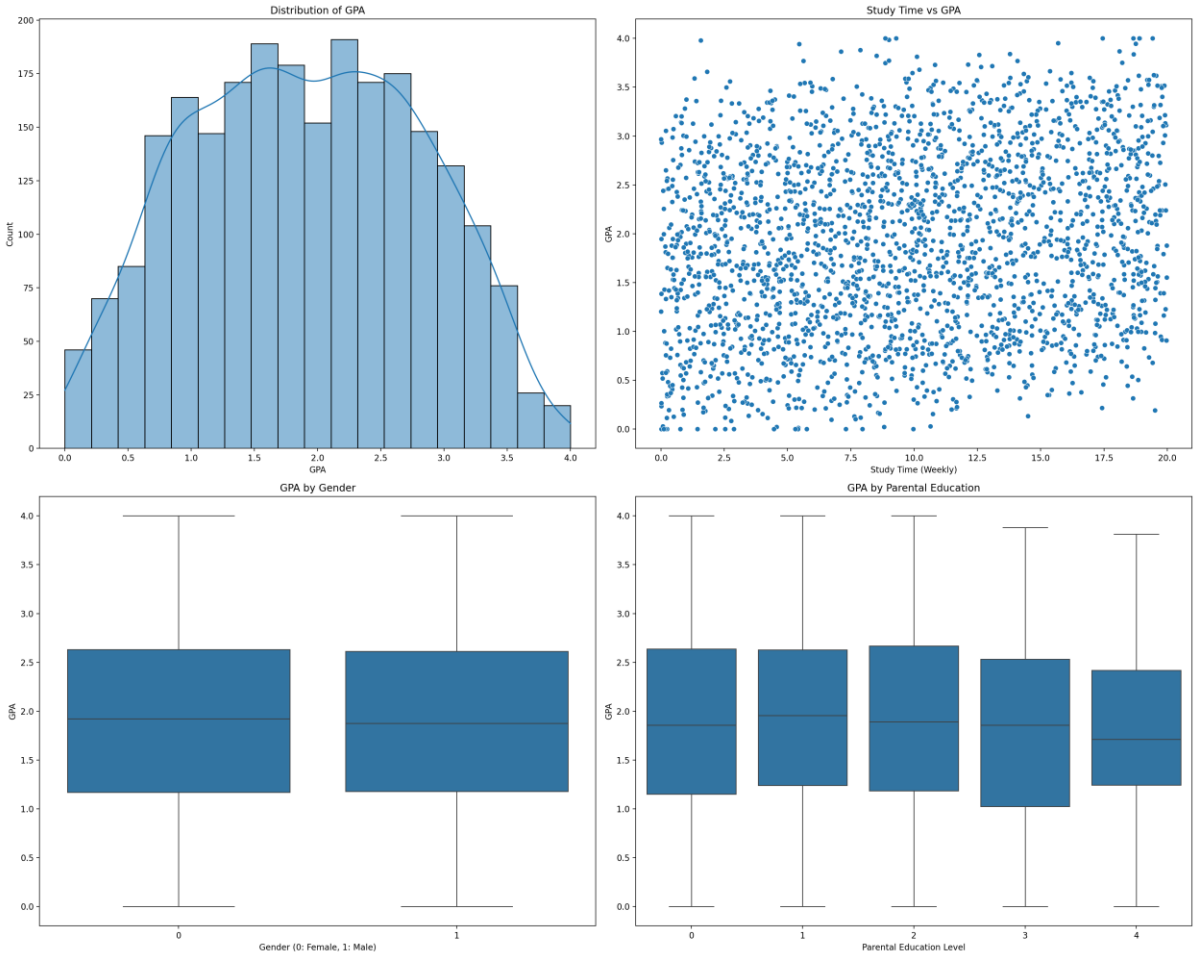
Choose a dataset for classification (e.g., student demographics and performance, customer segmentation, or any other dataset of interest). You can use a public dataset from sources such as Kaggle, UCI Machine Learning Repository, or similar.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

# Calculate evaluation metrics for each model
models = ['Logistic Regression', 'k-NN', 'Decision Tree']

# Accuracy
accuracy_scores = [
    accuracy_score(y_test, y_pred_log_reg),
    accuracy_score(y_test, y_pred_knn),
    accuracy_score(y_test, y_pred_tree)
]

# Precision (macro average)
precision_scores = [
    precision_score(y_test, y_pred_log_reg, average='macro'),
    precision_score(y_test, y_pred_knn, average='macro'),
    precision_score(y_test, y_pred_tree, average='macro')
]

# Recall (macro average)
recall_scores = [
    recall_score(y_test, y_pred_log_reg, average='macro'),
    recall_score(y_test, y_pred_knn, average='macro'),
    recall_score(y_test, y_pred_tree, average='macro')
]
```

```
# F1-Score (macro average)
f1_scores = [
    f1_score(y_test, y_pred_log_reg, average='macro'),
    f1_score(y_test, y_pred_knn, average='macro'),
    f1_score(y_test, y_pred_tree, average='macro')
]

# Create a DataFrame with the metrics
metrics_df = pd.DataFrame({
    'Model': models,
    'Accuracy': accuracy_scores,
    'Precision': precision_scores,
    'Recall': recall_scores,
    'F1-Score': f1_scores
})

# Melt the DataFrame for easier plotting
metrics_melted = metrics_df.melt(id_vars='Model', var_name='Metric', value_name='Score')

# Create a scatter plot comparing the models
plt.figure(figsize=(10, 6))
sns.scatterplot(data=metrics_melted, x='Model', y='Score', hue='Metric', style='Metric', s=100)

# Add title and labels
plt.title('Model Comparison: Accuracy, Precision, Recall, F1-Score', fontsize=14)
plt.ylabel('Score')
plt.xlabel('Model')

# Show plot
plt.show()
```



Logistic Regression is suitable for interpretable models but struggles with complex relationships in data.

k-NN is a good choice for non-parametric classification but suffers from computational inefficiency and sensitivity to feature scaling.

Decision Tree outperforms the other two models in this dataset but requires careful tuning to avoid overfitting.