Name: Sai Ram Gunturu
ID: C00313478

Note: This is a completely new notebook created by me.

Decision Tree Classifier - Predicting Customer Behavior

Data Analytics & Algorithms Log

- **Algorithm Used:** Decision Tree Classifier
- **Datasets Used:**
  - **Bank Marketing Dataset** (Predicting term deposit subscriptions)
  - **Credit Card Default Dataset** (Predicting loan defaults)
- **Framework:** CRISP-DM

Phases	Methods applied	Reason for the specific method	Duration	Difficulty level (1-10)
Datasets selection	Searched on the internet for wide range of datasets that suits for decision trees	With unsuited datasets we cannot get any valuable outcomes	40 min	5
Data preprocessing	Checked the null values, unnecessary columns and removed it.	Null values may lead to mis prediction, selected only meaningful columns that lead for a better prediction	40 min	6
Datasets preparation for training	LabelEncoder	Used Labelencoder for categorical variables to transform to numerical form.	1 hour	7
	StandardScaler	For numerical values I used a standard scalar for equal contribution during the model training.		
	Stratified Split	Used Stratified split traditional method according to the datasets		
Prediction with Base Model	Used default parameters and no max depth was set.	To analyse how the base model was predicting.	10 min	4

Further Enhancements (Bank dataset)	Implemented Hyperparameter tuning (GridSearchCV)	I used GridSearchCV since I used a medium sized dataset and wanted to try all possible combinations to get the best parameters.	4 hours	9
	Feature Selection	Used feature selection to reduce the complexity of model		
	Class Weight Adjustments	Although I haven't got effective results through feature selection due to data imbalance, I started focusing on data balance by class weight adjustments.		
	Pruning	To reduce overfitting, I used pruning to reduce less important splits.		
	Bagging	To stabilize the model, I used bagging		
Further Enhancements (Credit Card Dataset)	Implemented hyperparameter tuning (GridSearchCV)	After implementation of hyperparameter tuning, overall accuracy was significantly increased with a slight decrease in recall.	3 hours 30 min	9
	Class Weight Adjustments	To improve recall, I applied class weight adjustments which highly improved the recall with drastic decrease in overall model accuracy.		

	Pruning	With post-pruning I reduced overfitting and increased the overall accuracy with a drastic decrease in recall.		
	Bagging	Bagging didn't contribute much for recall and slightly increased the overall accuracy.		
Conclusions		Due to many disadvantages in decision trees that are leading to major recall issues. On the other hand, data imbalance has greatly affected the overall performance of the model, moving forward will try to use the same datasets in Random Forest algorithm to improve the prediction performance	30 min	6

1. Business Understanding

Objective

I wanted to create a **Decision Tree Model** to:

- Predict whether a **bank customer** will subscribe to a term deposit.
- Predict whether a **credit card holder** will default on payment.

Decision Trees were a **good starting point** for classification tasks as they are **easily interpretable** as well as **flexible**.

Challenges I Faced

- **Class Imbalance** → In the Bank dataset, **only 11.7% of customers subscribed**, thus making it difficult for the model to correctly predict the minority class.
- **Overfitting** → When **not tuned correctly**, Decision Trees **tend to overfit,** therefore making poor generalization.

2. Data Understanding

- **Dataset 1: Bank Marketing Dataset**
  - **Total Records:** 45,211
  - **No missing values**
- **Target Variable:** y (Subscription status: 1 = Yes, 0 = No)

Feature	Type	Description
Age	Numerical	Age of the client
Job	Categorical	Type of job
Marital	Categorical	Marital status
Education	Categorical	Level of education
Balance	Numerical	Account balance
Housing	Categorical	Has a housing loan?
Loan	Categorical	Has a personal loan?
Duration	Numerical	Call duration (seconds)

3. Data Preparation

- Used '*LabelEncoder*' to **encode categorical variables**
- **Standardized the numerical features** with '*StandardScaler*'.
- **Split data**: **80% train | 20% test (stratified split)**.

4. Baseline Decision Tree Model

Algorithm: DecisionTreeClassifier (Default Parameters)

- **Criterion:** Gini impurity

- **Max Depth:** Not set (Caused Overfitting)

Baseline Model Performance

Metric	Value
Accuracy	87.7%
Precision (Subscription - Class 1)	48%
Recall (Subscription - Class 1)	48%
F1-score (Subscription - Class 1)	48%

What I Noticed:

- **subscribers** had high accuracy, but low recall **(48%)**.
- **Overfitting Problem** → Decision Tree fits the training data too closely but poorly generalizes.

5. Model Improvements & Adjustments

5.1 Hyperparameter Tuning (GridSearchCV)

- I tuned: ***max_depth, min_samples_split, criterion, min_samples_leaf.***

- **Best Parameters Found:**

  {'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 10}

Performance After Tuning:

Metric	Value
Accuracy	89.68%

Recall (Class 1 - Subscription)	40%

Impact:

- Improved Accuracy from **87.7% → 89.68%**
- Precision and recall increased **slightly** but still **lots of false negatives**.

5.2 Feature Selection

I reduced the complexity of model by picking the **top ten most important features.**

Impact:

- **Less complex model** → Less Training Time.
- **Similar accuracy, but even more interpretable.**

5.3 Class Weight Adjustment (Dealing with Imbalance)

I modified **class weights** to put more emphasis on correctly predicting subscriptions, in order to **increase recall for people who subscribe.**

Performance After Class Weight Adjustment

Metric	Value
Accuracy	87.3%
Recall (Class 1 - Subscription)	66%
F1-score (Class 1 - Subscription)	55%

Impact:

- **Recall - Class 1 increased from 48% → 66%** (fewer false negatives).
- **Accuracy decreased a little more (89.6% → 87.3%)**, but **more balanced classes**.

5.4 Post-Pruning (Reducing Overfitting)

Used **cost complexity pruning** to apply **post-pruning.**

Best Alpha Found: *0.0028372626868269427*

Performance of Pruned Model

Metric	Value
Accuracy	89.2%
Recall (Class 1 - Subscription)	56%

Impact:

- **Less overfitting** → Model is **more generalizable**.
- **Slight enhancement of Class 1 recall (48% → 56%)**.

5.5 Bagging for Stability

I used **Bagging (Bootstrap Aggregation)** with **100 Decision Trees** to **reduce the variance**.

Bagging Model Performance

Metric	Value
Accuracy	88.8%
Recall (Class 1 - Subscription)	61%

Impact:

- **Lower variance** → **More stable** model.
- **Recall increased (56% → 61%)** decreasing **false negatives**.

6. Final Model Performance Comparison

Model	Accuracy	Recall (Class 1)	F1-score (Class 1)
Baseline Model	87.7%	48%	48%
GridSearch Optimized	89.68%	40%	48%
Balanced Model (Class Weights)	87.3%	66%	55%
Final Pruned Model	89.2%	56%	55%
Bagging Decision Tree	88.8%	61%	56%

Final Decision: Bagging Model produced the best results.

7. Transitioning to a Different Dataset: Credit Card Default Prediction

Now that I've done testing on the **bank dataset**, I want to take a different classification problem to check how **Decision Trees perform on financial data**.

Predicting whether a credit card holder will default on their payment.

7.1 Load and Explore the Dataset

Dataset Overview

- **Total Records:** 30,000
- **No missing values**
- **Class Imbalance:** 22% customers defaulted (Class 1) and 78% customers not defaulted (Class 0).

Feature	Description
LIMIT_BAL	Credit limit of the customer
SEX	Gender (1 = Male, 2 = Female)
EDUCATION	Education level (1 = Graduate, 2 = University, 3 = High School, 4 = Others)
MARRIAGE	Marital status (1 = Married, 2 = Single, 3 = Others)
AGE	Age of the client
PAY_0 to PAY_6	Payment history over the last six months
BILL_AMT1 to BILL_AMT6	Past monthly bill statements
PAY_AMT1 to PAY_AMT6	Amount paid in previous months
Y	Target variable (1 = Default, 0 = No Default)

Observation: This dataset has categorical and numerical features which need some appropriate preprocessing before training a model.

7.2 Data Preprocessing

- **I have dropped ID column** (Not relevant for modeling).
- **Looked for missing values** (None found).
- **Changed categorical variables** *(SEX, EDUCATION, MARRIAGE, PAY_X)* into numerical equivalents.
- **Standardized numerical features** *(LIMIT_BAL, BILL_AMT, PAY_AMT)* to improve model performance.
- **Divided data into a Training set (80%) and Testing set (20%)**.

Processed Dataset Summary

Data Split	Shape
Total Records After Processing	30,000
Training Set Size	24,000
Test Set Size	6,000
Feature Count (excluding target)	23

Insight: Only 22% of customers defaulted. Hence, we need to consider class imbalance that might hamper recall of defaulters (Class 1).

7.3 Performance of Baseline Decision Tree Model

I trained a **simple Decision Tree model** to establish baseline performance before I tuned.

Baseline Model Performance

Metric	Value
Accuracy	71.5%
Precision (Class 0 - No Default)	83%
Recall (Class 0 - No Default)	80%
Precision (Class 1 - Default)	37%
Recall (Class 1 - Default)	41%
F1-score (Class 1 - Default)	39%

Observation:

- The model works effectively **for non-defaulters (Class 0),** however recall for **defaulters (Class 1) is low (41%)**, i.e. quite a lot of actual defaulters are misclassified as non-defaulters.

7.4 Hyperparameter tuning (GridSearchCV)

To improve performance, I used **GridSearchCV** to fine-tune the following hyperparameters:

Criterion*: gini, entropy*

- *Max Depth: 5, 10, 15*
- *Min Samples Split: 2, 5, 10*
- *Min Samples Leaf: 1, 2, 5*

Best Parameters Found

{'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 2}

Performance After Hyperparameter Tuning

Metric	Value
Accuracy	81.8%
Precision (Class 1 - Default)	66%
Recall (Class 1 - Default)	36%
F1-score (Class 1 - Default)	46%

Observation: Accuracy increased **from 71.5% → 81.8%** but **Class 1** recall decreased from **(41% → 36%)**, i.e. **more false negatives**.

7.5 Balanced Model (Class Weights)

I set **class weights ({0:1, 1:3})** to help balance the prediction, to **increase recall for defaulters**.

Performance After Class Weight Adjustment

Metric	Value
Accuracy	78.1%
Recall (Class 1 - Default)	52%
Precision (Class 1 - Default)	50%
F1-score (Class 1 - Default)	51%

Observation:

- **Recall increased from 36% → 52%**, i.e. **fewer missed defaulters**.
- **Accuracy dropped slightly (81.8% → 78.1%)**, while class balance improved.

7.6 Post-Pruning: Final Pruned Model

I used **cost-complexity pruning**, to **reduce overfitting**,

- **Best Pruning Alpha Found**: *0.0006939517173115209*

Pruned Model Performance:

Metric	Value
Accuracy	81.6%
Recall (Class 1 - Default)	35%
Precision (Class 1 - Default)	66%
F1-score (Class 1 - Default)	46%

Observation:

- Accuracy was increased **(78.1 → 81.6%)**, showing that pruning had **prevented overfitting**.
- **Recall decreased slightly (52% → 35%)**, resulting in **more missed defaulters**.

7.7 Final Improved Model (Bagging Decision Tree)

In order to **reduce variance even more and improve stability**, I used **Bagging (Bootstrap Aggregation) with 100 Decision Trees**.

Performance of Final Bagging Model:

Metric	Value
Accuracy	81.8%
Precision (Class 1 - Default)	67%
Recall (Class 1 - Default)	35%
F1-score (Class 1 - Default)	46%

Observation:

- **Bagging decreased variance** enhancing model **stability.**
- **Precision improved from (66% → 67%)**, however **recall was low (35%)**, that is, many defaulters were still misclassified.

7.8 Conclusion

- **Data imbalance played a key role** in making recall enhancement complex.
- **Decision Trees also had limitations** leading to **poor recall for Class 1**, even after utilization of tuning and resampling techniques.
- **Bagging improved stability**, but **false negatives** remained a challenge.
- **Moving forward, we will try to improve results using Random Forest** to **reduce variance, improve recall and have better generalization.**