Bank Churn Prediction Model

Matthew Hale

Apr-2024

1. Introduction

In retail banking, much of the analytics, modelling and machine learning effort is focused on decision-making prior to customer on-boarding - which customers to extend credit to and at what price, how to price savings/deposits, which applications might be fraudulent etc etc.

Another focus area is customer behaviour post onboarding, and a key aspect of that relates to the question of customer attrition/retention.

Banks often have a good deal of data on their customer base - both acquired at the time of onboarding and through the course of the customer relationship - and can use this data to predict which customers are likely to leave the bank and take action to retain them (if desirable for the bank to do so).

This project uses kaggle data to replicate a typical process within a retail bank to predict the likelihood of any given customer exiting the bank.

This would normally be combined with profit modelling to determine the expected value if the customer was retained to support the decision-making on the action to take (if any), but this project focuses solely on modelling the likelihood of customer exit, with as much accuracy as possible.

1b) Initial Datasets

A dataset from the kaggle website has been used for this project. The link to the initial dataset (train.csv) is here

This was downloaded and saved alongside code/pdf/Rmd filess in the github repo here, which can be cloned to run the code in this project.

The dataset contains ~165k rows and 14 columns. The columns, with the data type and some sample entries are listed below. 'Exited' is the target variable, a binary 1-0 variable indicating if the customer exited the bank or not.

Table 1: Variables in bank churn dataset

Variable	Type	Examples
id	int	115263 91135 92647 21522 76620 16019 29635 99399 65976 117964
CustomerId	int	15602089 15652214 15672443 15739123 15648032 15599084 15793897 15612494 15777892 15807065
Surname	chr	"Lucchesi" "Nwabugwu" "Onyekachukwu" "Okwudilichukwu" …
CreditScore	int	594 749 526 651 721 469 850 540 592 633
Geography	chr	"France" "Spain" "France" "France" …
Gender	chr	"Female" "Male" "Male" "Female" …
Age	num	27 36 23 35 33 33 20 49 37 49

Variable	Type	Examples
Tenure	int	8 5 5 9 1 1 8 9 8 1
Balance	num	0 0 103391 0 0 . . .
NumOfProducts	int	2 2 2 2 2 3 2 1 1 3
HasCrCard	num	1 1 1 1 1 1 1 1 1 1
IsActiveMember	num	1 0 1 1 1 0 1 1 0 0
EstimatedSalary	num	131190 71167 79919 104485 118025 . . .
Exited	int	0 0 0 0 0 0 0 0 0 1

1c) Processed/Cleansed Datasets

Initial processing/cleansing was done to randomly split the dataset into a training set called **df_train** (80% of the data) and a test set called **df_test** (20% of the data). The training set was then to be used to build predictive models using a range of Machine Learning (ML) techniques, and once the best model was selected, the test set as a final set to evaluate the model's performance on previously unseen data.

Please see BankChurnProject.R for the cleansing/processing code.

The row counts for the two sets are below, showing the ~80/20 split.

```
Number of rows in training set df_train = 132,027
```

```
Number of rows in test set df_test = 33,007
```

During the course of the exploratory data analysis further processing was done on the data to aid analysis and modelling. This included:

- Banding the continuous variables: Age, Balance, Credit Score, salary
- Rebanding the categorical variable NumOfProducts due to the small volume of data in some categories
- Changing the variable types as required by each Machine Learning technique (e.g. some algorithms required character variables to be input as factors)

1d) Modelling Methodology

After initial exploratory data analysis on the df_train set, a number of Machine Learning algorithms were trained and tuned using that dataset to determine which would most probably give the best predictions of Exit likelihood. Primarily the training framework in the caret package was used, with the following methods tried:

- A baseline approach of using randomly generated probability, classified using using the average likelihood of exit across the training set
- Simple and multivariate logistic regression models (with variable selection developed using a self written forward selection algorithm based on the BIC metric)
- Decision trees
- More complex tree based methods including random forest and gradient boosting (incl XGBoost and CatBoost)
- Support Vector Machine (SVM)

- Neural Network

These models were tuned in the main using using k-fold cross validation in the caret package to select the best hyperparameters. K-fold validation involves splitting the training set into k equal parts, training the model on k-1 of these parts and validating on the remaining part, and then repeating this process k times.

The tuned models were then put through a separate k-fold function to produce a consistent set of performance metrics from multiple ML different algos averaged across the 5 validation sets.

These metrics included the following confusion matrix derived figures (therefore requiring a predicted classification for each customer - i.e. exited vs not exited - and therefore the selection of a probability threshold for some ML techniques which outputted predicted probabilities of exit):

- **Accuracy**: (True Positives + True Negatives) / All
  [What % of all predictions, be they exited or not, were correct]
- **Precision**: True Positives / (True Positives + False Positives)
  [Out of all predicted exits, what % were actually exits]
- **Recall** (aka Sensitivity): True Positives / (True Positives + False Negatives)
  [Out of all actual exits, what % did the model predict as exits]
- **F1**: 2 * (Precision * Recall) / (Precision + Recall)
  [Balanced measure of Precision and Recall]
- **Specificity**: True Negatives / (True Negatives + False Positives)
  [Out of all actual non-exits, what % did the model predict as non-exits]
- **Balanced Accuracy**: (Sensitivity + Specificity) / 2
  [A balanced measure of Sensitivity and Specificity]

Another two measures were also included, given they didn't require a probability threshold to be selected.

- **Area under the ROC curve (AUC)**; the ROC curve plots the True Positive Rate (Recall) against the False Positive Rate across all possible probability thresholds, and the area under this plotted curve provides a measure of how well the model separates exiting vs non exiting customers (better discriminating models have the ROC curve pulled up towards the top left corner, with higher True Positive Rates and lower False Positive Rates, giving a larger AUC)
- **Log Loss**. This is a measure of the dissimilarity between the predicted probabilities from an ML model and the actual outcomes. In a similar fashion to RMSE, MAE etc (measures which are less appropriate for binary prediction problems) lower values suggest a closer match between the predictions and actuals and a better model.

The metrics for the validation sets were used to select the best model (the validation sets being used as a good view on expected performance of the models on previously unseen data down the line - i.e. the test data).

This model, once chosen, was then applied to the df_test dataset, to give the final predictions and to assess these predictions vs the actual outcomes in the test set.

2. Methods/Analysis

The exploratory data analysis conducted is documented below. This primarily involved reviewing the distributions of the variables in the dataset and how those variables related to the target variable (Exited).

2.1 Missing Variables

The first step was a quick check to understand if there were missing data, and if so to determine how to deal with these cases.

Per below, the dataset has no missing data at all (which highlights the simulated nature of the data - this would be an extremely surprising result in a real-world dataset).

Table 2: Missing data count by variable

	MissingCount
id	0
CustomerId	0
Surname	0
CreditScore	0
Geography	0
Gender	0
Age	0
Tenure	0
Balance	0
NumOfProducts	0
HasCrCard	0
IsActiveMember	0
EstimatedSalary	0
Exited	0

2.2 Initial consideration of ID variables

At this stage, before getting into the distributions, I wanted to understand whether all the variables were candidate predictors - in particular, whether the Customer ID variable in the dataset was likely to be useful in predicting the target variable.

If there were multiple records for the same customer ID in the dataset, then the variable would potentially be useful (e.g. if a customer had repeatedly chosen not to exit, then you might expect that to continue in the future).

Per below, however, whilst there are multiple records present for some customer IDs, it became clear that a given customer ID with multiple records didn't have consistent surnames throughout. i.e. the customer ID doesn't appear to be a unique customer identifier. As such, I took the decision to drop the ID variables as candidate predictors.

Here are the top customer IDs in terms of records in the data:

Table 3: Customer IDs with most records in the data

CustomerId	n
15682355	102
15585835	80
15793331	75
15648067	73
15570194	72
15782530	71

Here is an example customer ID with multiple records, and it's clear the surname, gender, age, geography etc are not consistent. i.e. the customer Id is not a unique identifier of a given customer.

Table 4: Example Customer ID showing different surnames, genders, ages, geographies etc

id	CustomerId	Surname	CreditScore	Geography	Gender	Age
24385	15565796	Docherty	615	France	Male	39
63921	15565796	Toscani	577	France	Male	20
90009	15565796	Docherty	482	Germany	Male	48
111963	15565796	P'eng	683	Germany	Male	52
127874	15565796	Palerma	665	France	Male	36
143116	15565796	Y?an	704	France	Female	34

2.3 Distributions of Numerical Variables

Now, to get on with reviewing the variables in the training set, the following table shows the minimum, mean and maximum of the numerical variables.

	Min	Mean	Max
id	0.00	82638.20	165033.0
CustomerId	15565701.00	15692047.52	15815690.0
CreditScore	350.00	656.44	850.0
Age	18.00	38.13	92.0
Tenure	0.00	5.03	10.0
Balance	0.00	55537.47	250898.1
NumOfProducts	1.00	1.55	4.0
HasCrCard	0.00	0.75	1.0
IsActiveMember	0.00	0.50	1.0
EstimatedSalary	11.58	112602.09	199992.5
Exited	0.00	0.21	1.0

Some of these, whilst numerical, are effectively categorical or binary (e.g. Tenure, NumOfProducts, HasCrCard, IsActiveMember), and of course Exited is the target variable.

Of the remaining numerical predictor variables, the following histograms show their distributions:

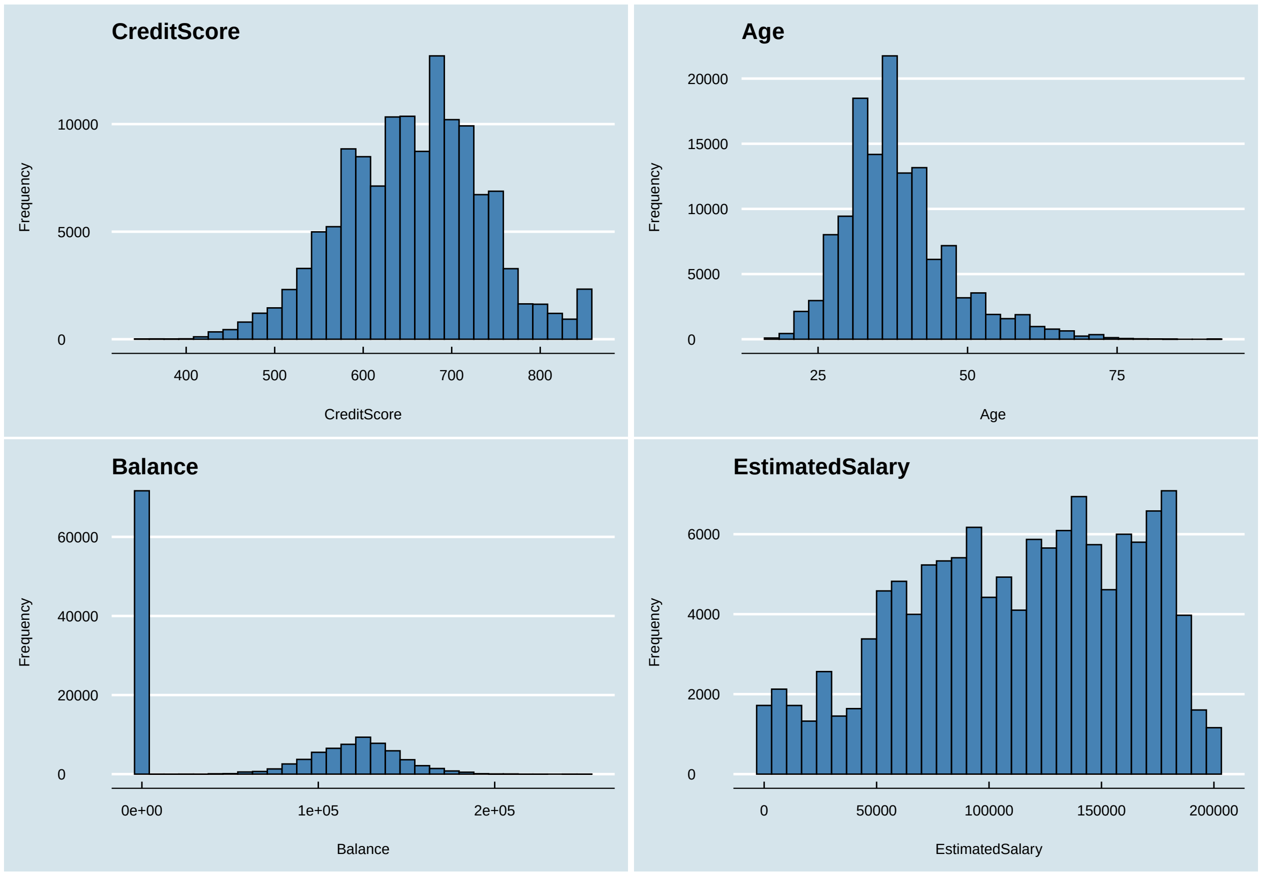


Figure 1: Histograms of Numerical Variables

So, a sensible looking credit score and age range, rather inflated estimated salaries (must be a US rather than UK dataset!), and a balance variable with a large number of 0s (which may be a concern - it's possible that these are missing values, or that the bank has a large number of customers with no balance).

2.4 Distributions of Categorical Variables and relationship to Exit Likelihood

The next step was to plot categorical variables to understand their distributions, and also overlay the average percentage of exits in each category to see if any patterns started to emerge.

In this step, I also plotted banded versions of the numerical variables, again for an early view on relationships between these variables and the target variable.

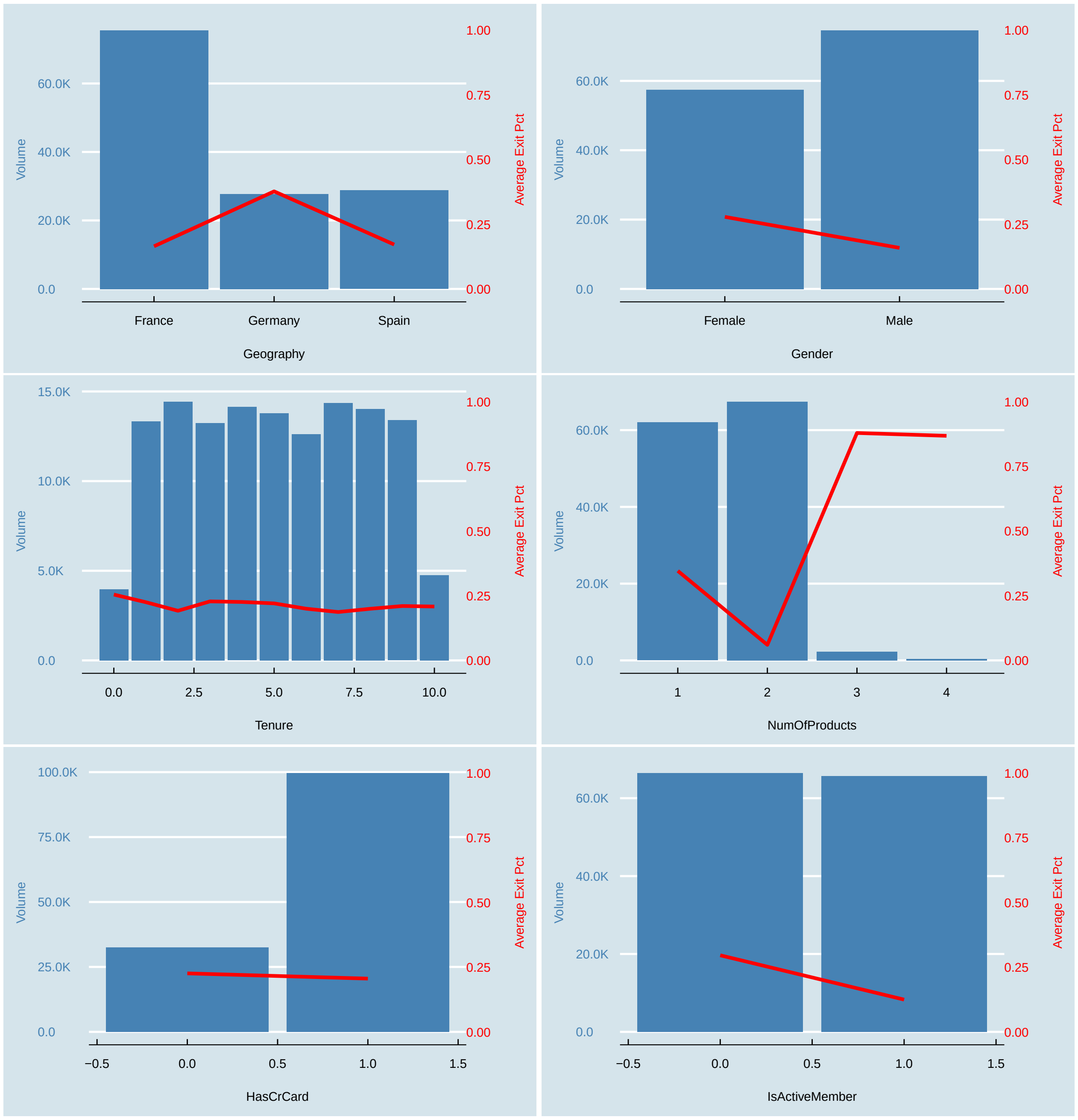


Figure 2: Categorical Vars vs Exit Pct

The charts show some interesting relationships between the categorical variables and the target variable. For example, the proportion of exits is higher in Germany and for female customers on average.

Other variables appear to have little impact on the attrition likelihood, such as tenure and whether the customer holds a credit card with the bank.

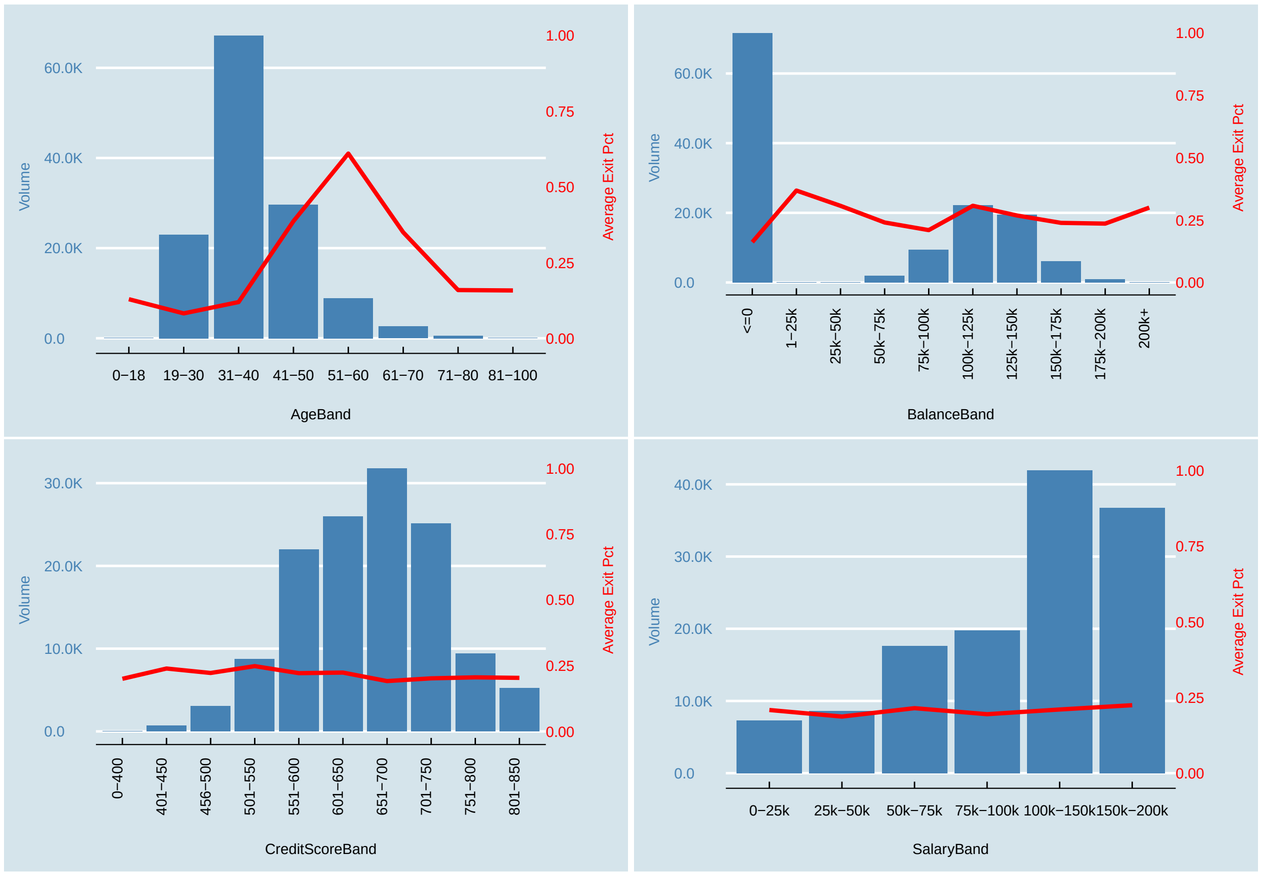
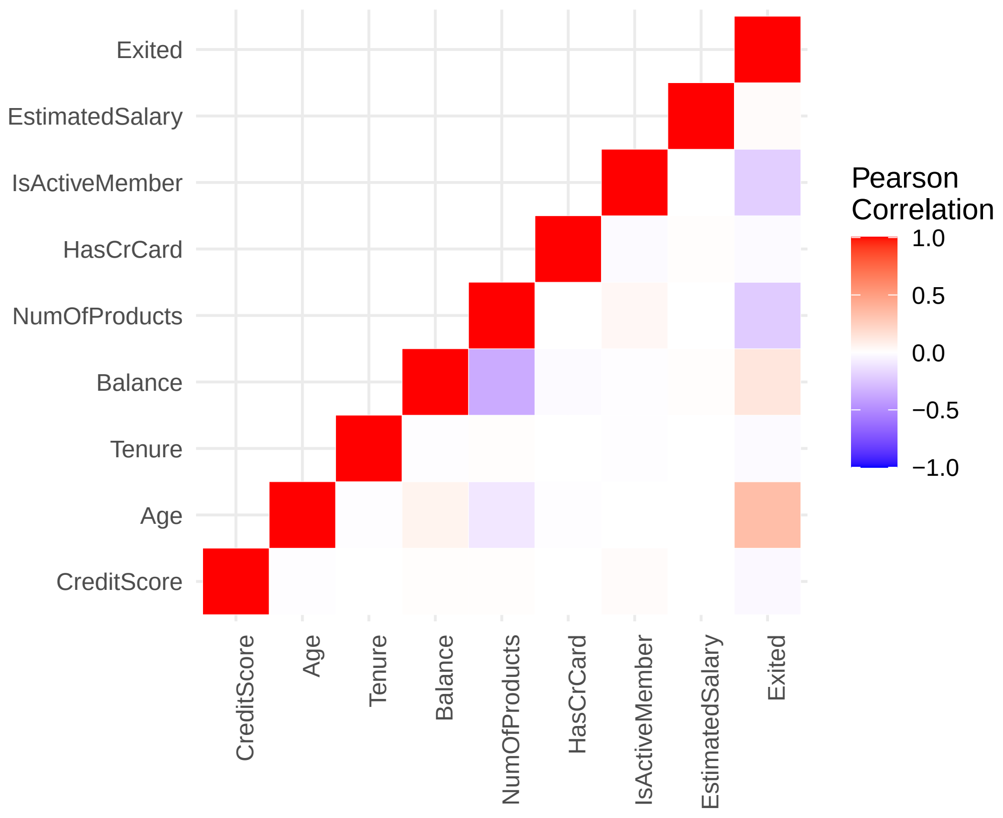


Figure 3: Banded Numerical Vars vs Exit Pct

Here we see age looking to have a key impact, with attrition likelihood increasing with age until around retirement age, and then dropping off. The other numerical variables appear to have relatively little impact, although the large proportion of customers on zero balance does appear to have a lower attrition likelihood on average.

2.5 Correlation Matrix

Next, a correlation matrix between the predictors and the target variable was produced to understand the relationships between the predictor variables (to ascertain if collinearity may be an issue for some ML algos) and between each predictor variable and the target variable.



As the plot shows, there appear to be several correlations with the target variable Exited, with Age, Balance, NumOfProducts and IsActiveMember all showing some correlation with the target variable.

Mostly the correlations between the predictor variables seem to be low, which is good news for the modelling process - it suggests that collinearity is not likely to be a significant issue. That said the correlations between Age, Balance and NumOfProducts are higher than the others, and this may be something to watch out for in the modelling process.

2.6 Information Values

To get a numerical representation of which of the variables may have the most impact on the target variable, Information Values were calculated for each variable.

This measures the differences in the distribution of the target variable across the different categories of each predictor variable (with numerical variables banded as part of the process), and is a good way to understand which variables may be most important in predicting the target variable.

Table 6: Information Values for each variable; higher values suggest more predictive power

	Variable	IV
4	Age	0.9149562
7	NumOfProducts	0.6359923
9	IsActiveMember	0.2785657
2	Geography	0.2327477
3	Gender	0.1270114

	Variable	IV
6	Balance	0.1103168
1	CreditScore	0.0118548
5	Tenure	0.0096067
10	EstimatedSalary	0.0058009
8	HasCrCard	0.0026945

So this supports the charting in suggesting age as a very key driver of exit likelihood, followed by NumOfProducts, IsActiveMember, Geography, Gender and Balance.

The other variables have very little impact on exit likelihood, which is a little surprising - features like credit score you would typically expect to be a driver of attrition, given high scoring customers are unlikely to be short of good offers to bank (and get credit products) with alternative providers, and also tend to be more price sensitive.

3. Modelling Methodology and Results

Having analysed the training dataset to improve understanding and to assess potential model drivers, the modelling process was undertaken to construct a prediction for each customer of their likelihood of leaving the bank.

As previously noted, a range of Machine Learning techniques were trained and tuned using the training dataset (tuning using k-fold validation), and the results for the best models selected based on the performance in the validation sets are provided in the results below.

3.1 Modelling Results

The metrics for each model in the validation sets are shown below. These are the average metrics across the 5 validation sets produced in the k-fold process, so should be a good indicator of expected performance in unseen test data.

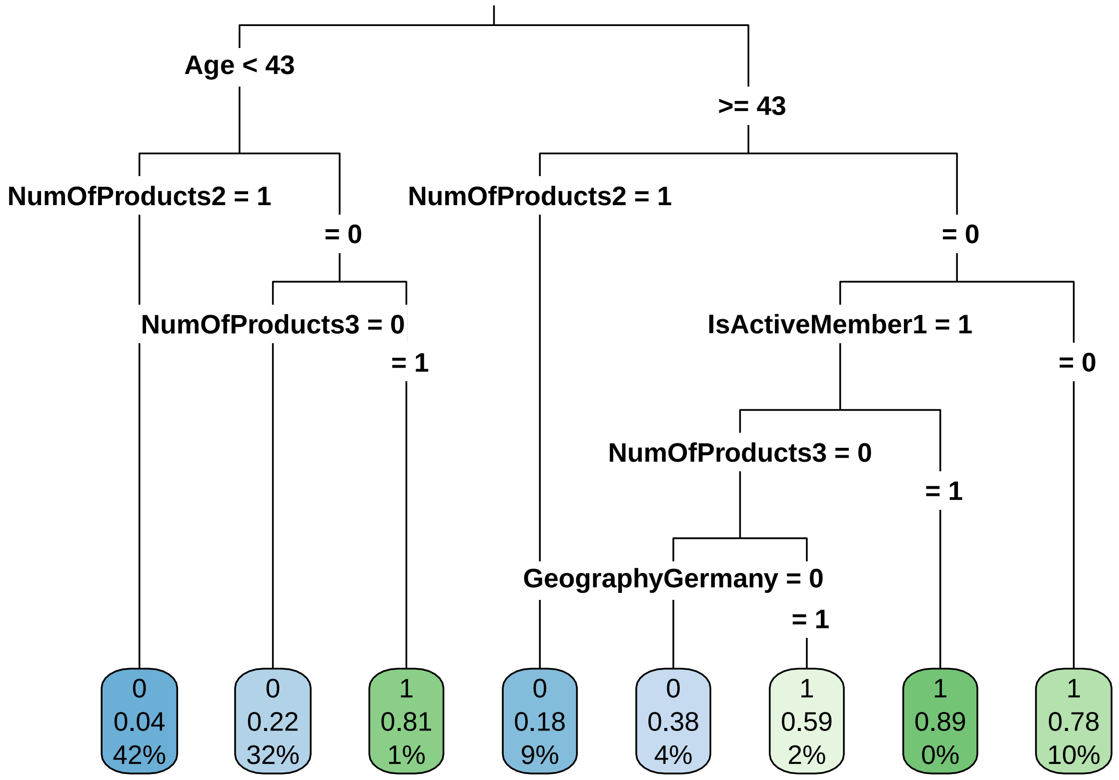
Table 7: Machine Learning Results Summary Table

ModelName	Accuracy	Precision	Recall	F1	Specificity	BalancedAccuracy	AUC	LogLoss
0_Base	0.668	0.212	0.212	0.212	0.790	0.501	0.498	0.996
1_LogReg_smpl	0.660	0.368	0.854	0.514	0.608	0.731	0.746	0.427
2_LogReg_cmplx	0.860	0.745	0.508	0.604	0.954	0.731	0.874	0.337
3a_Tree_simple	0.853	0.756	0.446	0.561	0.962	0.704	0.826	0.371
3b_Tree_complex	0.867	0.745	0.563	0.641	0.948	0.756	0.863	0.341
4_RF	0.865	0.747	0.545	0.630	0.951	0.748	0.878	0.356
5_SVM	0.864	0.788	0.462	0.582	0.968	0.715	0.807	0.367
6_GBM	0.865	0.754	0.531	0.623	0.954	0.742	0.887	0.324
7_XGB	0.866	0.751	0.543	0.630	0.952	0.748	0.888	0.320
8_CatBoost	0.866	0.750	0.545	0.632	0.951	0.748	0.889	0.320
9_NNet	0.865	0.748	0.546	0.631	0.951	0.748	0.888	0.322

Some notes on these results:

- The simple logistic regression referenced in the table is the best performing single variable logistic regression having experimented with all potential drivers of exit likelihood. The driver was number of products held.

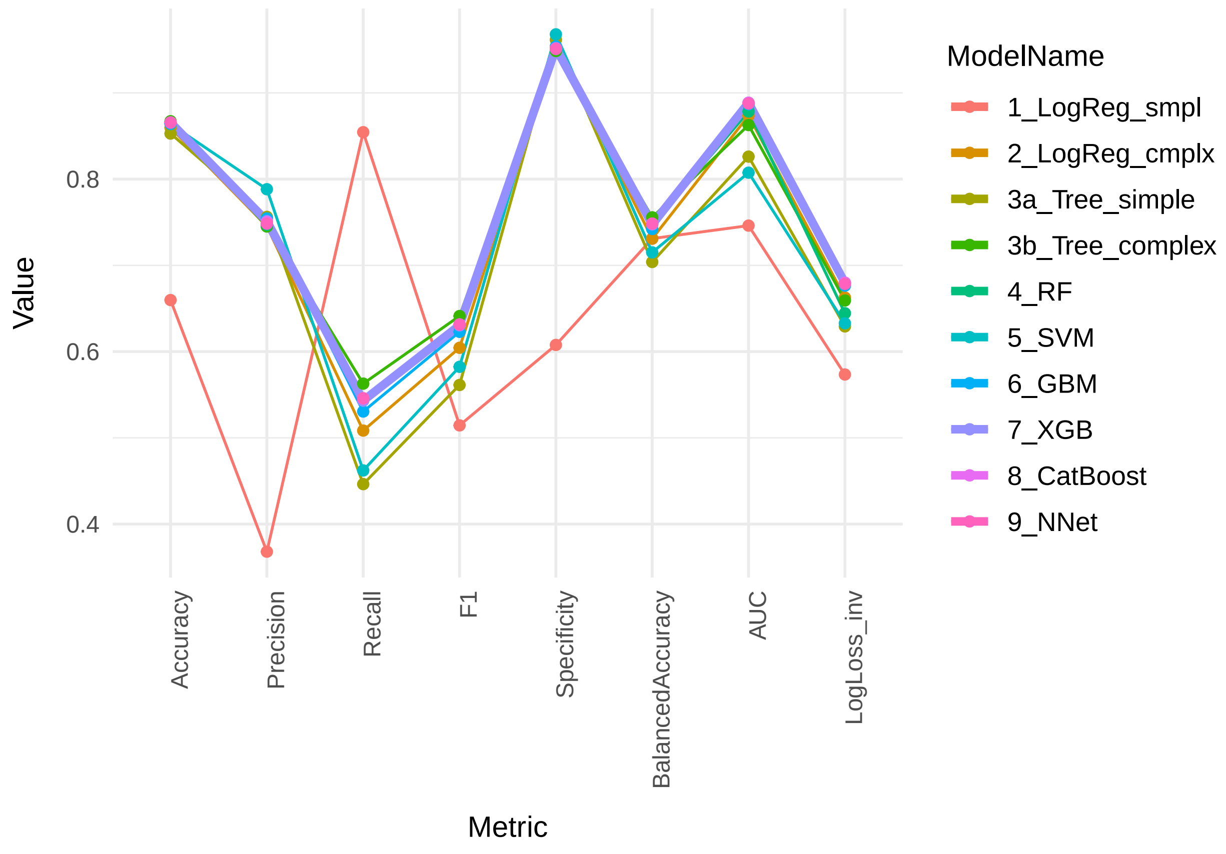
- The complex logistic regression was selected based on a forward variable selection algorithm, driven by BIC - which penalises for data size and number of parameters in the model, and therefore tends to deliver more parsimonious models. Confusion matrix figures reported were derived using an optimal probability threshold to max accuracy. The variables included as drivers were Number of Products, Age Band, Is Active Member, Geography, Gender and Balance Band.
- The simple decision tree was produced with a fixed complexity parameter of 0.01, to keep the tree relatively small, and produced the following, again with similar variables being the key drivers here: age, number of products, is active member, geography. The information in the leaves shows the prediction of exit (1) vs not (0), the proportion of exits, and the % of cases assigned to that leaf.



- Tuning the complexity parameter for accuracy led to the more complex tree referenced in the table
- The Random Forest was produced using the Rborist package within the caret framework, and the minimum node size was tuned as well as the number of predictors to test for each split (i.e. the predictors would be sampled without replacement). The mean probability output was not aligned with the average probability of exit across the training set, and therefore an extra step was added to calibrate the probabilities - this retained the discriminatory power of the model, but brought the mean predicted probabilities in line with the actual mean probabilities of exit in the training set
- The Support Vector Machine (SVM) was proving time consuming to train, and as such the training was done on a sample of 20,000 records, including the k-fold. i.e. the results in the table are not directly comparable to the other models, but are included for completeness. If increasing the sample size had been significantly improving performance, I may have invested the hours of runtime to do a full training set training, but it seemed unlikely to be able to compete with other methods giving significantly better results.

- The 3 boosted trees were produced using the gbm, xgbTree and catboost and gbm packages within caret. Rather than the random forest approach, which produces a forest of decision trees each generated by sampling the training data and the parameters to use, these boosted trees are sequential models, where each tree is built to correct the errors of the previous tree. The gbm package is the original package for this type of model, and the xgboost and catboost packages are more recent, and have been developed to be faster and more accurate.
- The Neural Net is again produced via the caret package, which uses the nnet package under the hood. This produces a shallow neural network with only one hidden layer (so one layer between the input layer of feature values and the output layer of Exit predictions), and then tunes for the number of nodes in the one hidden layer (as well as the decay parameter, which is a regularisation parameter to prevent overfitting). The results are therefore for a relatively simple, not a deep, neural network. The best results were obtained when normalising the continuous variables - something that isn't typically required for classification tree methods.

A plot of the performance on the respective measures is provided below (note 1- log loss has been used, so all metrics are "bigger = better"). The chosen XGBoost model is highlighted in the plot.



3.2 Application of the final chosen model to the test set

I decided to select the XGBoost model as the 'winner'. Even though it was very slightly outperformed by catboost on the validation sets, the difference was negligible and the training time for XGBoost was significantly quicker than catboost.

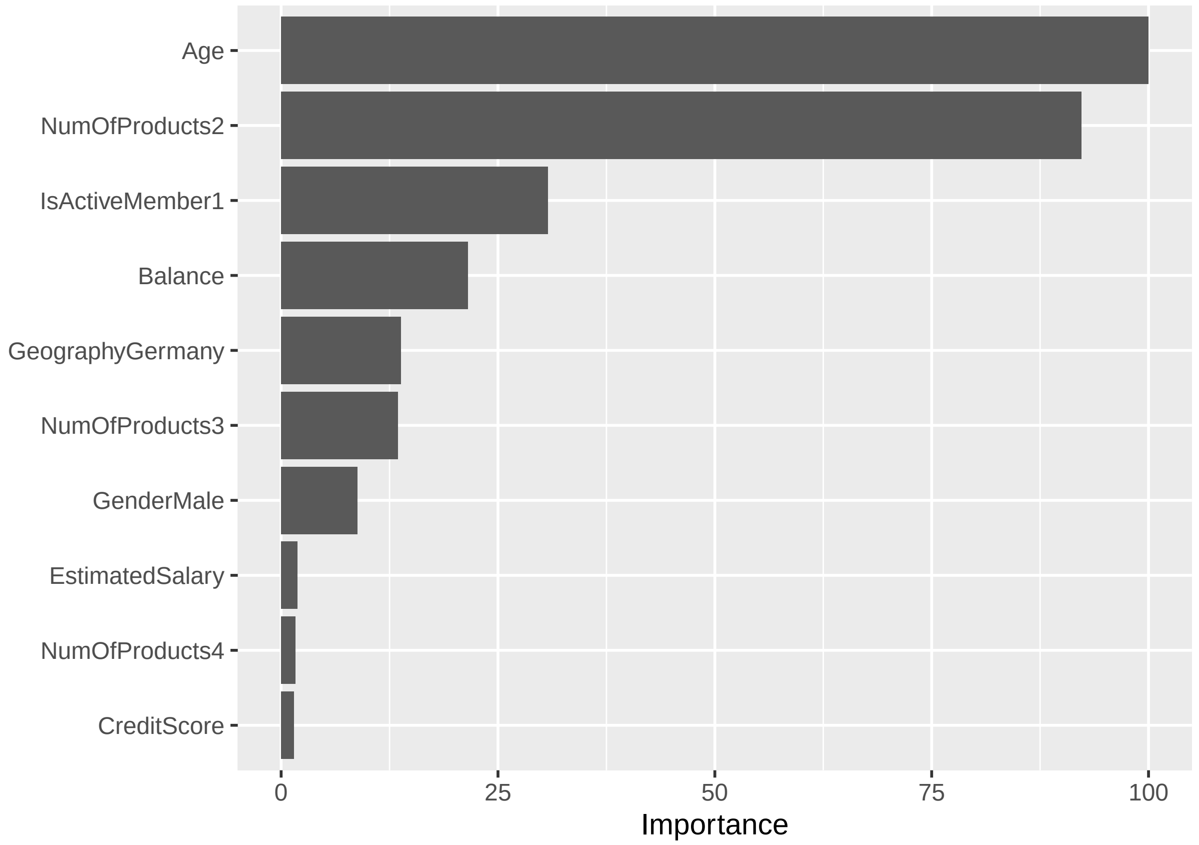
The 'more complex' single decision tree, did do very well on the discriminatory side, but performed less well on AUC and log loss, which is why I decided not to go with this approach.

I felt in a real world scenario, the predicted probability of exit was likely to be key, as this probability would be combined with calculations of predicted returns if the customer was retained to give an overall expected returns metric, and it would be this metric that would likely drive the decision making.

Table 8: Final Model: XGB - Test Set Results

ModelName	Accuracy	Precision	Recall	F1	Specificity	BalancedAccuracy	AUC	LogLoss
Final Model: XGB	0.862	0.751	0.534	0.624	0.952	0.743	0.89	0.322

The variable/feature importance plot for the final XGB model is shown below. This shows the top 10 most important features in the model, with the Age and number of products held being the most important features in predicting exit likelihood, which aligns with the Information Values plots in the initial analysis and findings throughout the modelling process.



4. Conclusion

The final XGBoost model gives a good prediction of the likelihood of a customer exiting the bank, and performs well across a range of performance metrics. The AUC being high suggests that the model is good at separating exiting vs non-exiting customers across a range of cut-offs, and the log loss being low suggests that the predicted probabilities are close to the actual probabilities of exit in the test set.

With more time and resources, the following are potential avenues for further investigation:

1. Digging deeper into the customer IDs to see if any value could be added there (even if, per the logic previously mentioned, it looked to be unlikely to be a unique identifier of a given customer). Given the high dimensionality of the ID variable though, some form of dimensionality reduction would likely be required on this route.
2. The addition of other features to the model - for example seeing whether adding 1-0 binary for balance vs no balance might have helped.
3. Trying some of the models outside of the caret package - caret, whilst convenient in offering standard coding for training and prediction, does limit the parameters that can be tuned vs the underlying packages, and so it's possible that a more tuned model could have been produced using the underlying packages directly.
4. Deeper neural networks may be an interesting avenue to explore, given the simple 1 hidden layer network produced in this analysis.

Finally, the number of features on offer here was fairly limited, and in a real world scenario, you'd expect to have a lot more data to work with and other useful variables correlating with exit likelihood, relating to both the customers and their product holdings with the bank.

Overall however, I am satisfied with the performance of the model, and I believe it would provide a useful tool in a commercial context.

Appendix: References

- The github repository for this project can be found at github.com/m-d-hale/BankChurn
- The data has been sourced from kaggle.com/datasets/rangalamahesh/bank-churn/data?select=train.csv