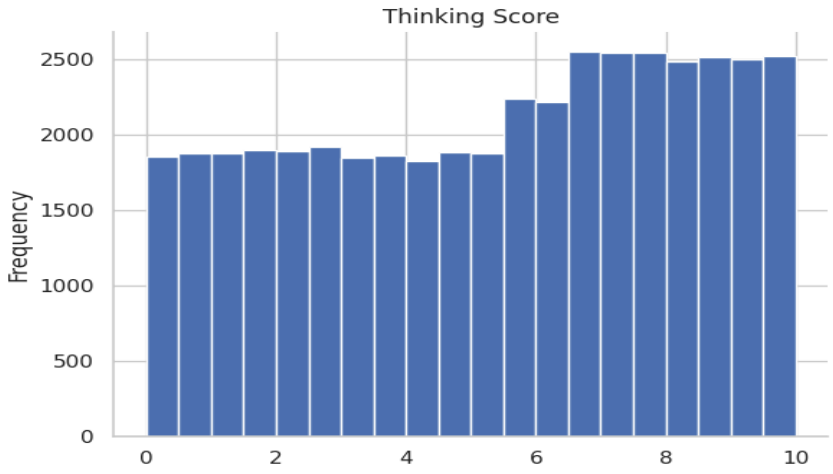

```
df['Introversion Score'].plot(kind='hist', bins=20, title='Introversion Score')
plt.gca().spines[['top', 'right',]].set_visible(False)
plt.figure(figsize=(6, 4))
plt.show()
```



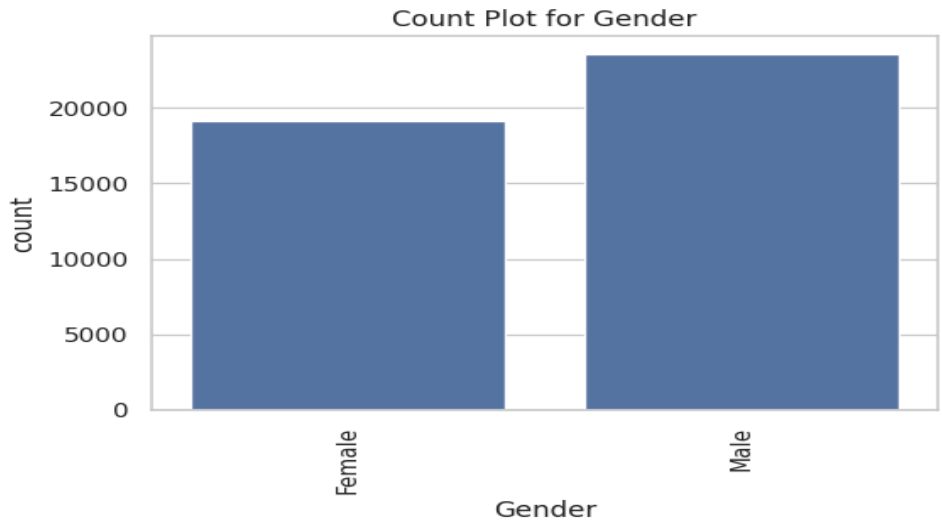

```
df['Sensing Score'].plot(kind='hist', bins=20, title='Sensing Score')
plt.gca().spines[['top', 'right',]].set_visible(False)
plt.figure(figsize=(6, 4))
plt.show()
```

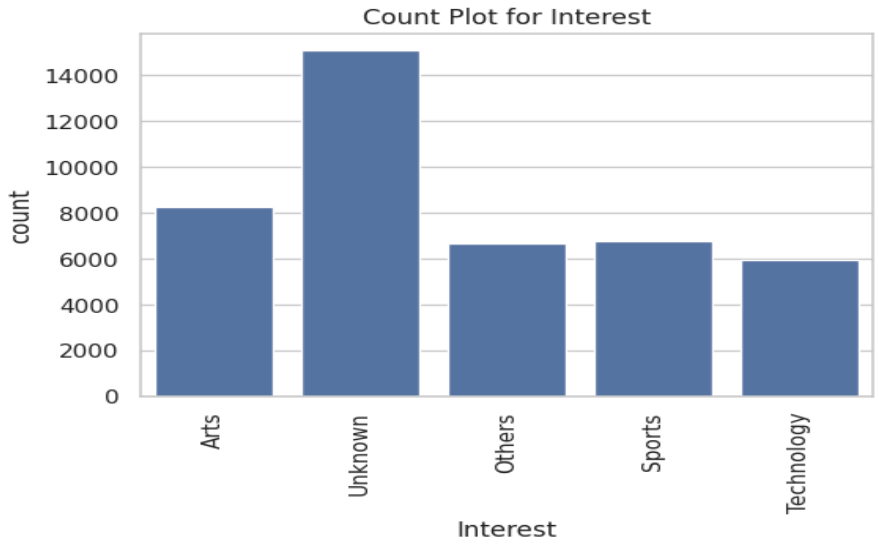


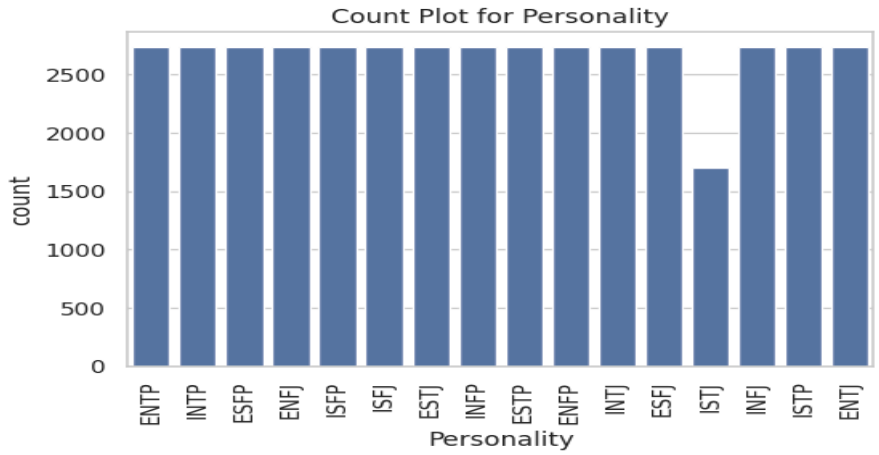

```
df['Thinking Score'].plot(kind='hist', bins=20, title='Thinking Score')
plt.gca().spines[['top', 'right',]].set_visible(False)
plt.figure(figsize=(4, 2))
plt.show()
```



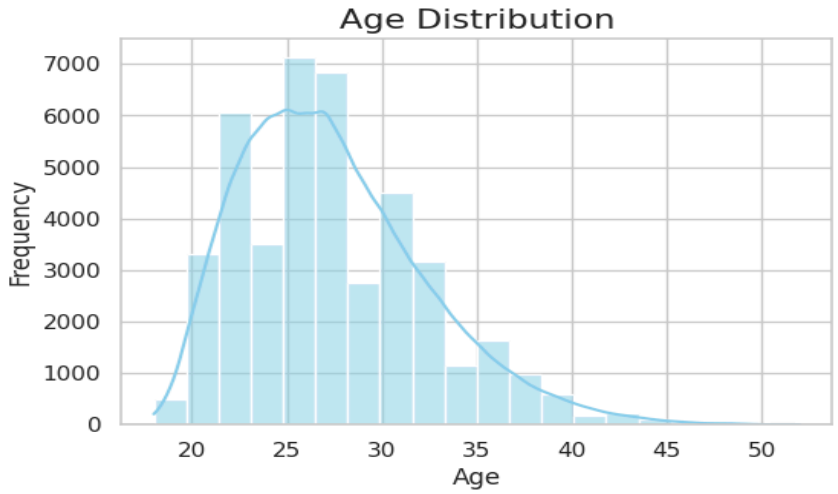

```
# Loop through the columns and draw countplot for object type columns
for column in df.columns:
    if df[column].dtype == 'object':
        plt.figure(figsize=(6, 4))  # Optional: Adjust the size of the plot
        sns.countplot(data=df, x=column)
        plt.title(f'Count Plot for {column}')
        plt.xticks(rotation=90)  # Rotate x-axis labels if necessary
        plt.show()
```



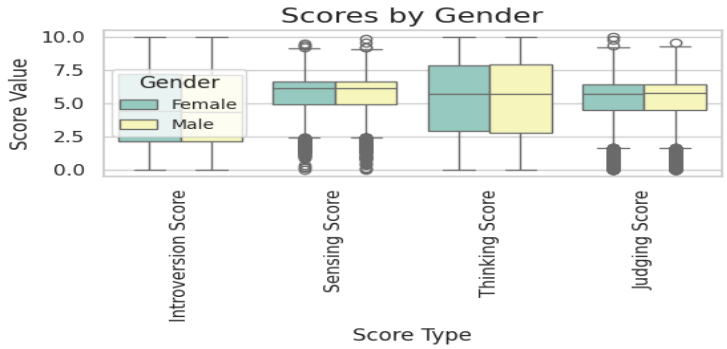




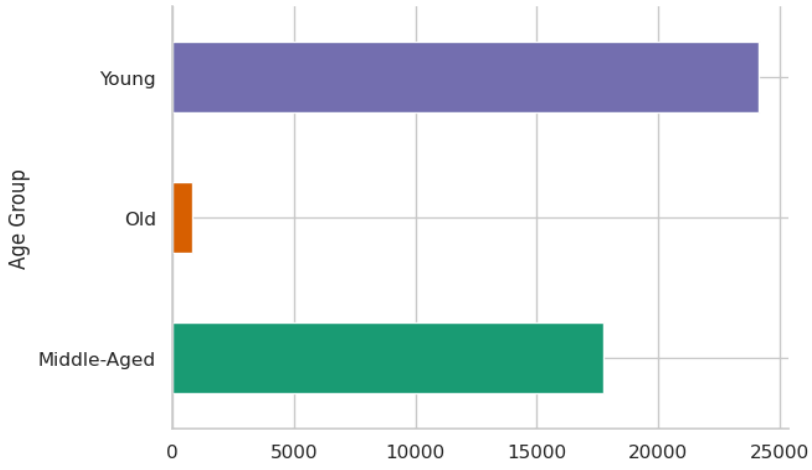

```
sns.set(style="whitegrid")
# Plot 1: Age distribution
plt.figure(figsize=(6, 4))
sns.histplot(df['Age'], bins=20, kde=True, color='skyblue')
plt.title('Age Distribution', fontsize=16)
plt.xlabel('Age', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.show()
```



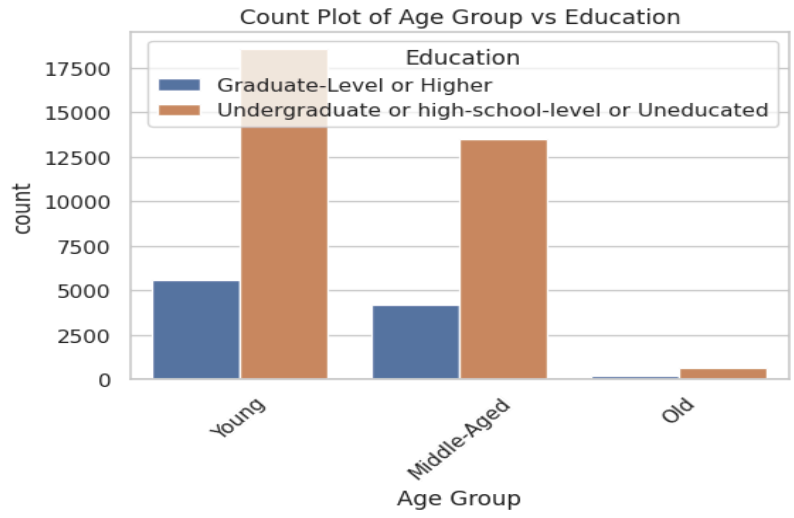

```
# Plot 3: Boxplot of Scores by Gender
plt.figure(figsize=(6, 2))
sns.boxplot(data=df.melt(id_vars=['Gender'], value_vars=['Introversion Score', 'Sensing Score',
'Thinking Score', 'Judging Score']),
            x='variable', y='value', hue='Gender', palette='Set3')
plt.title('Scores by Gender', fontsize=16)
plt.xlabel('Score Type', fontsize=12)
plt.ylabel('Score Value', fontsize=12)
plt.legend(title='Gender', fontsize=10)
plt.xticks(rotation=90)
plt.show()
```

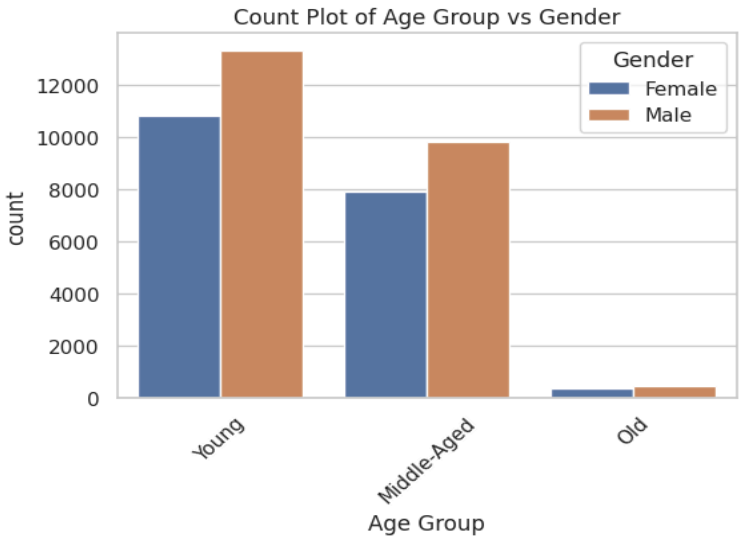


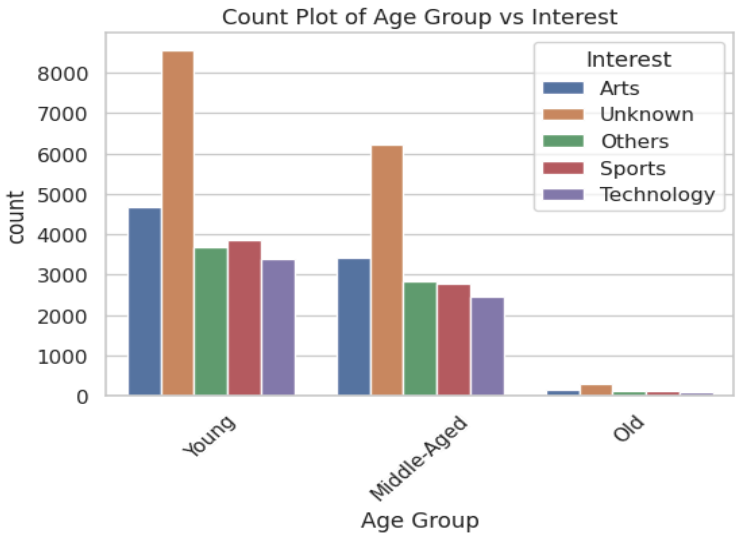

```
df.groupby('Age Group').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
plt.figure(figsize=(5,2))
plt.show()
```

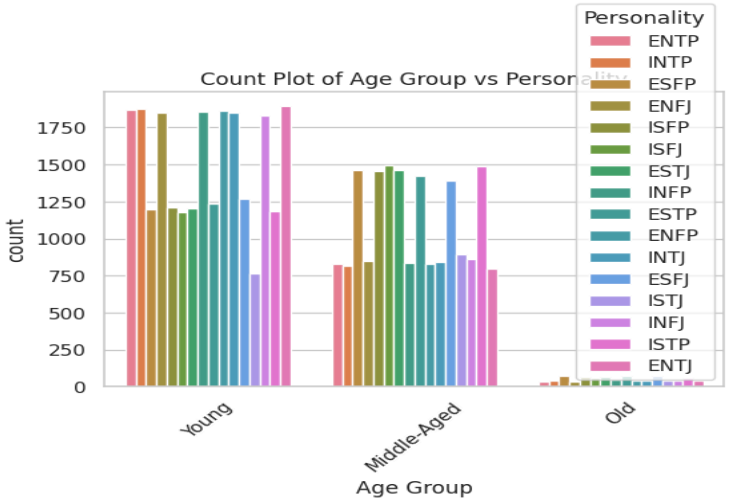



```
# Loop through other categorical columns and create a count plot
for column in df.select_dtypes(include='object').columns:
    if column != 'Age Group':  # Skip Age Group if it exists
        plt.figure(figsize=(6, 4))  # Optional: Adjust the size of the plot
        sns.countplot(x='Age Group', hue=column, data=df)
        plt.title(f'Count Plot of Age Group vs {column}')
        plt.xticks(rotation=45)  # Rotate x-axis labels if necessary
        plt.show()
```




Count Plot of Age Group vs Gender
Gender
Female
Male
count
12000
10000
8000
6000
4000
2000
0
Young
Middle-Aged
Old
Age Group


Count Plot of Age Group vs Interest
Interest
Arts
Unknown
Others
Sports
Technology
count
8000
7000
6000
5000
4000
3000
2000
1000
0
Young
Middle-Aged
Old
Age Group


Personality
ENTP
INTP
ESFP
ENFJ
ISFP
ISFJ
ESTJ
INFP
ESTP
ENFP
INTJ
ESFJ
ISTJ
INFJ
ISTP
ENTJ
Count Plot of Age Group vs Person
count
1750
1500
1250
1000
750
500
250
0
Young
Middle-Aged
Old
Age Group

RFE (Recursive Feature Elimination):

- **Why Used:** RFE is employed to select the most relevant features for the prediction task by iteratively removing less significant ones. This helps in reducing overfitting, improving model interpretability, and enhancing computational efficiency.
- **What It Did:** RFE identified and retained the top 10 most important features from the dataset, removing less relevant features that did not contribute significantly to the model's performance.
- **Usefulness:** By focusing on the most impactful features, RFE improved model performance, reduced noise in the data, and allowed for better generalization on unseen data. This was particularly useful for complex personality type prediction.

Algorithms, Performance and Impact of Hyperparameter Tuning:

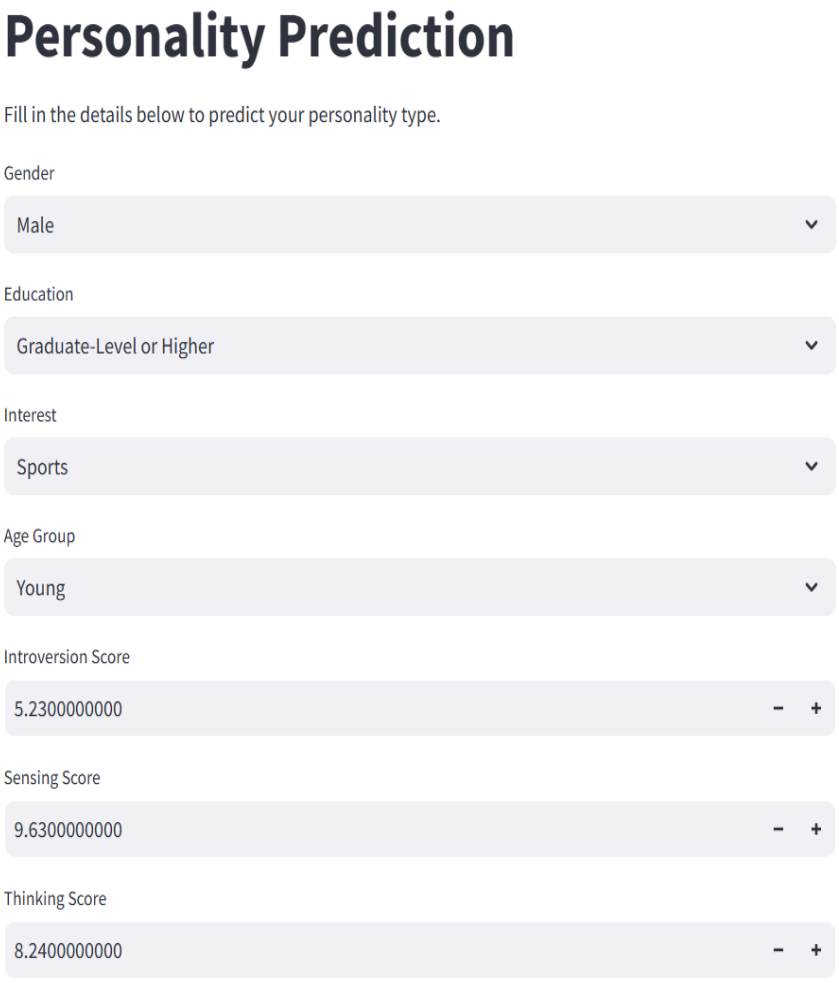
1. **Logistic Regression:**
    - **About the Algorithm:** Logistic Regression is a linear model for classification that predicts the probability of a target belonging to a class. It is effective for linearly separable data.
    - **Performance:** Achieved **77% accuracy**. It provided a good baseline model but struggled to capture complex, non-linear relationships in the data.
    - **Parameters Tuned:** `C` (regularization strength), `solver`, `max_iter`.
    - **Impact:**
        - The best parameter combination (`C=10`, `max_iter=300`, `solver='lbfgs'`) allowed the model to generalize better by balancing bias and variance.
        - Test accuracy improved slightly to **77.54%**, showing that the optimized model handled the data's complexity better.
2. **K-Nearest Neighbors (KNN):**
    - **About the Algorithm:** KNN is a non-parametric model that classifies samples based on the majority class among its nearest neighbors. It is simple yet effective for small, well-separated datasets.
    - **Performance:** Achieved **72% accuracy**. While it performed reasonably well, it was computationally expensive for this dataset, and accuracy was lower compared to other models.
    - **Parameters Tuned:** `n_neighbors` (number of neighbors), `weights` (uniform/distance-based), `algorithm` (search strategy).
    - **Impact:**
        - Best parameters (`n_neighbors=9`, `weights='distance'`, `algorithm='auto'`) improved the model's performance, achieving **74.0% accuracy**.

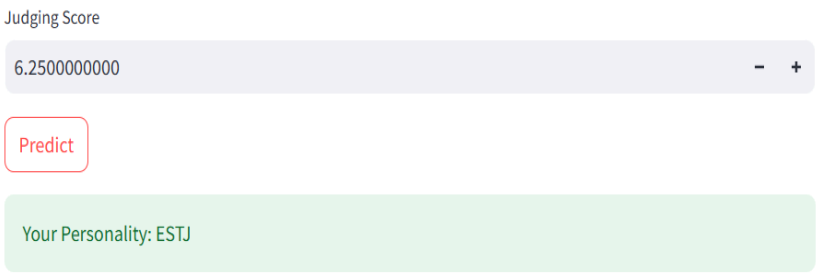
        - Using `distance` weights gave more importance to closer neighbors, leading to better predictions.
    -
3. **Support Vector Machine (SVM):**
    - **About the Algorithm:** SVM creates a decision boundary (hyperplane) to separate classes in a high-dimensional space. It is particularly effective for non-linear data when used with kernels.
    - **Performance:** Achieved **80% accuracy**, showing robust performance. SVM captured non-linear relationships better than Logistic Regression and KNN but was computationally intensive.
    - **Performance:** Achieved **80% accuracy**, demonstrating robust performance by capturing non-linear relationships better than Logistic Regression and KNN. However, it was computationally intensive for the dataset.
    -
4. **Naive Bayes:**
    - **About the Algorithm:** Naive Bayes is a probabilistic classifier that applies Bayes' Theorem with the assumption of feature independence.
    - **Performance:** Achieved **58% accuracy**, which was the lowest among all models. Naive Bayes struggled with the complex relationships between features in the dataset.
    - **Parameters Tuned:** `var_smoothing` (variance added to prevent zero probabilities).
    - **Impact:**
        - Best parameter (`var_smoothing=1e-07`) slightly improved the model's performance to **62.45% accuracy**.
        - Despite optimization, Naive Bayes struggled with the dataset's complexity due to its independence assumption.
5. **Decision Tree Classifier:**
    - **About the Algorithm:** A Decision Tree splits the data into subsets based on feature values, building a tree-like structure for classification. It handles both numerical and categorical data well.
    - **Performance:** Achieved **83% accuracy**, the highest among all models. It effectively captured non-linear relationships and was interpretable but required hyperparameter tuning to prevent overfitting.
    - **Parameters Tuned:** `criterion` (split measure), `max_depth` (tree depth), `min_samples_split` (minimum samples to split), `min_samples_leaf` (minimum samples per leaf).
    - **Impact:**
        - Best parameters (`criterion='entropy', max_depth=10, min_samples_split=10, min_samples_leaf=4`) significantly enhanced the model's ability to generalize, achieving **87.44% accuracy**.
        - Limiting tree depth and ensuring sufficient samples per split/leaf reduced overfitting and improved generalization.

Best Model: Decision Tree Classifier

- **Why It Was the Best Model:**
  - **Performance:** Decision Tree Classifier achieved the highest test accuracy of **87.44%**, outperforming other models significantly.
  - **Robustness:** By optimizing parameters such as `max_depth` and `min_samples_leaf`, the model was able to balance complexity and accuracy, avoiding overfitting while capturing non-linear relationships.
  - **Interpretable:** Decision Trees are inherently interpretable, making it easier to understand the relationships between features and predictions.
  - **Versatility:** It handled both categorical and numerical data effectively and performed well across all personality types, as shown by its high precision, recall, and F1-scores.

Interface:





Conclusion:

The analysis successfully demonstrated the use of demographic and personality-related features to predict Myers-Briggs Type Indicator (MBTI) personality types. Key features like Introversion, Thinking, and Judging scores played a significant role in distinguishing personality types. The Decision Tree Classifier emerged as the best-performing model, achieving an accuracy of **87.44%**, thanks to its ability to handle both categorical and numerical features effectively. Hyperparameter tuning further enhanced the model's generalization and performance. These findings indicate that machine learning can be a valuable tool for personality prediction, providing insights into the relationships between personality traits and external factors such as age, gender, and interests.