Recent Research Topics in Finance

Predicting Loan Default Using a Machine Learning Algorithm

Table of Contents

Question 1

The objective of this project is to predict loan default. In this section, we describe the pre-processing steps performed on the original dataset to get it ready for modelling.

Initial Filtering

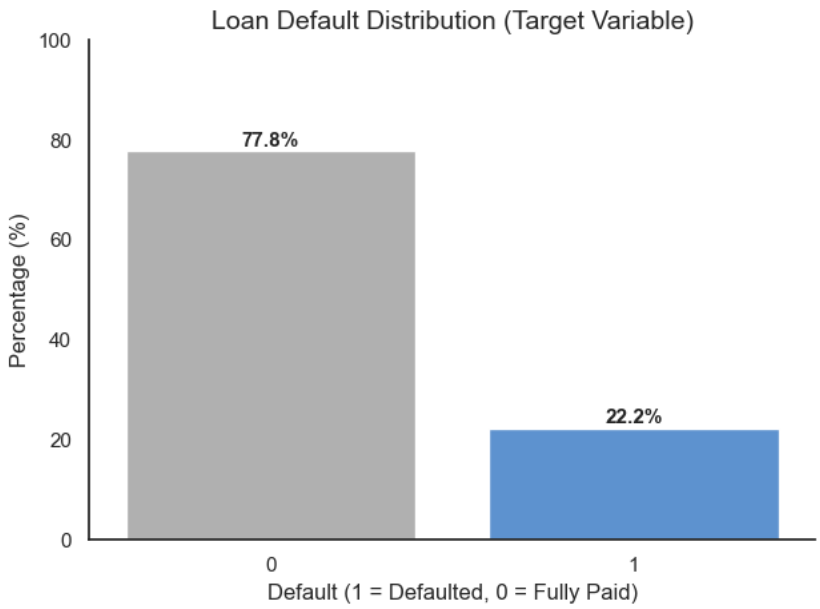
We removed observations where the loan was still active and hadn't yet defaulted. Including these could mislead our models, as such loans may still default in the future. We also considered retaining loans with a balance of 0 (which might indicate full repayment and thus a definite non-default), but the BalanceGross variable was of very low quality and unsuitable for this purpose.

Additionally, we excluded loans with a Term value of 0, as these corresponded to short-term deferrals or microloan programs rather than actual loan durations.

These steps reduced the dataset size from 899,164 to 711,197 observations.

Class Imbalance

Only 22.2% of loans in the dataset resulted in default, introducing a substantial class imbalance. This imbalance risks biasing predictive models towards favouring the majority class (i.e. non-default). Our approach to mitigating this imbalance is discussed in later sections.



Removal of Variables

Prior to exploratory data analysis, we removed several variables that were unsuitable for modelling as follows:

Variable(s)	Reason	Explanation
LoanNr_ChkkDgt, Name	Non-Predictive	ID-like variables with no predictive value.
Minority	Protected	Removed to comply with the Equal Credit Opportunity Act (1974) anti-discrimination laws.
ChgoffDate, ChgOffPrinGr, DisbursementDate, DisbursementGross, BalanceGross	Leakage	Not available at time of application.
Bank	Irrelevant	Not relevant to our model's objective of building models for a specific bank.
NAICS	Redundant	Duplicates information given in 'Industry'.

Data Cleaning and Analysis

The industry variable was mapped to the provided descriptions for clarity and interpretability. Several other variables were converted into binary flags for interpretability as follows:

Variable	New Variable (1 for "Yes", 0 for "No)
NewExist	Is_New
FranchiseCode	Is_Franchise
UrbanRural	Is_Rural
RevLineCr	Is_RevLineCr
LowDoc	Is_LowDoc

All variables were then cast to appropriate data types, and data accuracy tests were carried out. Descriptive statistics were calculated for numeric variables, while categorical variables were reviewed for cardinality. The resulting tables are below:

Numeric

Variable	% Missing	Mean	Mode	Standard Deviation	Minimum	Q1 (25%)	Median (50%)	Q3 (75%)	Maximum	% Outliers	Skewness	Kurtosis
ApprovalFY	0.00	2000.68	2006	5.91	1962	1996	2002	2005	2014	0.40	-0.69	-0.09
Term	0.00	84.12	84	52.12	1	60	84	87	461	1.01	1.54	3.10
NoEmp	0.00	10.59	1	78.12	0	2	4	9	9999	0.13	79.18	7557.93
CreateJob	0.00	9.48	0	265.33	0	0	0	1	8800	0.09	33.02	1090.02
RetainedJob	0.00	11.81	0	265.34	0	0	1	4	8800	0.09	32.98	1088.43
GrAppv	0.00	133,717.04	50,000	199,049.22	200	30,000	65,000	150,000	5,000,000	2.39	4.35	35.05
SBA_Appv	0.00	99,868.20	25,000	153,197.57	100	17,500	45,000	116,250	4,500,000	2.54	4.29	41.20
Is_New	0.00	0.29	0	0.45	0	0	0	1	1	0.00	0.91	-1.16
Is_Franchise	0.00	0.06	0	0.23	0	0	0	0	1	5.53	3.89	13.14
Is_Rural	37.74	0.17	0	0.38	0	0	0	0	1	0.00	1.73	1.00
Is_LowDoc	0.00	0.15	0	0.35	0	0	0	0	1	0.00	1.98	1.94
Is_RevLineCr	0.00	0.26	0	0.44	0	0	0	1	1	0.00	1.10	-0.80

Categorical

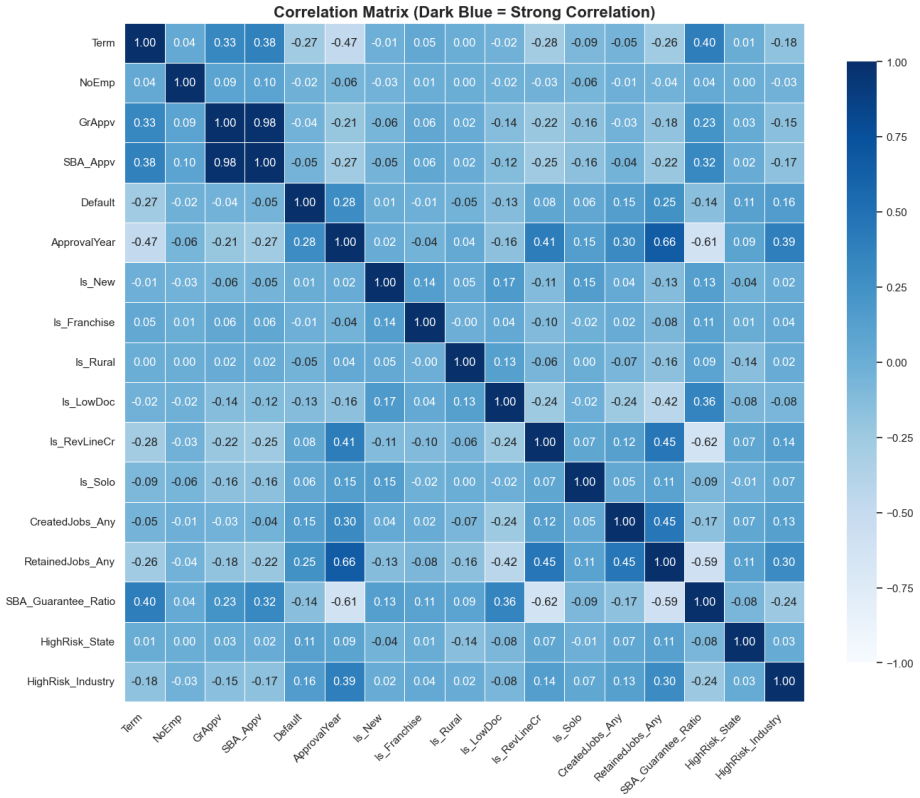
Variable	% Missing	Distinct Categories
City	0.00	28,952
State	0.00	51
Zip	0.00	31,904
Industry	0.00	21
BankState	0.00	56

Feature Engineering

Below is a summary table of how each of these variables were handled, based on the tables above and further analysis:

Variable	Decision	Explanation	Final Variable Name(s)
Term	Unchanged		Term
GrAppv	Unchanged		GrAppv
Is_New	Unchanged		Is_New
Is_Franchise	Unchanged		Is_Franchise
Is_RevLineCr	Unchanged		Is_RevLineCr
Is_LowDoc	Unchanged		Is_LowDoc
NoEmp	Unchanged	Right-skewed but kept as is for interpretability reasons. Added Is_Solo to flag single-employee businesses.	NoEmp, Is_Solo
SBA_Appv	Unchanged	Created SBA_Guarantee_Ratio, which represents the share of the loan backed by the SBA. Calculated as SBA_Appv / GrAppv.	SBAAppv, SBA_Guarentee_Ratio
CreateJob	Adjusted	Converted to CreatedJobs_Any to avoid ethical bias related to optimistic or strategic raw numeric responses. Indicates if any jobs are expected to be created due to the loan.	CreatedJobs_Any
RetainedJob	Adjusted	Same approach as above, RetainedJobs_Any flags whether any jobs are expected to be retained due to the loan.	RetainedJobs_Any
State	Adjusted	Binary flag of HighRisk_State based on a high mean default rate (>25%) and sufficient sample size (>= 1,000) to reduce dimensionality.	HighRisk_State
Industry	Adjusted	Almost identical to above, but also speficially excluded the “Other” category from this.	HighRisk_Industry
City	Dropped	Too granular with sparse observations, making it prone to overfitting.	
Zip	Dropped	Dropped for same reason as above, along with ethical concern of potentially indirectly encoding sensitive demographic information (e.g. race).	
BankState	Dropped	Redundant as already have State, which provides more relevant geographic information for our modelling goal.	
ApprovalDate	Dropped	Year excluded to avoid outdated macro trends, month/quarter had flat default rates so non-predictive.	
ApprovalFY	Dropped	Same reason as above	
Is_Rural	Dropped	37.74% of values missing. Only populated post-2000, indicating strong temporal bias.	

Multicollinearity Analysis



Multicollinearity analysis is done in order to identify highly correlated variables that may distort model coefficients, reduce interpretability, or introduce instability in algorithms. The correlation matrix above shows that there's only one strongly collinear pair. It's described in the following table:

Variables	Correlation Value	Decision Made
GrAppv, SBA_Appv	0.98	Retained GrAppv since it contains total loan size, which is more complete and better suits our modelling objective.

Conclusion

Following all cleaning, transformation, and variable selection steps, we retained 13 predictor variables. This number reflects a careful trade-off between predictive accuracy and computational efficiency. The variables with the strongest correlation to default ($|\rho| > 0.10$), and thus quite likely to be predictive for modelling, are listed below:

Variable	Correlation With Default
Term	-0.27
RetainedJobs_Any	0.24
HighRisk_Industry	0.16
CreatedJobs_Any	0.14
SBA_Guarantee_Ratio	-0.14
Is_LowDoc	-0.13
HighRisk_State	0.11

Question 2

Logistic Regression on the Full Dataset

A logistic regression model was fitted to the full dataset to estimate the probability of loan default, with the dependent variable binary-coded such that Default = 1 and Non-default = 0.

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

The model used all features retained after EDA, and performance was evaluated across four probability thresholds. Corresponding confusion matrices are provided.

Actual Loan Status

10%	Default	No Default
Predicted Loan Status: Default	143,771	358,301
Predicted Loan Status: No Default	13,448	185,629

Actual Loan Status

20%	Default	No Default
Predicted Loan Status: Default	127,946	192,984
Predicted Loan Status: No Default	29,273	350,946

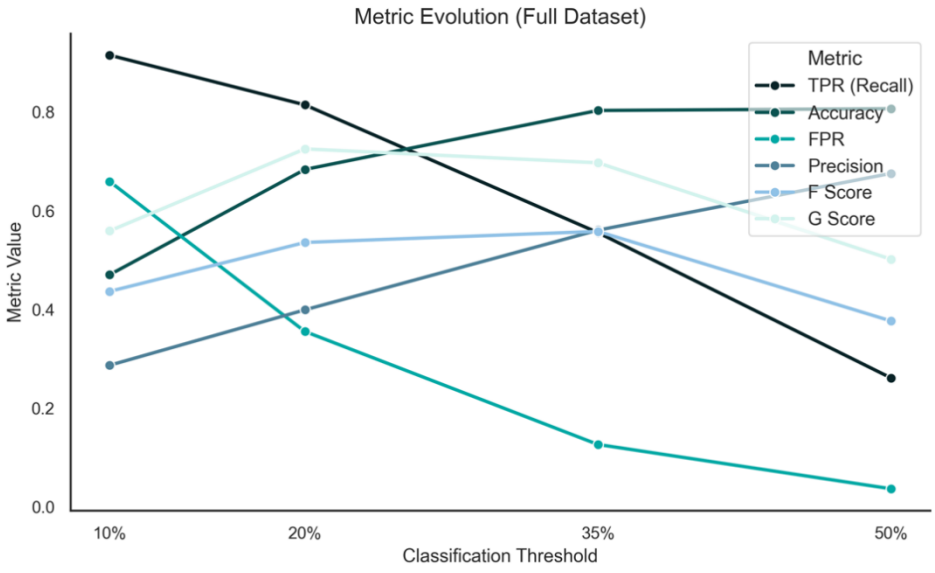
Actual Loan Status

35%	Default	No Default
Predicted Loan Status: Default	87,273	68,548
Predicted Loan Status: No Default	69,946	475,382

Actual Loan Status

50%	Default	No Default
Predicted Loan Status: Default	41,003	19,743
Predicted Loan Status: No Default	116,216	524,187

The chart below visualises model performance across thresholds, highlighting the trade-offs involved.



The following table summarises key metrics used to determine the optimal threshold. Here, the true positive rate (TPR) reflects the model's ability to detect actual defaults, while the false positive rate (FPR) shows how often non-defaulting loans are incorrectly flagged. The false negative rate (FNR) reflects missed defaults - loans predicted as safe that ultimately default. Given defaults comprise just 22.2% of the dataset, the model operates in a class-imbalanced setting where higher TPRs often increase FPR.

	Probability Threshold	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Full Dataset	50%	0.2609	0.0369	0.7391	0.9631	0.8057	0.6716	0.3759	0.5013
	35%	0.5547	0.1257	0.4453	0.8743	0.8026	0.5605	0.5576	0.6964
	20%	0.8137	0.3528	0.1863	0.6472	0.6846	0.4000	0.5363	0.7257
	10%	0.9145	0.6570	0.0855	0.3430	0.4711	0.2869	0.4368	0.5601

Lower thresholds yield higher TPR but more false positives. Precision improves at higher thresholds as false positives decline. The F score, balancing precision and recall, reflects this trade-off. From a lender's perspective, false negatives i.e. missed defaults, are more costly than false positives, which merely reduce lending volume.

A 20% threshold strikes a practical balance: capturing over 80% of defaults while maintaining a much lower FPR than 10%. It also achieves a strong F score and the highest G score, signalling a favourable sensitivity–precision trade-off. Its FNR is also substantially lower than at 50%, a crucial factor in default prediction. Operationally, this threshold reflects a cautious lending strategy: flagging loans with moderate default probability while keeping false alarms manageable. This risk-aware stance aligns with the SBA's partial guarantee and helps protect the bank from overexposure.

70:30 Train Test Split and Cross-Validation

To assess out-of-sample performance, we used a 70:30 train-test split. This simulates real-world deployment by training the model on historical data and testing it on unseen loans.

Actual Loan Status

Predicted Loan Status (10%)	Default	No Default
Default	43,144	107,997
No Default	4,022	55,182

Actual Loan Status

Predicted Loan Status (20%)	Default	No Default
Default	38,459	58,023
No Default	8,707	105,156

Actual Loan Status

Predicted Loan Status (35%)	Default	No Default
Default	26,079	20,366
No Default	21,087	142,813

Actual Loan Status

Predicted Loan Status (50%)	Default	No Default
Default	12,185	5,853
No Default	34,981	157,326

	Probability Threshold	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
70:30 Training Test Split	50%	0.2587	0.0364	0.7413	0.9636	0.8056	0.6727	0.3737	0.4993
	35%	0.5529	0.1246	0.4471	0.8754	0.8031	0.5619	0.5574	0.6957
	20%	0.8150	0.3541	0.1850	0.6459	0.6839	0.3995	0.5362	0.7256
	10%	0.9147	0.6605	0.0853	0.3395	0.4685	0.2858	0.4356	0.5572

Results were consistent: lower thresholds increased TPR but also raised FPR. Again, 20% stood out, achieving strong TPR with lower FPR than 10%, along with the second-highest F score and improved accuracy, further supporting its selection.

K-fold Cross Validation

To estimate performance on unseen data, we implemented stratified k-fold cross-validation. This splits the training set into k equally sized folds, maintaining class proportions. The model is trained on k - 1 folds and validated on the remaining one. This is repeated k times so each fold serves once as a validation set.

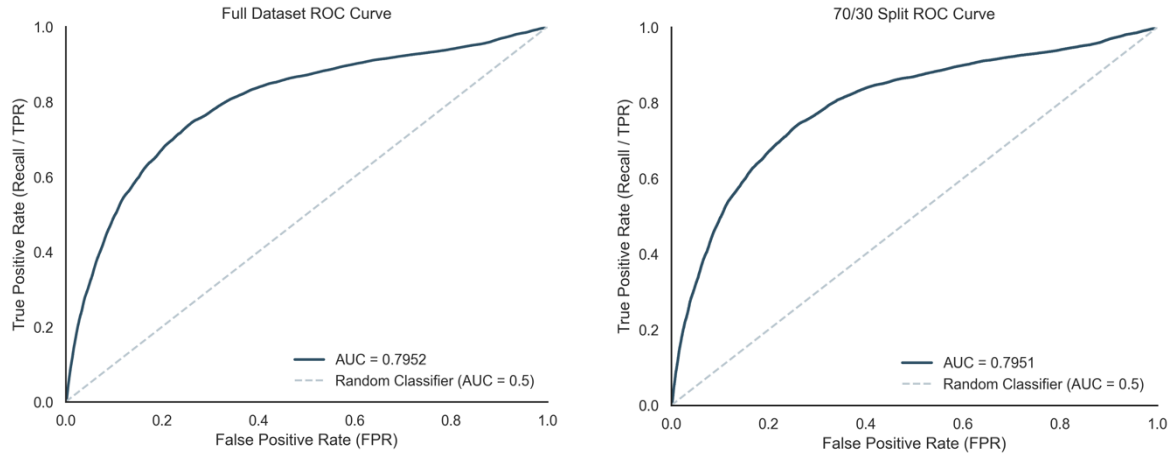
The choice of k reflects a bias-variance trade off : larger k typically reduces bias at the expense of greater variance. We used k = 5 and k = 10, both common choices which typically yield stable performance without excessive bias or variance.

	Threshold = 0.2		Accuracy	Precision	TPR (Recall)	F Score	ROC AUC	MSE
K-Fold Cross Validation	K = 5	*Mean*	0.6829	0.3985	0.8133	0.5349	0.7943	0.1372
		St. Dev	0.0012	0.0012	0.0021	0.0015	0.0022	0.0004
	K = 10	*Mean*	0.6829	0.3985	0.8132	0.5349	0.7943	0.1372
		St. Dev	0.0024	0.0022	0.0022	0.0022	0.0024	0.0005

Results aligned with earlier findings: the 20% threshold delivered strong recall and a balanced sensitivity-precision trade-off, confirming robustness across data splits.

ROC and AUC Analysis

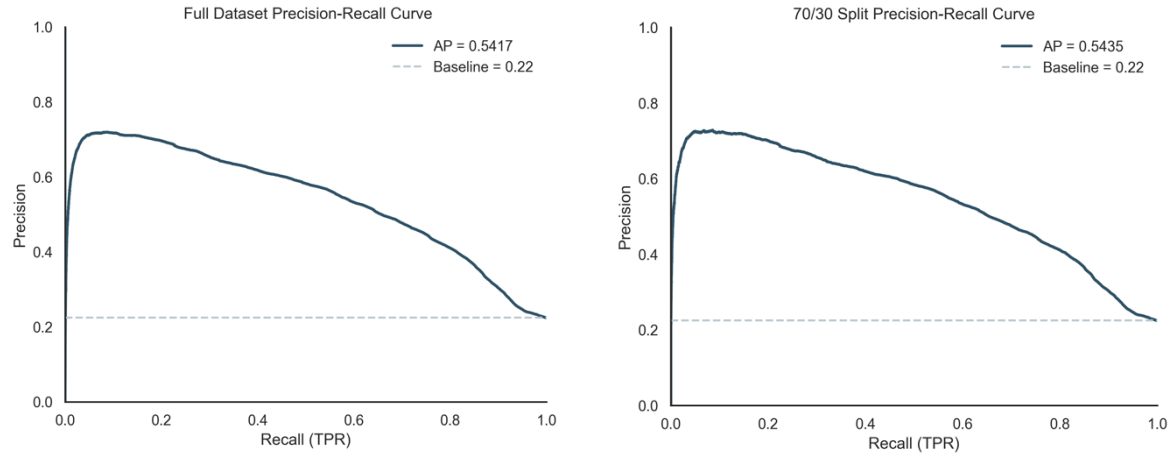
The ROC curve plots TPR against FPR for every conceivable probability threshold. AUC measures a classifier's ability to distinguish between classes, with values between 0 to 1. A higher AUC indicates greater predictive capacity, while a value of 0.5 implies no better than random guessing. Ideally, the ROC curve hugs the top-left corner, reflecting high TPR and low FPR.



ROC curves for both the full dataset and the 70:30 split show strong discriminatory power, with AUCs of 0.7952 and 0.7951 respectively. However, AUC can overstate performance in imbalanced settings where identifying the minority class is key.

P-R Curve

To address this, we also examined the Precision-Recall (PR) curve, which focuses on the positive class. An ideal PR curve hugs the top-right corner, combining high precision with high recall.



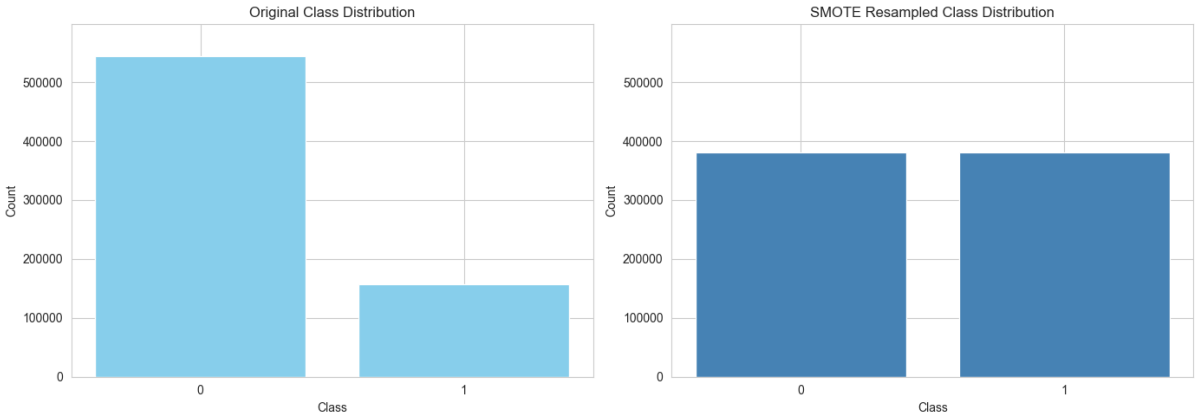
Both PR curves show average precision of around 0.54, suggesting the model strikes a reasonable balance between capturing true defaults and limiting false positives, despite class imbalance.

Question 3

(a)

Throughout our analysis, we faced a trade-off between reducing false negatives (FNR) or false positives (FPR) – in essence: should the bank prioritise avoiding defaults or enabling more lending? This would be straightforward were it not for our context. Given the SBA absorbs part of the default risk, some banks may lower safeguards, knowing they don't bear the full cost. We aimed to minimise both metrics, but gave preference to reducing FNR, as defaults likely carry greater monetary and regulatory costs.

We balanced data during model tuning and training, but tested on the original imbalanced set. This approach was chosen due to the commercially sensitive nature of the minority class (Defaulters). Balancing during training exposed models to more defaults (the minority class), improving their ability to detect defaults when evaluated on real-world, imbalanced test data.

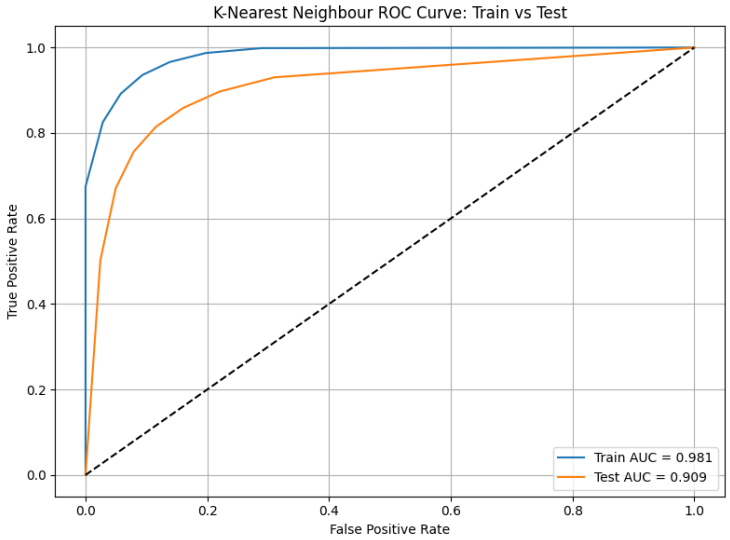


We used RandomSearchCV with 5-fold cross-validation folds to tune hyperparameters for our models as its increased computational efficiency justified the marginal trade-off in accuracy compared to GridSearch.

K-Nearest Neighbours Classification

K-Nearest Neighbours (KNN) classifies a datapoint based on the majority class among its K closest neighbours in feature space.

We used AUC for scoring during hyperparameter tuning to prevent overfitting (e.g., K=1), which typically yields high recall but poor generalisation. Since KNN lacks continuous probability output and is sensitive to local noise, AUC provides a more reliable guide.



Actual Loan Status

Predicted Loan Status	K=1	Default	No Default
	Default	34,283	15,376
	No Default	12,883	147,803

Actual Loan Status

Predicted Loan Status	K=3	Default	No Default
	Default	37,140	17,131
	No Default	10,026	146,048

Actual Loan Status

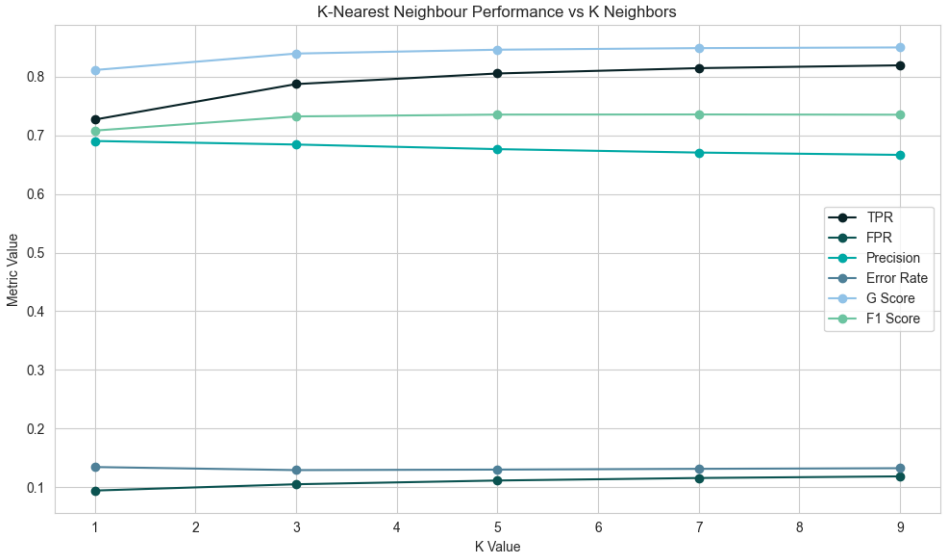
Predicted Loan Status	K=5	Default	No Default
	Default	37,993	18,169
	No Default	9,173	145,010

Actual Loan Status

Predicted Loan Status	K=7	Default	No Default
	Default	38,423	18,871
	No Default	8,743	144,308

Actual Loan Status

K=9	Default	No Default
Predicted Loan Status: Default	38,648	19,320
Predicted Loan Status: No Default	8,518	143,859



Model	K-Neighbours	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score	Error Rate	AUC of ROC
K-Nearest Neighbour	1	0.7269	0.0942	0.2731	0.9058	0.8657	0.6904	0.7081	0.8113	0.1343	0.8160
	3	0.7874	0.1050	0.2126	0.8950	0.8710	0.6843	0.7322	0.8395	0.1291	0.8820
	5	0.8055	0.1113	0.1945	0.8887	0.8700	0.6765	0.7354	0.8461	0.1300	0.9000
	7	0.8146	0.1156	0.1854	0.8844	0.8687	0.6706	0.7357	0.8488	0.1313	0.9090
	9	0.8194	0.1184	0.1806	0.8816	0.8677	0.6667	0.7352	0.8499	0.1323	0.9140

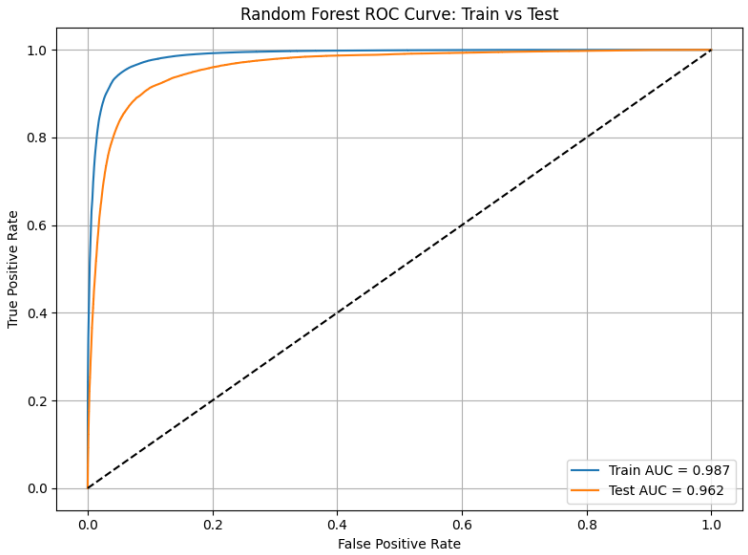
We only considered K = [1, 3, 5, 7, 9] in our evaluation, excluding even values to avoid the possibility of ties between classes. Model performance improves as K increases, showing rising recall and falling FNR while still maintaining reasonable FPR levels. We chose K=7 as our optimal K value due to its strong performance compared to other K, with the highest F-Score across the models. Overall, despite a strong test AUC of 0.909, our model reports a quite unfavourable FNR metric with 18.54% of loan defaults not accurately detected by the K=7 model which would prove costly to the bank.

Model	K Number of Neighbours	TPR (Recall)	FPR	FNR	Precision	F Score	Error Rate
KNN 5-Fold Cross Validation*	1	0.7232	0.0961	0.2768	0.6851	0.7036	0.1366
	3	0.7838	0.1086	0.2162	0.6760	0.7259	0.1327
	5	0.8020	0.1163	0.1980	0.6660	0.7277	0.1346
	7	0.8095	0.1203	0.1905	0.6605	0.7274	0.1360
	9	0.8131	0.1232	0.1869	0.6561	0.7262	0.1375

* All reported scores are the mean scores of the validation folds

Random Forest Classification

Random Forest is an ensemble method that builds multiple decision trees on bootstrapped samples, using random subsets of predictors. This reduces correlation between trees and improves predictive accuracy, particularly in high-dimensional settings. It is also valued for its robustness and ease of interpreting variable importance.



Actual Loan Status

10%	Default	No Default
Predicted Loan Status: Default	45,474	35,331
Predicted Loan Status: No Default	1,692	127,848

Actual Loan Status

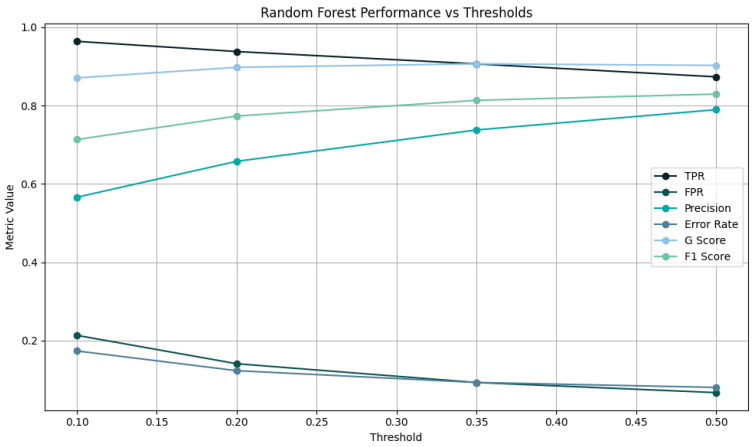
20%	Default	No Default
Predicted Loan Status: Default	44,244	23,435
Predicted Loan Status: No Default	2,922	139,744

Actual Loan Status

35%	Default	No Default
Predicted Loan Status: Default	42,721	15,153
Predicted Loan Status: No Default	4,445	148,026

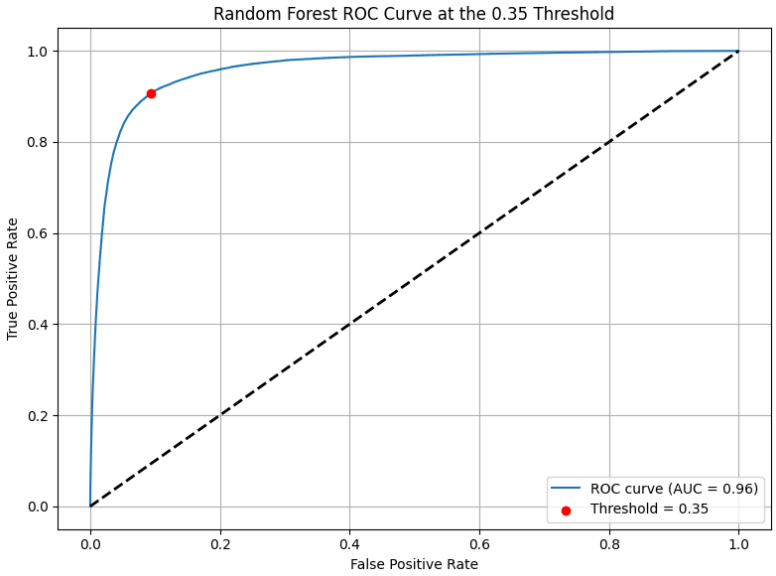
Actual Loan Status

50%	Default	No Default
Predicted Loan Status: Default	41,163	10,880
Predicted Loan Status: No Default	6,003	152,299



Model	Probability Threshold	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Random Forest	50%	0.8734	0.0673	0.1266	0.9327	0.9194	0.7896	0.8294	0.9026
	35%	0.9065	0.0931	0.0935	0.9069	0.9068	0.7378	0.8135	0.9067
	20%	0.9380	0.1411	0.0620	0.8589	0.8767	0.6578	0.7733	0.8976
	10%	0.9640	0.2135	0.0360	0.7865	0.8263	0.5662	0.7134	0.8707

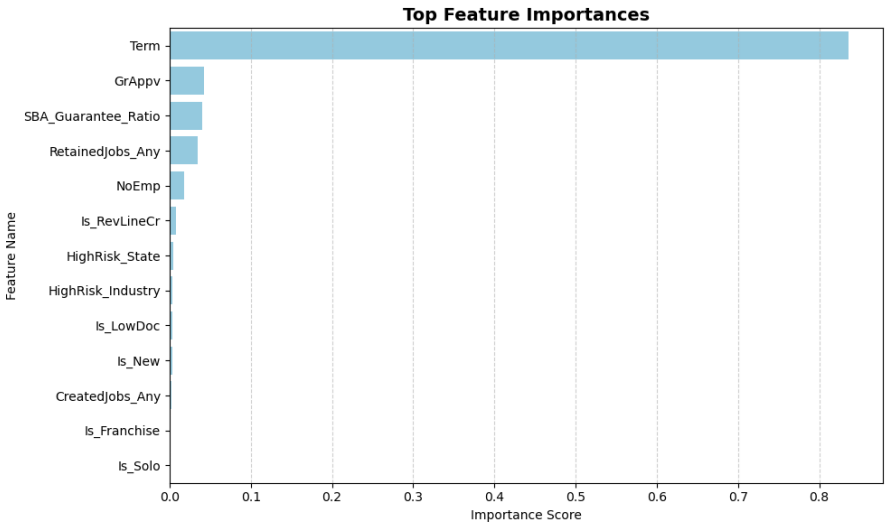
The Random Forest model performed well across thresholds. Consistent with our focus on minimising FNR while maintaining acceptable FPR, the 35% threshold offered the best balance. A test AUC of 0.962 from the ROC curve confirms the model's strong discriminatory power between defaulting and non-defaulting loans.



Model	Probability Threshold	TPR (Recall)	FPR	FNR	Precision	F Score	Error Rate
Random Forest 5-Fold Cross Validation*	50%	0.8709	0.0662	0.1291	0.7917	0.8294	0.0803
	35%	0.9034	0.0916	0.0966	0.7403	0.8137	0.0928
	20%	0.9358	0.1407	0.0642	0.6578	0.7725	0.1236
	10%	0.9609	0.2142	0.0391	0.5646	0.7113	0.1749

* All reported scores are the mean scores of the validation folds

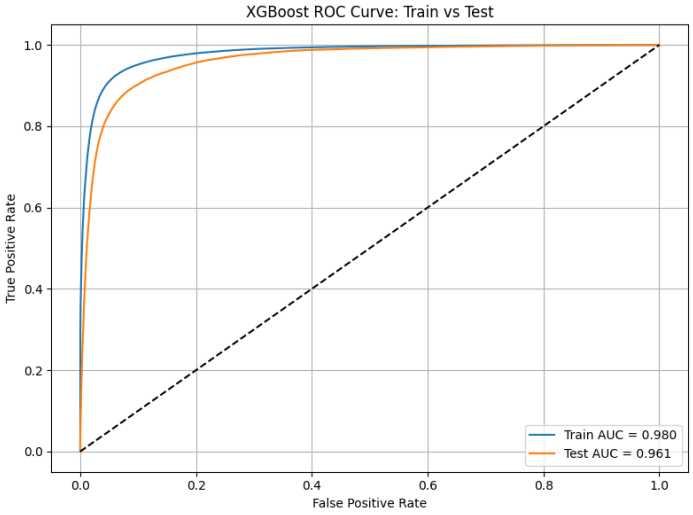
The above cross-validation results further support our selection.



"Term" is by far the most important feature for predicting loan defaults, which makes sense, as short-term SBA loans, often given to newer businesses with less collateral and tighter repayment timelines, tend to carry higher default risk than longer-term, asset-backed loans.

XGBoost

XGBoost is a gradient boosting algorithm that builds decision trees sequentially, correcting prior errors to improve accuracy and reduce bias. It performs well on tabular data, handles complexity, and is robust to overfitting. It often outperforms other models and provides interpretable feature importance measures.



Actual Loan Status

10%	Default	No Default
Predicted Loan Status: Default	45,863	42,811
Predicted Loan Status: No Default	1,303	120,368

Actual Loan Status

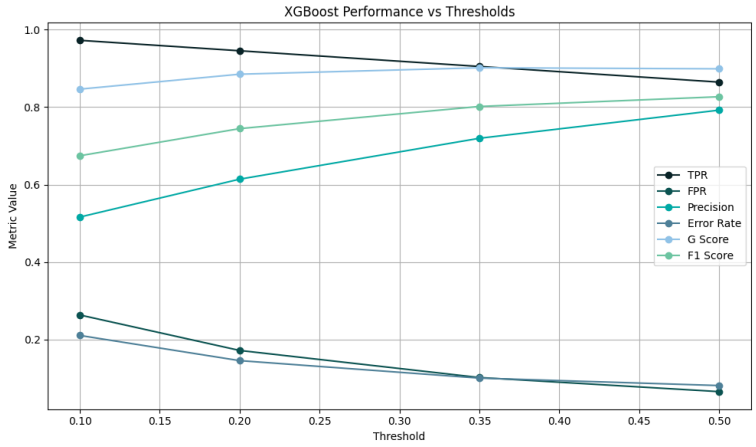
20%	Default	No Default
Predicted Loan Status: Default	44,590	28,007
Predicted Loan Status: No Default	2,576	135,172

Actual Loan Status

35%	Default	No Default
Predicted Loan Status: Default	42,719	16,677
Predicted Loan Status: No Default	4,447	146,502

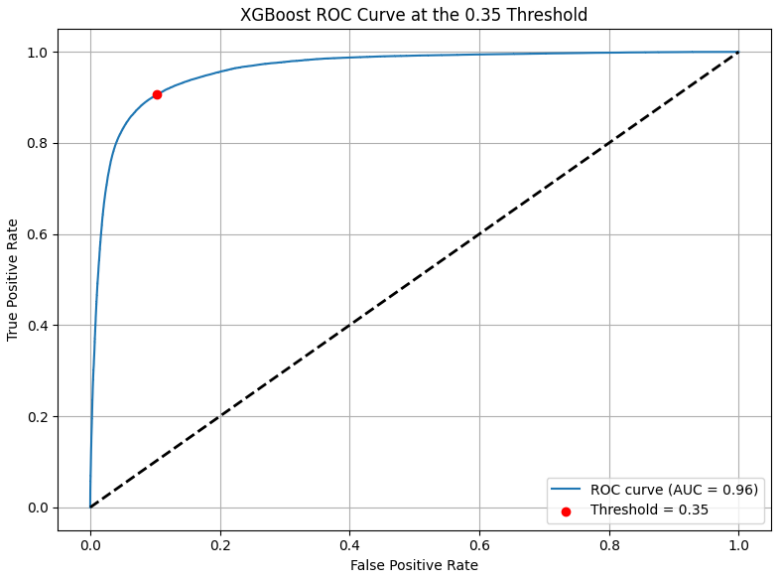
Actual Loan Status

50%	Default	No Default
Predicted Loan Status: Default	40,693	10,755
Predicted Loan Status: No Default	6,473	155,424



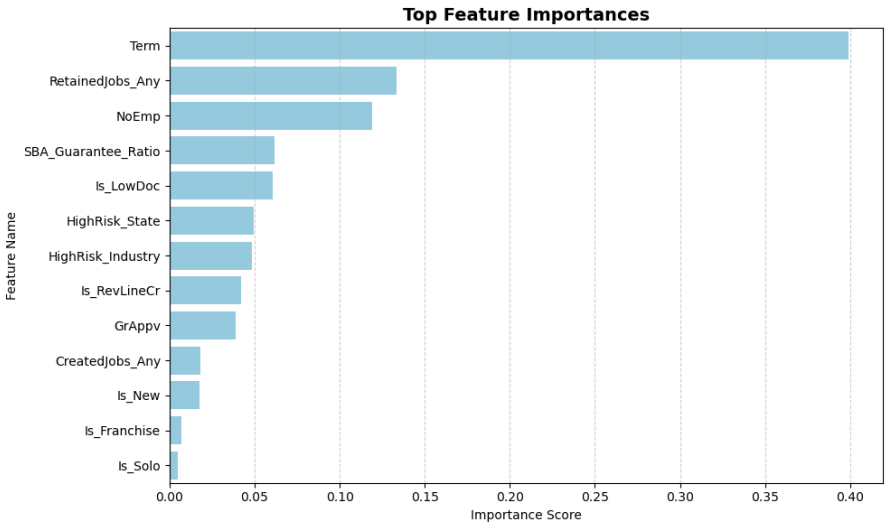
Model	Probability Threshold	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
XGBoost	50%	0.8644	0.0656	0.1356	0.9344	0.9187	0.7920	0.8266	0.8987
	35%	0.9047	0.1020	0.0953	0.8980	0.8995	0.7194	0.8015	0.9014
	20%	0.9452	0.1718	0.0548	0.8282	0.8544	0.6139	0.7440	0.8848
	10%	0.9722	0.2634	0.0278	0.7366	0.7894	0.5162	0.6743	0.8462

The XGBoost model performed best at the 35% threshold, which best captures our preferred trade-off outlined at the beginning of Q3. The ROC curve further supports the model's strong performance, with an AUC of 0.961 on the test set indicating excellent discriminatory ability between classes.



Model	Probability Threshold	TPR (Recall)	FPR	FNR	Precision	F Score	Error Rate
XGBoost 5-Fold Cross Validation*	50%	0.8637	0.0649	0.1363	0.7937	0.8272	0.0809
	35%	0.9043	0.0996	0.0957	0.7242	0.8043	0.0987
	20%	0.9449	0.1662	0.0551	0.6217	0.7499	0.1413
	10%	0.9703	0.2546	0.0297	0.5243	0.6807	0.2041

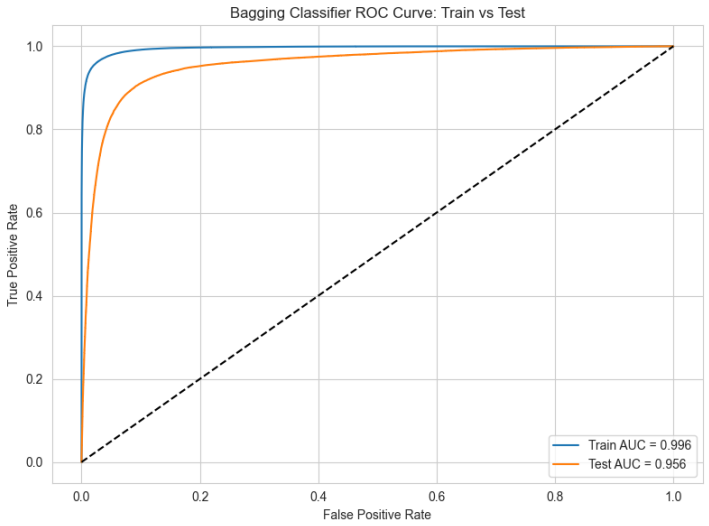
* All reported scores are the mean scores of the validation folds



As with Random Forest, the "Term" feature stands out with the highest importance score. "RetainedJobs_Any" also contributes meaningfully. Its importance may stem from the fact that indicating jobs will be retained as a result of the loan may signal underlying financial difficulty, suggesting the business is already in distress and therefore more likely to default. The "NoEmp" feature is also significant, as businesses with more employees tend to be more stable and reliable borrowers.

Bagging

Bagging is another ensemble method that builds multiple models in parallel using different bootstrapped subsets of the data. By averaging the predictions of these base models, it reduces variance and helps prevent overfitting. Bagging is especially effective when used with high-variance models and performs well in improving stability and accuracy on noisy datasets.



Actual Loan Status

10%	Default	No Default
Predicted Loan Status: Default	46,081	68,767
Predicted Loan Status: No Default	1,085	94,412

Actual Loan Status

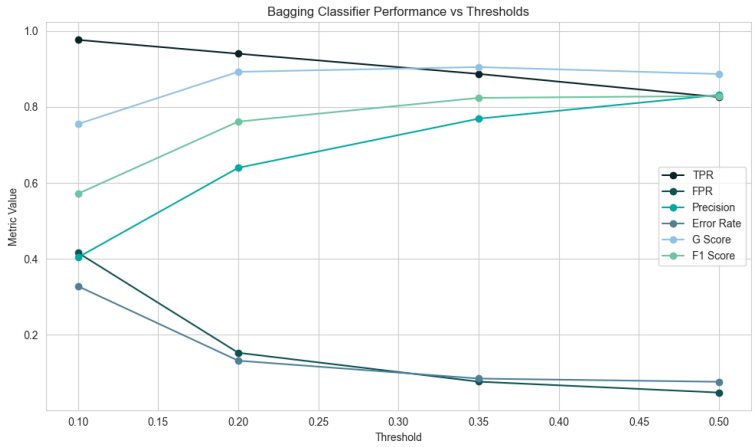
20%	Default	No Default
Predicted Loan Status: Default	44,392	25,392
Predicted Loan Status: No Default	2,774	137,787

Actual Loan Status

35%	Default	No Default
Predicted Loan Status: Default	41,946	12,810
Predicted Loan Status: No Default	5,220	150,369

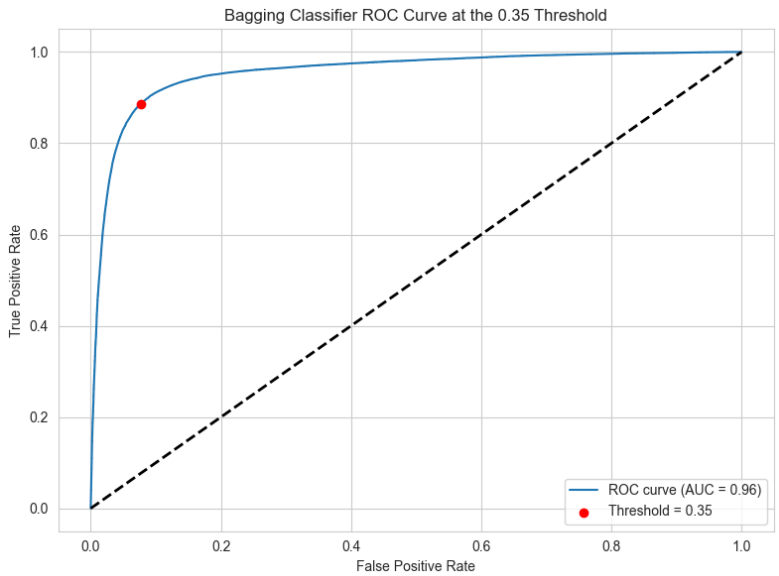
Actual Loan Status

50%	Default	No Default
Predicted Loan Status: Default	39,022	8,052
Predicted Loan Status: No Default	8,144	155,127



Model	Probability Threshold	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Bagging	50%	0.8256	0.0484	0.1744	0.9516	0.9233	0.8314	0.8285	0.8863
	35%	0.8867	0.0770	0.1133	0.9230	0.9148	0.7689	0.8236	0.9047
	20%	0.9398	0.1530	0.0602	0.8470	0.8678	0.6396	0.7612	0.8922
	10%	0.9762	0.4163	0.0238	0.5837	0.6717	0.4040	0.5715	0.7549

At the 20% threshold, Bagging offers a strong balance between FPR and FNR, with high recall and low FNR, though FPR is higher than at 35% and 50%. Its AUC of 0.956 reflects solid performance, slightly below other ensemble models. The 35% threshold provides the best FPR/FNR balance, but the higher FNR is a drawback. Despite this, we report the 35% threshold in the summary table to facilitate consistent comparison with our other models.



Model	Probability Threshold	TPR (Recall)	FPR	FNR	Precision	F Score	Error Rate
Bagging 5-Fold Cross Validation*	50%	0.8265	0.0485	0.1735	0.8313	0.8289	0.0765
	35%	0.8877	0.0772	0.1123	0.7687	0.8239	0.0851
	20%	0.9403	0.1537	0.0597	0.6387	0.7607	0.1326
	10%	0.9756	0.4158	0.0244	0.4041	0.5715	0.3280

* All reported scores are the mean scores of the validation folds

Model Comparison

Model	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score	AUC of ROC
KNN	0.8146	0.1156	0.1854	0.8844	0.8687	0.6706	0.7357	0.8488	0.9090
Random Forest	**0.9065**	0.0931	**0.0935**	0.9069	0.9068	0.7378	0.8135	**0.9067**	**0.9620**
XGBoost	0.9047	0.1020	0.0953	0.8980	0.8995	0.7194	0.8015	0.9014	0.9610
Bagging	0.8867	**0.0770**	0.1133	**0.9230**	**0.9148**	**0.7689**	**0.8236**	0.9047	0.9560

Random Forest led in recall, G-score, AUC, and had the lowest FNR, showcasing strong overall balance. Bagging topped precision, accuracy, F-score, FPR, and TNR, demonstrating excellent classification despite slightly lower recall. XGBoost was competitive, with recall, FNR, and AUC close to Random Forest, but fell short in precision and F-score. KNN performed well in recall and accuracy but trailed the ensemble models overall.

Threshold = 0.35 / K=7	Model	TPR (Recall)	FPR	FNR	Precision	F Score	Error Rate
5-Fold Cross Validation* Model Summaries	KNN	0.8095	0.1203	0.1905	0.6605	0.7274	0.1360
	Random Forest	0.9034	0.0916	0.0966	0.7403	0.8137	0.0928
	XGBoost	**0.9043**	0.0996	**0.0957**	0.7242	0.8043	0.0987
	Bagging	0.8877	**0.0772**	0.1123	**0.7687**	**0.8239**	**0.0851**

* All reported scores are the mean scores of the validation folds

Our cross-validation results reinforce much of our previous findings.

(b)

Model Comparison

For the purpose of comparison, we evaluated each classification model using a fixed probability threshold of 20%**,** chosen based on the optimal performance observed in Logistic Regression from Section 2. The K=7 model is reported for KNN

Model	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score	AUC of ROC
Logistic Regression	0.8150	0.3541	0.1850	0.6459	0.6839	0.3995	0.5362	0.7256	0.7951
KNN	0.8146	**0.1156**	0.1854	**0.8844**	0.8687	**0.6706**	0.7357	0.8488	0.9090
Random Forest	0.9380	0.1411	0.0620	0.8589	**0.8767**	0.6578	**0.7733**	**0.8976**	**0.9620**
XGBoost	**0.9452**	0.1718	**0.0548**	0.8282	0.8544	0.6139	0.7440	0.8848	0.9610
Bagging	0.9398	0.1530	0.0602	0.8470	0.8678	0.6396	0.7612	0.8922	0.9560

.Results are mixed, with no single model excelling across all metrics. Logistic Regression underperforms overall, with high FPR and FNR and low precision. XGBoost achieves the highest recall and lowest FNR but at the cost of higher FPR and weaker precision. Bagging shows strong precision, accuracy, and balanced performance. KNN performs moderately, with the lowest FPR but lags behind ensemble models.

Random Forest stands out as the best fit for our use case, offering high recall, low FPR and FNR, and leading in F-score, G-score, and AUC.

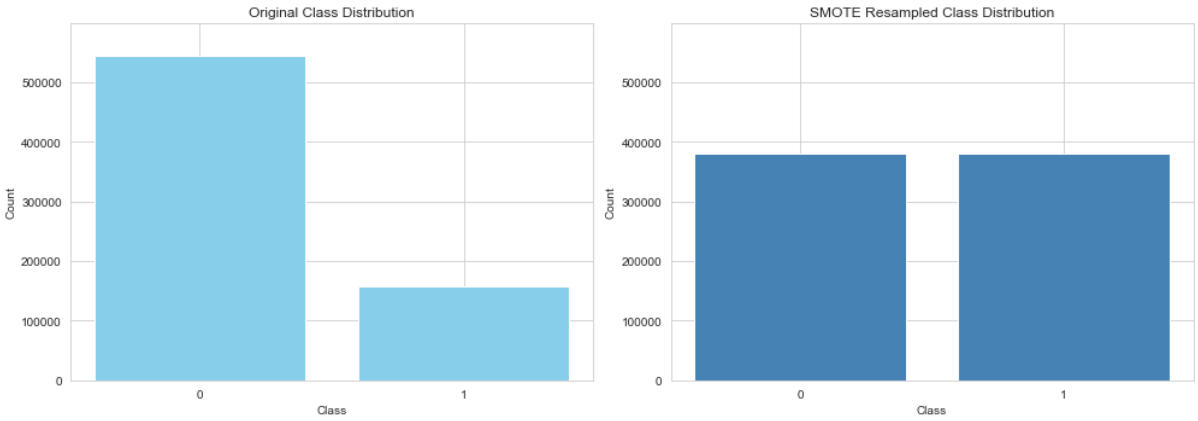
Question 4

Unsupervised learning techniques are well-suited for exploring complex, high-dimensional datasets, particularly when the goal is to uncover hidden patterns or structure without predefined labels. In this analysis, we apply three such methods: Principal Component Analysis (PCA), K-Means Clustering and Hierarchical Clustering.

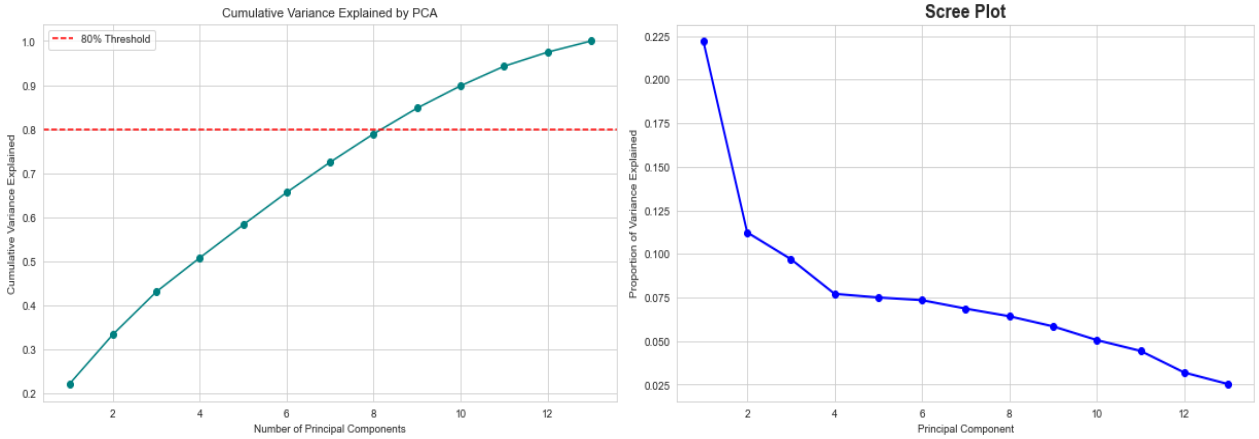
While these techniques are valuable for data exploration and dimensionality reduction, their limited interpretability poses challenges in applications like loan default prediction, where clear decision boundaries and transparency are essential.

PCA

PCA transforms high-dimensional data into a smaller set of uncorrelated principal components (PCs) by identifying and combining patterns in the original features. It preserves key variance while reducing dimensionality, simplifying analysis, and mitigating overfitting and multicollinearity. The cleaned dataset showed no evidence of multicollinearity between variables so using PCA for multicollinearity reduction purposes was no longer viable, so we focus on using it from a dimensionality reduction standpoint.

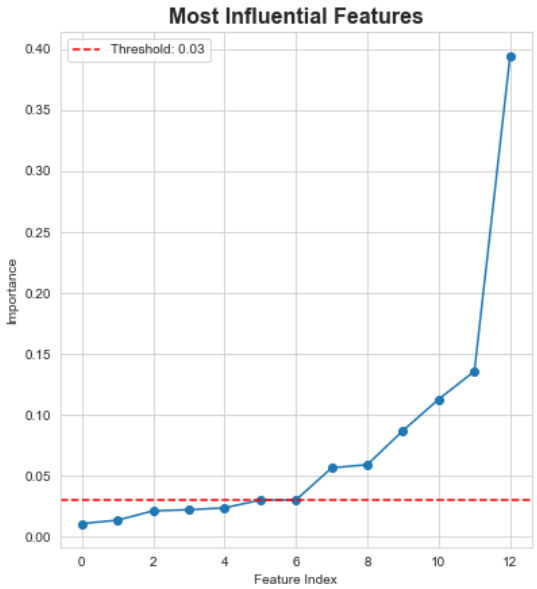


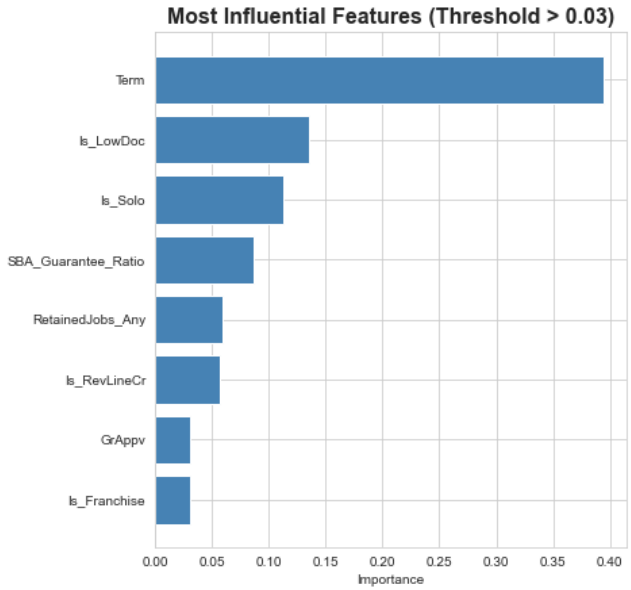
After rebalancing using SMOTE, PCA was applied. The graph shows that variance gain levels off after the first few components. 9 PCs capture about 80% of the variance, offering a good trade-off between feature retention and dimensionality reduction.



Feature Importance

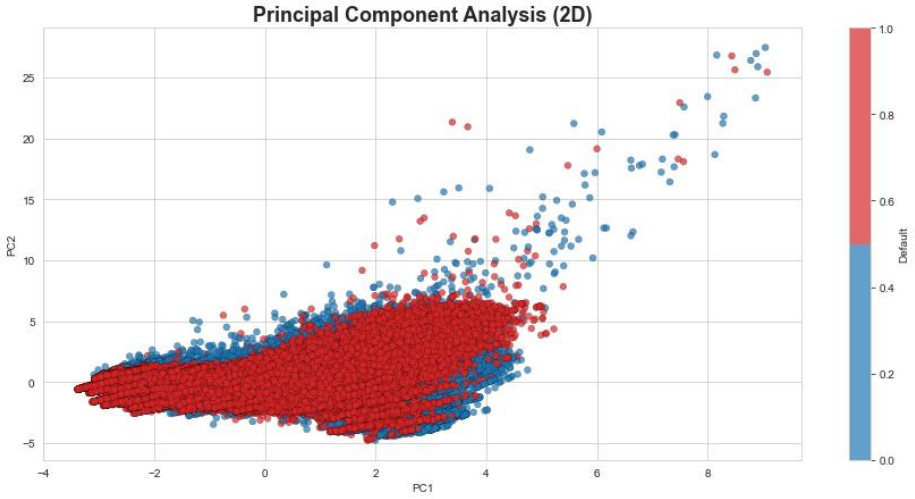
Shows Term, Is_LowDoc and Is_Solo to be the features with the highest relative loadings.





Index	Feature Name	Importance
1	Term	0.3947
2	Is_LowDoc	0.1358
3	Is_Solo	0.113
4	SBA_Guarantee_Ratio	0.0872
5	RetainedJobs_Any	0.0594
6	Is_RevLineCr	0.0567
7	GrAppv	0.0304
8	Is_Franchise	0.0303

The graph below, showing the first two PC's shows no clear separation between default and non-default groups.



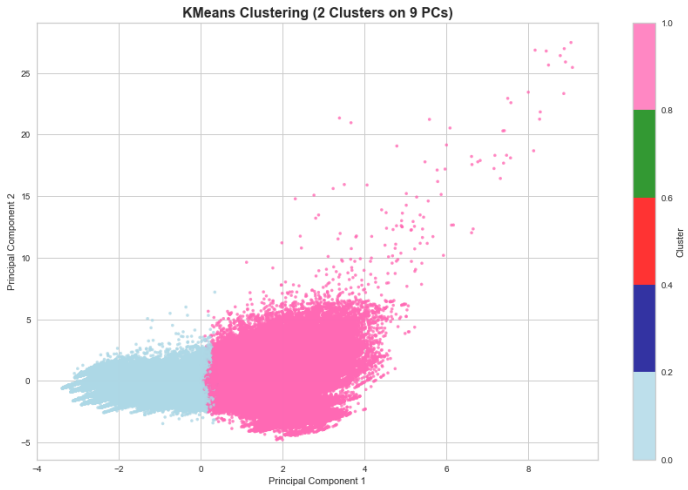
K-Means Clustering

Partitions the data into K distinct, non-overlapping clusters by minimising the within-cluster sum of squares (WCSS) - the average squared Euclidean distance between points and their cluster centroid. The method is

sensitive to both the choice of K and the initial centroid placement, making careful selection of K essential for meaningful interpretation.

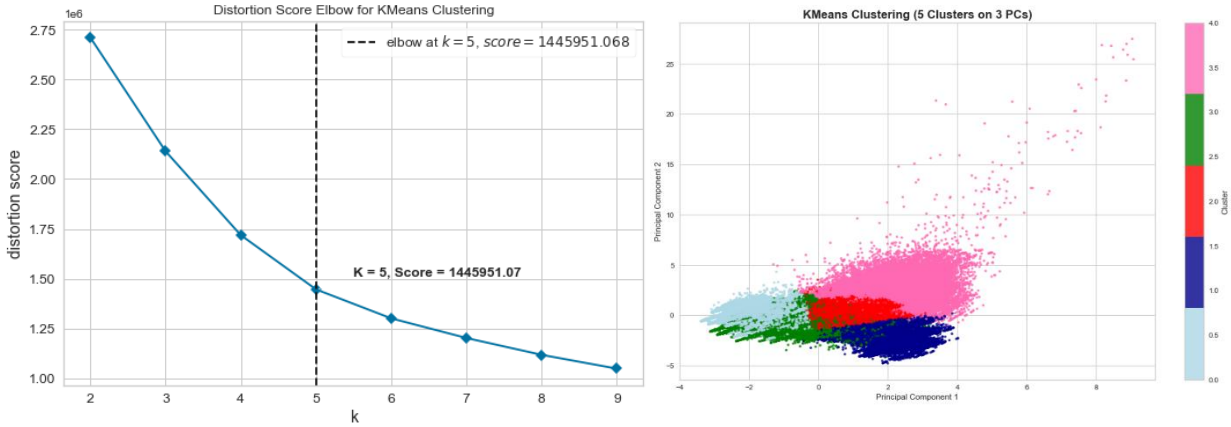
Elbow and silhouette techniques failed to establish optimal number of clusters for the 9 PC dataset, so we choose 2 clusters for simplicity. Performing K-means clustering on the 3 PC dataset, an elbow plot was obtained to visually observe the optimal number of clusters which was 5.

2 Clusters on 9 PCs:



Clustering on 9 principal components shows clearer binary separation with minimal overlap, suggesting the higher-dimensional space better preserves the data's structure for coherent clusters.

5 Clusters on 3 PCs:



Clustering into 5 groups on 3 PCs shows finer segmentation with compact, visible groupings. However, notable overlap suggests the reduced space may lack sufficient separation for clear boundaries.

We applied PCA and clustering to the supervised models, testing three setups: 9 PCs, 9 PCs with 2 clusters, and 3 PCs with 5 clusters (per elbow plot). Model performance was evaluated using a 20% threshold and 9 nearest neighbours for KNN.

Logistic Regression	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Full Dataset	0.815	0.3541	0.185	0.6459	0.6839	0.3995	0.5362	0.7256
9 PCs	0.9889	0.9579	0.0111	0.0421	0.5155	0.508	0.6712	0.204
9 PCs & 2 Clusters	0.9892	0.9585	0.0108	0.0415	0.5154	0.5079	0.6712	0.2027
3 PCs & 5 Clusters	0.9828	0.9234	0.0172	0.0766	0.5297	0.5156	0.6764	0.2745

Random Forest	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Full Dataset	0.938	0.1411	0.062	0.8589	0.8767	0.6578	0.7733	0.8976
9 PCs	0.9342	0.1361	0.0658	0.8639	0.8783	0.6398	0.7595	0.8984
9 PCs & 2 Clusters	0.9651	0.2992	0.0349	0.7008	0.833	0.7634	0.8525	0.8224
3 PCs & 5 Clusters	0.9697	0.3813	0.0303	0.6187	0.7942	0.7177	0.8249	0.7745

XG Boost Random Search	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Full Dataset	0.9452	0.1718	0.0548	0.8282	0.8544	0.6139	0.744	0.8848
9 PCs	0.9693	0.5473	0.0307	0.4527	0.711	0.6391	0.7703	0.6624
9 PCs & 2 Clusters	0.9689	0.461	0.0311	0.539	0.7539	0.6776	0.7975	0.7226
3 PCs & 5 Clusters	1	1	0	0	0.5	0.5	0.6667	0

KNN	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Full Dataset	0.8194	0.1184	0.1806	0.8816	0.8677	0.6667	0.7352	0.8499
9 PCs	0.9587	0.3525	0.0413	0.6475	0.8031	0.7312	0.8296	0.7879
9 PCs & 2 Clusters	0.952	0.3248	0.048	0.6752	0.8136	0.7456	0.8362	0.8017
3 PCs & 5 Clusters	0.9557	0.3641	0.0443	0.6359	0.7958	0.7242	0.824	0.7796

** 9 nearest neghbours*

Bagging	TPR (Recall)	FPR	FNR	TNR	Accuracy	Precision	F Score	G Score
Full Dataset	0.9398	0.153	0.0602	0.847	0.8678	0.6396	0.7612	0.8922
9 PCs	0.9687	0.3151	0.0313	0.6849	0.8268	0.7545	0.8483	0.8145
9 PCs & 2 Clusters	0.9683	0.3163	0.0317	0.6837	0.826	0.7538	0.8477	0.8137
3 PCs & 5 Clusters	0.9563	0.3459	0.0437	0.6541	0.8052	0.7343	0.8308	0.7909

Results showed that applying PCA with clustering yielded competitive performance in several models, particularly improving recall and F-score, which are crucial in minimizing false negatives in default prediction. For instance, tree-based models such as Random Forest and Bagging maintained strong F-scores under PCA and clustering configurations.

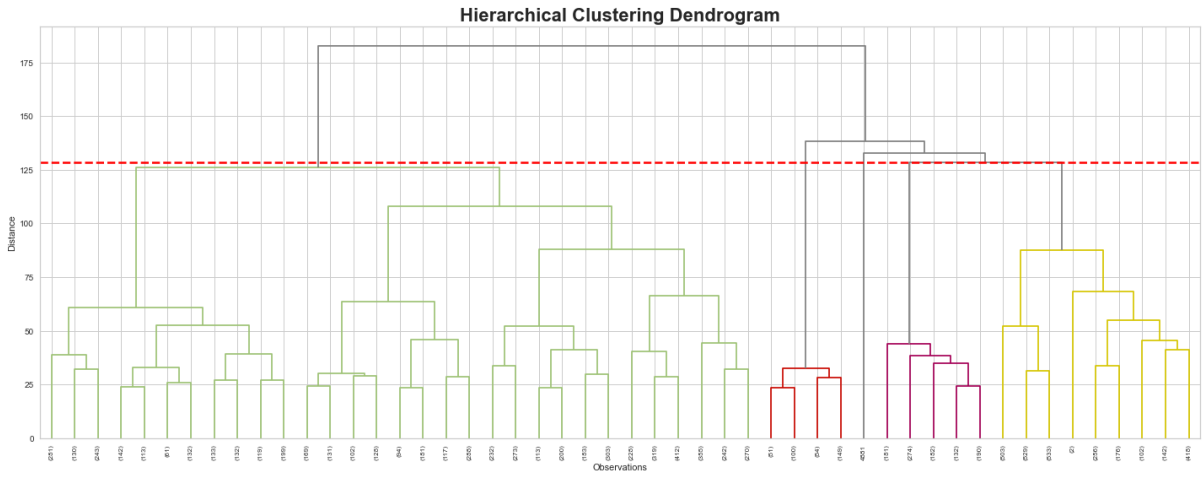
Using 3 PCs and 5 Clusters, excessive reduction, often resulted in performance degradation due to loss of key predictive information. In many cases, the addition of cluster labels provided only marginal improvements or even added noise, depending on the model's sensitivity to feature space changes.

Overall, while unsupervised techniques improved performance in certain models their effectiveness was inconsistent, and the full feature set remained superior in others.

Hierarchical Clustering

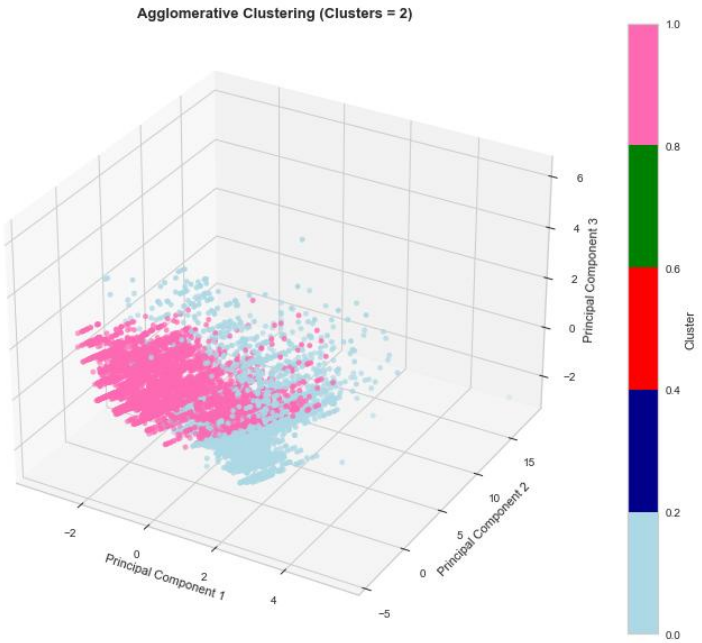
Another method of cluster analysis which seeks to build a hierarchy of clusters. The agglomerative approach is considered, whereby a dendrogram is created starting from the leaves and combining clusters up to the trunk.

Running hierarchical clustering on the 9 PC dataset we set 5 as the threshold number of clusters. Calinski-Harabasz Score and we determined the optimal number of clusters was 2.



K	Calinski-Harabasz Score
2	6982.5
3	6289.75
4	6841.41
5	6863.44
6	6043.79

We proceed to visualise the separating ability of the data using K = 2 clusters. Agglomerative clustering with 2 clusters reveals moderate separation along the first principal component, though some overlap remains between groups.



Number of Observations in Each Cluster	
Cluster	No. Observation
0	6025
1	3975

Conclusion on Unsupervised Learning

Unsupervised dimensionality reduction showed mixed results in improving supervised model performance compared to models trained on the original 70/30 split. While PCA with clustering yielded competitive results for models like Random Forest and Bagging, full-feature models often outperformed them in accuracy and balance. Using 3 PCs and 5 clusters frequently led to poorer performance, likely due to loss of key predictive information. Though dimensionality reduction can reduce noise and improve generalisation, its effectiveness appears model-dependent

It is also important to acknowledge the hesitation within financial institutions to adopt unsupervised learning approaches. This reluctance often stems from the limited interpretability of such models, which are frequently regarded as "black boxes" and pose challenges in terms of regulatory transparency and compliance.

Bibliography

United States Congress. (1974) *Equal Credit Opportunity Act*, Pub. L. No. 93–495, 88 Stat. 1521. Available at: https://www.govinfo.gov/content/pkg/USCODE-2011-title15/html/USCODE-2011-title15-chap41-subchapIV.htm (Accessed: 8 April 2025).

U.S. Small Business Administration (n.d.) *SBA Debt Relief*. Available at: https://www.sba.gov/funding-programs/loans/covid-19-relief-options/sba-debt-relief (Accessed: 12 April 2025).

U.S. Small Business Administration. (2017) *CDC Best Practices Guidance – Jobs Created and Retained Reporting*. Available at: https://www.sba.gov/document/support--cdc-best-practices-guidance-jobs-created-and-retained-reporting(Accessed: 11 April 2025).