Customer Churn

We excluded the 'CustomerID' attribute from our churn prediction analysis as it's not pivotal. The dataset has no missing values. Using OneHotEncoder, we handled categorical and boolean columns. Employing Pipeline and ColumnTransformer, we established a Machine Learning Pipeline for better workflow organization. We enhanced classifier performance via grid search with cross-validation, identifying optimal parameters through Param_Grid to improve predictive accuracy.

Performance Measurement		AdaBoost		Random Forest
		Decision Tree (Default)	Random Forest	
Time Taken (in sec)		9.56	539.86	636.23
Memory Used (in KB)		1200.0	43524.0	6704.0
Accuracy Score	On Training Data (0.8)	0.54	1.00	1.00
	On Test Data (0.2)	0.53	0.52	0.51
Optimal Parameters (With GridSearchCV)		classifier_algorithm: SAMME, classifier_estimator: None, classifier_learning_rate: 0.1, classifier_n_estimators: 100	classifier__base_ estimator_max_depth: 10, classifier_learning_rate: 0.01, classifier_n_estimators: 50	classifier_bootstrap: False, classifier_max_depth: 20, classifier__min_samples_leaf: 2, classifier__min_samples_split: 5, classifier__n_estimators: 200

Observations and Inference:

1. **Robustness of Ensemble Methods**: Adaboost and random forests, both ensemble methods, combine weak learners to create stronger models. Similar performance on accuracy scores suggests their effectiveness in capturing underlying data patterns.

2. **Simplicity Wins**: Similar accuracy suggests that AdaBoost's sequential error correlation may not significantly outperform random forests' parallel decision tree ensembles for this dataset.

Supermarket Sales

We excluded the 'InvoiceID' column as it's irrelevant for predicting gender and rating. We also ensured there are no missing values.

Gender Prediction

We used LabelEncoder and OneHotEncoder to handle categorical data. These were integrated into our workflow using a pipeline. Prior to modelling, we handled outliers in the dataset by applying Winsorization to the numerical columns. To optimize classifier performance, we employed grid search with cross-validation and Param_grid for hyperparameter tuning.

Performance Measurement		Decision Tree	Random Forest
Time Taken (in sec)		2.69	38.57
Memory Used (in KB)		2048	384
Accuracy Score	On Training Data	0.88	0.81
	On Test Data	0.50	0.50
Optimal Parameters (With GridSearchCV)		criterion='entropy', min_samples_split=10, random_state=42	max_depth=5, random_state=42

Observations and Inference

From inspecting the above table, we see that the accuracy for both decision trees and random forest is more than 0.80 for training data, but it falls to 0.50 on test data. Hence, we can conclude:

1. The dataset's complexity or noisy features might be causing both models to overfit. Despite random forests usually being more robust, the shared decrease in accuracy implies significant challenges from the dataset for models.
2. Both decision trees and random forests can capture non-linear relationships, yet if the dataset harbors intricate non-linear relationships, it may lead to overfitting and diminished generalization on the test set.
3. Data quality issues, like errors or biases, could affect both training and test data, impacting model performance. Addressing these issues through preprocessing can mitigate their impact and enhance model performance.

Rating Prediction

We first setup the simple linear regression model and then established the Decision Tree Regressor with grid search and cross-validation to examine the measurements and calculations. Prior to modelling, we handled outliers in the dataset by applying Winsorization to the numerical columns.

Metric	Linear Regression	Decision Tree Regressor
Mean Absolute Error	1.49057	1.50875
Mean Square Error	2.96844	3.05122
R2 – Score	-0.01274	-0.04098
Adjusted R – Squared	-0.06071	-0.09029

Observations and Inference

From the Visual inspection of Actual vs Predicted Ratings (available in the Jupyter Notebook) and the values of metrics shown above:

1. **Linear Regression**: The plot shows a scattered pattern, hinting at potential issues with model fit, complexity, or violated assumptions. Negative Adjusted R-Squared and R2 scores further confirm this observation.
2. **Decision Tree Regressor**: In the plot, we notice parallel lines close to the x-axis, suggesting systematic bias or limited flexibility in the model's predictions, failing to capture the variability in the target variable.
3. **Multicollinearity**: The correlation matrix indicates potential multicollinearity issues between highly correlated independent variables. For instance, "Tax" and "Total" have a correlation of 1.0, indicating perfect correlation. Including both in a linear regression model could result in multicollinearity, impacting the stability and interpretability of model coefficients.