Name: Sai Ram Gunturu
ID: C00313478

Random Forest Classifier - Credit Card Default Prediction

Data Analytics & Algorithms Log

- **Algorithm Used:** Random Forest Classifier
- **Dataset Used:** Credit Card Default Dataset
- **Framework:** CRISP-DM

---

1. Business Understanding

Objective: To build a **Random Forest Model** to predict whether a **credit card holder will default on their payment**. To improve efficiency, I Optimized Hyperparameters, applied Class Balancing techniques, experimented with feature selection to optimize recall for defaulters.

Challenges I Faced

- **Class Imbalance → Only 22%** of customers have defaulted, resulting in **biased predictions**.
- **Feature Correlation** → There can be redundant information in features like **BILL_AMT1-6, PAY_0-6**.
- **Overfitting Risk** → Too Many Trees → Random Forest can become **computationally costly**.

2. Data Understanding

- **Dataset Overview**
- **Total Records: 30,000**
- **No missing values**
- **Converted categorical labels: (Default → 1, No Default → 0)**

Initial Class Distribution

Class	Count	Percentage
No Default (0)	23,660	78%
Default (1)	6,340	22%

Key Insight: The number of defaulters are much fewer and hence I needed to employ **class balancing techniques** to help improve recall.

3. Data Preparation

- **Removed irrelevant columns** (ID column).
- **Label Encoding was used to encode categorical variables**.

- Used **StandardScaler to standardize numerical features**.
- **Did a train-test split (80% train/ 20% test)**.

Processed Dataset Summary

Data Split	Total Records	Features	Train Size	Test Size
Credit Card	30,000	23	24,000	6,000

Key Insight: There were **many correlated features** in the dataset so I later tested feature selection.

4. Baseline Random Forest Model

Algorithm: *RandomForestClassifier (Default Parameters)*

Baseline Model Performance

Metric	Value
Accuracy	81.2%
Precision (Class 1 - Default)	63%
Recall (Class 1 - Default)	36%
F1-score (Class 1 - Default)	46%

What I Noticed:

Higher accuracy (81.2%) than Decision Tree (71.5%), but **recall is still low (36%)**, meaning **lots of defaulters are classified wrong**.

5. Using Class Balancing (Weighted Random Forest)

Because recall was low, I added **class weights** ***{0:1, 1:3}*** so that the model would focus on predicting defaults better.

After Class Weight Adjustment - Performance

Metric	Value
Accuracy	79.2%
Precision (Class 1 - Default)	53%
Recall (Class 1 - Default)	54%
F1-score (Class 1 - Default)	54%

Key Impact:

- **Recall improved from 36% → 54%**, so there are **fewer false negatives**.
- **Accuracy dropped a little bit (81.2% → 79.2%)**, which is acceptable since recall increased significantly.

6. Feature Selection for Better Model Performance

To avoid complexity, I reduce features and selected the **top 10 features by importance,** while maintaining performance.

Top 10 Selected Features:

LIMIT_BAL, AGE, PAY_0, PAY_2, PAY_3, BILL_AMT1, BILL_AMT2, PAY_AMT1, PAY_AMT2, PAY_AMT3

Feature-Selected Model Performance

Metric	Value
Accuracy	79.1%
Precision (Class 1 - Default)	53%
Recall (Class 1 - Default)	55%
F1-score (Class 1 - Default)	54%

What I Found:

- **Recall went up slightly from 54% → 55%**, but overall performance remained the same.
- **Accuracy was not significantly affected by Feature selection.**

7. Bagging Random Forest for Stability

I used **Bootstrap Aggregation (Bagging)** with **50 trees** for variance reduction.

Bagging Model Performance

Metric	Value
Accuracy	79.6%
Precision (Class 1 - Default)	54%
Recall (Class 1 - Default)	53%
F1-score (Class 1 - Default)	54%

What I Found:

- **Bagging helped improve model stability** but had little effect on **recall**.
- **Results consistent with weighted and feature-selected models**.

8. Final Model Comparison

Model	Accuracy	Precision (Class 1)	Recall (Class 1)	F1-score (Class 1)
Baseline Decision Tree	71.5%	48%	48%	48%
Optimized Decision Tree	81.8%	66%	36%	46%
Balanced Decision Tree	78.1%	50%	52%	51%
Baseline Random Forest	81.2%	63%	36%	46%

Weighted Random Forest	79.2%	53%	54%	54%
Feature-Selected Random Forest	79.1%	53%	55%	54%
Bagging Random Forest	79.6%	54%	53%	54%

Final Decision: However, **Weighted Random Forest performed well** and has the **best recall improvement (54%) score** considering **balance.**

9. Random Forest Model -- Bank Dataset

Here, after analyzing the **Credit Card Default dataset**, I used the **Random Forest algorithm** on the **Bank Marketing Dataset**.

The task was to **predict** if based on some features, **customers will subscribe a term deposit or not**.

9.1 Baseline Random Forest Model -- Bank Dataset

Algorithm: *RandomForestClassifier (Default Parameters)*

Baseline Model Performance

Metric	Value
Accuracy	90.6%
Precision (Class 1 - Subscribed)	66%
Recall (Class 1 - Subscribed)	42%
F1-score (Class 1 - Subscribed)	51%

Key Observations:

- **More accurate than Decision Tree (90.6% vs. 87.7%)**, which indicates a better overall classification performance in general.
- **Precision increased from 48% (Decision Tree) → 66% (Random Forest)**, i.e. fewer false positives.

- **Recall remained low (42%)**, indicating **many subscribers were classified incorrectly as non-subscriber**.
- **There is still a challenge** with **Class imbalance**– the **majority class (Non-Subscribers)** was favored by the model.

9.2 Hyperparameter Tuning with RandomizedSearchCV

I optimized the below **hyperparameters**:

- n_estimators: **[50, 100, 150]**
- max_depth: **[10, 20, 30]**
- min_samples_split: **[2, 5, 10]**
- min_samples_leaf: **[1, 2, 4]**
- bootstrap: **[True, False]**

Best Parameters Found

{'n_estimators': 150, 'min_samples_split': 5, 'min_samples_leaf': 4, 'max_depth': 30, 'bootstrap': False}

Performance of the Optimized Model

Metric	Value
Accuracy	90.6%
Precision (Class 1 - Subscribed)	65%
Recall (Class 1 - Subscribed)	42%
F1-score (Class 1 - Subscribed)	51%

Key Observations:

- **No major increase in accuracy from baseline** (90.6%) which remained the same.
- **Recall for Class 1 stayed at 42%**, which meant subscribers continued to be **misclassified**.
- **More consistent model performance**, yet **imbalanced data continued to be a problem**.

9.3 Weighted Random Forest (Handling Imbalance)

As recall was low, I applied **class weights {0:1, 1:3}** so that the model would pay more attention to positively predicting subscriptions.

Performance of Weighted Model :

Metric	Value
Accuracy	90.5%
Precision (Class 1 - Subscribed)	59%
Recall (Class 1 - Subscribed)	64%
F1-score (Class 1 - Subscribed)	61%

Key Observations:

- **Subscribers Recall improved from 42% → 64%**, which indicates **fewer false negatives**.
- **Accuracy decreased a little bit (90.6% → 90.5%)** as expected because of the class weighting.
- **F1-score increased to 61%**, i.e. improved **balance between recall and precision**.

9.4 Bagging Random Forest Model

I employed **Bootstrap Aggregation (Bagging)** to **reduce variance even more**,

Bagging Model Performance

Metric	Value
Accuracy	90.6%
Precision (Class 1 - Subscribed)	67%
Recall (Class 1 - Subscribed)	40%
F1-score (Class 1 - Subscribed)	50%

Key Observations:

- **Bagging aided in decreasing the variance of the model** and in turn making the predictions **more stable**.

- **Accuracy stayed at 90.6%**, which means there is **no improvement in overall performance**.
- **Recall decreased a little bit from 42% → 40%**, which indicates **bagging had insignificant differences in regards to minority class predictions**.

9.5 Final Model Comparison -- Bank Dataset

Model	Accuracy	Precision (Class 1)	Recall (Class 1)	F1-score (Class 1)
Baseline Random Forest	90.6%	66%	42%	51%
Optimized Random Forest	90.6%	65%	42%	51%
Weighted Random Forest	90.5%	59%	64%	61%
Bagging Random Forest	90.6%	67%	40%	50%

Final Decision: Weighted Random Forest performed the best (64% recall, 61% F1-score), which balanced both recall and precision.

Conclusion - Random Forest vs Decision Tree Performance

In the end, after testing Random Forest model and Decision Tree model on both datasets (Credit Card Default and Bank Marketing), I analyzed both models, effectiveness, stability, and improvement of recall.

Key Findings & Improvements

Metric	Decision Tree Best Performance	Random Forest Best Performance	Improvement
Bank Dataset Accuracy	89.68%	90.6%	Slight Increase
Bank Dataset Recall (Class 1)	66% (Balanced DT)	64% (Weighted RF)	Slight Drop

Bank Dataset F1-score (Class 1)	56%	61%	Improved

Part 1 Conclusion (Credit Card Default Prediction)

- Random Forest dominated Decision Tree by a margin in accuracy (81.2% vs. 71.5%) and serves a better model for this dataset.
- Weighted Random Forest (Class balancing techniques) improved recall of defaulters from 36% → 54%, lowering false negatives.
- Bagging aided in stabilizing performance, but did not improve recall considerably.
- Class imbalance was still pervasive despite optimization, making recall improvements difficult.

Part 2 Conclusion (Bank Marketing Prediction)

- Due to slight increase in accuracy in random forest than Decision Tree (90.6% vs. 89.68%) we conclude that Random Forest is a better fit for this dataset.
- Weighted Random Forest increased recall from 42% → 64%, thus decreasing incorrectly classified subscribers.
- Class weighting boosted recall while decreasing accuracy a bit —a tradeoff for improved subscriber identification.
- Bagging ensured stability but had no impact on recall, like in the Credit Card dataset.

Challenges Faced

- **Model Performance affected by Class Imbalance**
  - R**ecall for subscribers remained an issue,** even with class weighting.
  - **In identifying positive cases** Imbalanced data **constrained optimizations.**
- **No Significant Improvement in Accuracy**
  - **Random Forest accuracy was roughly the same as with Decision Tree (90.6%)**.
  - The more complex model **didn’t generalize better**.
- **Computational Complexity & Training Time**
  - **Random Forest was much more computationally expensive** than Decision Tree.
  - **GridSearch and Bagging were much slower** with minimal improvements.

Final Thoughts & Next Steps

- Random Forest had better stability but did not significantly outperform Decision Tree.
- Recall improvements for subscribers were limited by Class imbalance.
- Bagging achieved stability, but no significant recall gain.