



Dublin Business School

CUSTOMER SEGMENTATION AND CHURN ANALYSIS: A CASE STUDY ON A GLOBAL FASHION RETAILER

Final Report

Higher Diploma in Science in Data Analytics

Nicola Killoran

10637107

10637107@mydbs.ie

24/05/2024

Supervisor: Rory O'Donnell

Abstract

Emerging in the 1990s, fast-fashion is a business model characterised by offering affordable clothing to the mainstream consumer that follows the latest trends or imitates designer brands. Key features of this business model include; affordable pricing, frequent product turnover and regular purchases of low- to medium-value. Rising population, increased disposable income and technological advancements are some key market drivers.

However, in recent years, this industry has faced mounting criticism regarding unethical manufacturing practices, textile waste and overconsumption (Joy *et al.*, 2012). Most global fast-fashion retailers have therefore incorporated sustainability initiatives in an effort to mitigate their environmental impact. The industry also faces growth constraints due to heightened competition and market saturation. Namely, ultra-fast-fashion companies like Temu and Shein have grown in popularity in recent years.

In the context of this project, understanding the purchase behaviours of customers, and the customer segmentation and churn analysis that follows this, is crucial to devise innovative customer retention strategies to maintain growth.

Acknowledgements

I would like to thank my supervisor Rory O'Donnell for providing his advice and guidance throughout the delivery of this project. I would also like to express my gratitude to lecturer Mehran Rafiee for his initial guidance and knowledge provided in the project.

Contents

Chapter 1: Introduction	4
Chapter 2: Literature Review	5
Chapter 3: Requirements Specification and Design	6
Chapter 4: Implementation	7
Business Understanding	7
Data Understanding	8
Data Preparation & Pre-processing	9
Data Modelling	11
Customer Segmentation	11
Customer Churn	13
Evaluation	14
Customer Segmentation	14
Customer Churn	14
Deployment	15
Chapter 5: Testing and Results	15
Chapter 6: Conclusions and Future Work	16
References and Bibliography	18
Appendix	19
Tables	19
Figures	20
Management Progress Report	30

Chapter 1: Introduction

Customer Lifetime Value (CLV) is a core metric in any customer relationship management strategy, which aims to build a long-lasting and profitable retention of customers. Such retention can improve firm value and help to coordinate differential marketing initiatives targeted to each customer segment (Kumar *et al.*, 2004); (Gupta *et al.*, 2004). The fast-fashion retail industry is fiercely competitive. Retailers must build loyalty and differentiate themselves through product offering and marketing in order to improve customer satisfaction, or risk facing high customer acquisition costs. In this context, the analytics behind gaining and maintaining a loyal customer base is ever topical. Without the use of big data and predictive analytics in the retail industry, it is difficult to competitively sustain a business. Such techniques can help to improve decision-making and maximise ROI, as retailers get timely and accurate information on their markets, customers and sales (Zineb, Najat and Jaafar, 2021).

This project is conducted on the Swedish multinational clothing company Hennes & Mauritz, also commonly known as H&M Group. H&M was founded in 1947, in Västerås, Sweden, a pioneer of this industry that has now expanded into one of the world's largest and most successful fast-fashion retailers with global net sales of approximately 236,035 million SEK in 2023 (H&M Group, 2024). The overall purpose of this study is to gain a better understanding into customer purchase and churn behaviour through the combination of exploratory and predictive techniques. The development of predictive models based on historical customer and transactional data is crucial in identifying the contributory factors and evaluating the risk of customers churning in an ever competitive environment.

RFM (Recency, Frequency and Monetary) analysis is used as the basis to segment customers and predict customer churn. RFM segmentation analyses customer value through three metrics: recency, frequency and monetary. It derives a score for each customer, differentiating between loyal, low-frequency and high-value customers in order to enable personalised marketing strategies for each segment that emerges. Customer churn, in a predictive context, is defined as the percentage of customers that will not likely make a purchase within a predefined time frame in the near future. Therefore, the connection to recency is apparent. While RFM is frequently implemented for market segmentation in the fashion industry, it is rarely combined and used as attributes in predictive churn modelling. To the best of my knowledge, this is the first study carried out to investigate customer segmentation and churn with RFM analysis using data from H&M.

Project scope:

- Establish business requirements
- Exploratory Data Analysis (EDA)

- Data Cleaning, Pre-processing and Transformation
- RFM Analysis
- Customer segmentation model with RFM values
- Binary classification models to predict customer churn likelihood
- Evaluate model results using appropriate techniques, improve model where necessary
- Determine the most robust and appropriate model suitable to business needs, translate to business insights
- Deploy model to H&M's customers

Chapter 2: Literature Review

Customer Segmentation

The concept of RFM was introduced by Bult and Wansbeek (1995) as a selection technique for direct marketing responses. RFM are numerical purchase-related variables that score the revenue and loyalty of individual customers. Empirically, this analysis has been proven as a useful method on large-scale data to explore and cultivate successful Customer Relationship Management (CRM) (Song *et al.*, 2017). Long-lasting relationships with loyal customers can maintain or increase market share. In addition, CLV can be calculated based on a weighted RFM function for each segment, however formula weights will differ based on business model and industry (Khajvand *et al.*, 2010).

In the context of the fashion industry, several recent empirical studies have implemented RFM values in customer segmentation. K-means clustering algorithms define homogeneous customer groups in an effort to devise targeted marketing strategies (Anitha and Patil, 2022; Sommella and Sorrentino, 2023). However, as RFM uses historical data, it therefore has some limitations on predicting the future purchase possibilities for customers (Reinartz and Kumar, 2002).

Customer Churn

Churn prediction is critical in a CRM strategy for the fashion industry in order to identify loyal, high-value and risky customers. Managers can retain loyal customers by providing them with satisfactory products and services that meet their needs. Additionally, by identifying the factors that persuade a customer to churn, managers can take mitigating actions (Edelstein, 2001). The integration of RFM and classification techniques has been studied by (Khodabandehlou and Zivari Rahman, 2017). In this study, RFM variables were implemented into various ML algorithms to predict customer churn. Compared to the bagging model (decision trees), boosting algorithms, such as artificial neural networks, were superior and achieved the highest prediction accuracy.

Data Science Process: CRISP-DM

The industry-proven CRISP-DM methodology is chosen for this project to ensure a structured and timely implementation. The process is not linear, where multiple iterations of each step is possible, in particular with information extraction, modelling and evaluation stages (Chapman *et al.*, 2000). There are six stages to the CRISP-DM framework:

1. Business Understanding: Translate primary business objectives into a data mining problem.
2. Data Understanding: Explore and describe data to check quality.
3. Data Preparation: Clean, format and feature-engineer the data. Good quality data is critical in achieving successful results, therefore data exploration and transformation is an iterative process.
4. Modelling: Build the selected data-mining models.
5. Evaluation: Assess model results against business objectives. If not, the modelling stage is iterated until satisfactory results are achieved.
6. Deployment: Plan the deployment, monitor and maintain the model in a production environment in order to facilitate business decisions.

The implemented CRISP-DM methodology is discussed in Chapter 4 below.

Chapter 3: Requirements Specification and Design

As per the high-level design below, the following technologies and platforms will be implemented to complete the scope of this project:

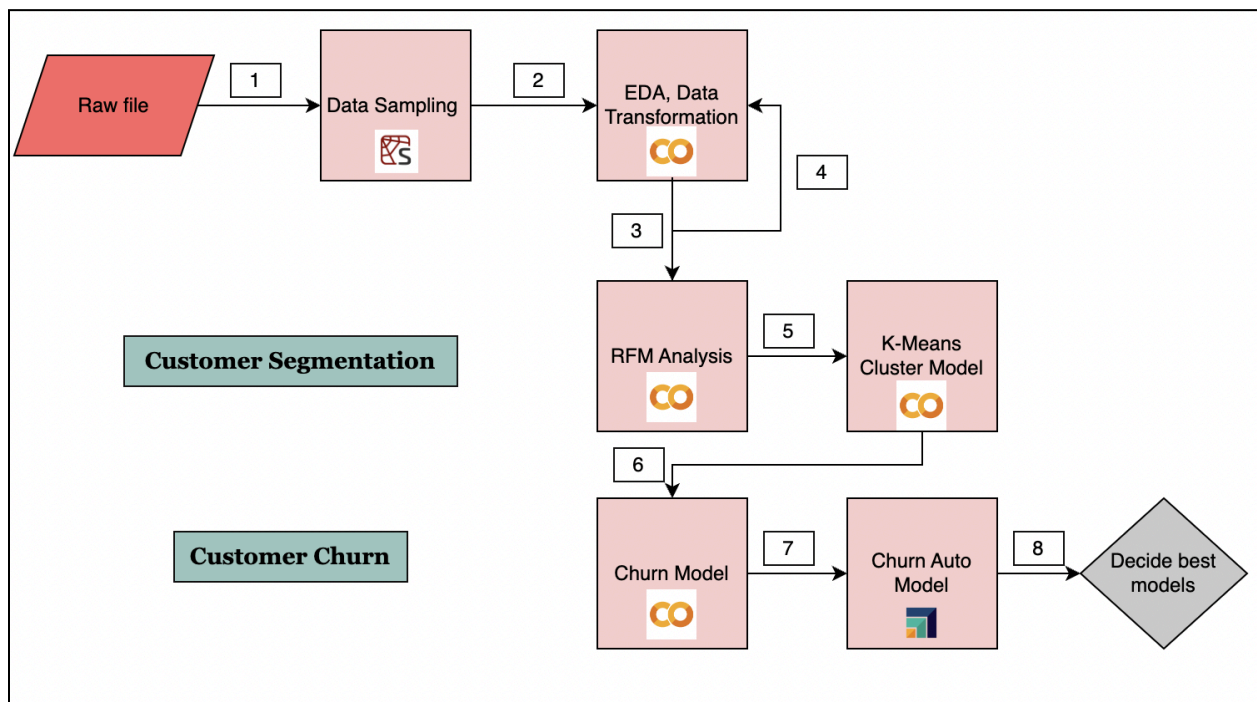
- Spyder 5.5.3
- Google Colab IDE (Python 3.10)
- RapidMiner Studio

High-level Design Steps

1. Load original dataset into Spyder IDE to perform data sampling. Export sampled data as csv file.
2. Load sampled dataset into the Google Colab IDE to easily perform initial EDA, data pre-processing and transformation techniques.
3. Perform RFM Analysis in Google Colab.
4. Feedback loop. As previously discussed in the literature review, data transformation and pre-processing is an iterative process.

5. Perform K-Means Clustering modelling based on RFM values in Google Colab.
6. In preparation for classification models, perform feature engineering and selection. Run multiple classification models in Google Colab to predict the likelihood of customer churn. Tune model parameters and implement cross validation to improve results. This is an iterative process to achieve a robust, optimal model. Finally, evaluate models using appropriate metrics.
7. Export cleaned, one-hot encoded dataframe to RapidMiner AutoModel to validate results.
8. Compare results with models run on Google Colab. Identify the best churn model that meets business requirements.

High Level Design Graph



Chapter 4: Implementation

As previously discussed, this project will follow the CRISP-DM methodology as follows:

Business Understanding

This step involves research on customer segmentation and churn, alongside H&M's business model in order to explore what H&M could expect to gain from this project. According to (Chapman *et al.*, 2000), the determination of the data mining project objective is one of the most critical steps in this phase.

Primary Business Objectives: Primary business objectives are set for H&M in order to combat the ever-evolving challenges of the fast-fashion industry. As discussed in previous chapters, fast-fashion retailers are facing increased competition. H&M must find ways to remain profitable despite rising customer acquisition costs. The company must cultivate positive existing customer relationships in order to maximise CLV and market share. .

Data Mining Project Objectives: The project goals are to use historical transactional information to profile different customer segments and generate a binary classification model that predicts the likelihood of customer churn. With churn prediction, a false negative (predicting a customer will not churn when they actually do) is more costly due to high customer acquisition costs. Therefore, as the cost of false negatives outweighs the cost of false positives, the most important business success criteria is defined as recall. However, an acceptable level of precision will be maintained to avoid the excessive costs from too many false positives.

Data Understanding

H&M customer transaction data is obtained for a timeframe spanning two years, from 20/09/2018 - 22/09/2020. A brief description of each variable in the original dataset can be found in Table 1.1.

Random Sampling

As the original dataset contained over 31 million rows, a random sample is taken using a fixed number of 1,000 customer IDs. All transactional data rows for each sampled customer were retained. The sampled dataset contains 23,031 rows and 11 columns, making it easier to work with.

Exploratory Data Analysis (EDA)

EDA was carried out through visual inference and descriptive statistics in order to gain an initial understanding of customers and their transactions at an aggregate level. In addition, pandas profiling was installed to generate a comprehensive report of the structure and characteristics of the dataset, including variable distributions and the cardinality of categorical variables. From this preliminary analysis, further areas of data preprocessing were determined. This report, alongside the initial visual analysis of the dataset, can be seen in the Python script.

Correlation

A correlation heatmap is created to understand the direction and strength of association between variables (see Figure 1.1).

Data Preparation & Pre-processing

Alongside multiple data manipulation processes, the following steps were followed to improve the quality of the dataset:

Missing Values

Missing values exist in the dataset, concentrated in certain columns (see Figure 1.2).

- Missing values in the *Age* column are missing completely at random (MCAR) and therefore replaced with the average age.
- As seen in the missing values heatmap (Figure 1.3), missing values in *Active Customer* and *FN* columns are not missing at random (NCAR). There is a positive correlation between variables because there is no customer need, i.e. customers are no longer active and therefore do not have an FN subscription. In these instances, NaN values are replaced with 0 to indicate an inactive customer or subscription.

Remove Duplicate Rows

Duplicate purchases are rare in the dataset, and therefore do not represent genuine customer behaviour, but data entry error. It is very unlikely that a customer bought more than one of the same products on the same day. Therefore duplicates are removed to improve data integrity.

One-Hot Encoding

Categorical columns *club_member_status* and *fashion_news_frequency* are converted to numerical values suitable for ML algorithms.

Outlier Removal with Z-scores

Outliers can skew model generalisation, and therefore adversely impact predictive power and accuracy. Using box-plots, only a scatter of irregularities appear in *Age*. This is deemed as noise, not outliers, and are thus not removed. Irregularities in *Price* are removed using a z-score threshold of ± 5 standard deviations. In experimentation, a higher z-score threshold removes fewer data points compared to lower thresholds. This suggests that extreme outliers are relatively rare as prices are relatively concentrated around the mean. Therefore, a higher z-score threshold captures only the most extreme outliers, which are few in number.

This threshold limit is justifiable given business context. H&M is a fast-fashion brand that offers trending fashion at affordable prices. Therefore, a maximum price of €422, as in the original dataset, is deemed extreme for H&M's business model and is thus removed as an outlier. After outlier removal at this z-score threshold, the maximum price is €122.

Feature Preparation for Customer Churn Model

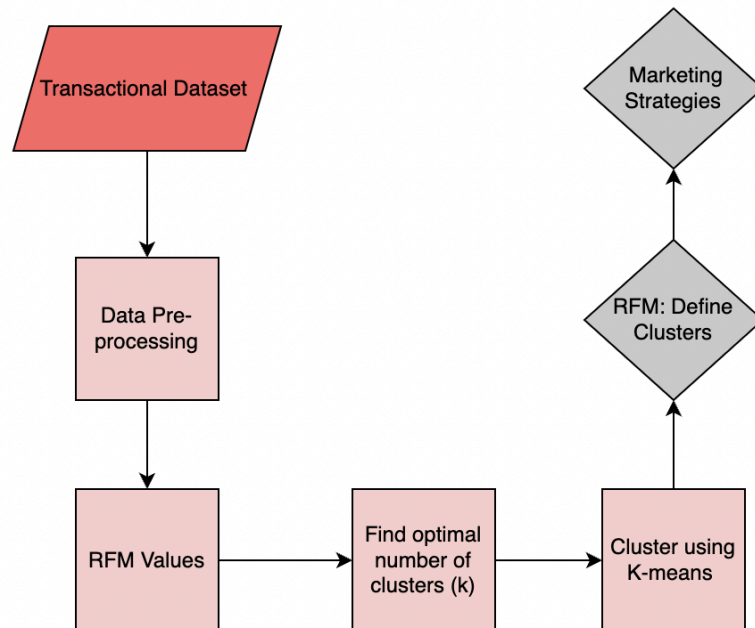
Customers' likelihood to churn is measured using a supervised churn prediction model. The preparation steps for this are outlined below:

1. *Merge and Aggregate Columns*: RFM values are merged onto the cleaned, one-hot encoded dataframe. In addition, *Price* is aggregated to one row per customer, whilst still preserving the integrity of all other features.
2. *Define a proxy churn indicator*: As this dataset contains no historical binary churn labels, a proxy indicator is created using *Recency*. A churned customer (1) is defined as customers who have not made a purchase within the last 6 months. All other customers are defined as not churned (0). A 6-month churn definition is suitable for H&M as it aligns with seasonal purchasing cycles. The business model is designed to make customers purchase regularly, so to keep updated with fast-fashion trends.
3. *Feature Engineering*: The average purchase value for each customer is engineered by dividing *Monetary* by *Frequency*.
4. *Feature Selection*: Identifier columns (*Article_ID* and *customer_ID*) are dropped. If predictors are highly correlated with each other or with the target variable, it can lead to multicollinearity issues. *Recency* is used to define churn, and therefore it must be dropped. *Frequency* and *Monetary* do not have a strong correlation with the target, but they do with each other. In addition to the variables that have a VIF of over 10, the following variables are also dropped; *t_dat*, *Active*, *Price*. The correlation matrix can be seen in Figure 4.1.
5. *Standard Scaling*: Numerical variables are standard scaled.
6. *Target Variable Imbalance*: SMOTE analysis is implemented to remove bias towards the majority class by oversampling the minority class (churners). Although such balance is rarely existent in reality for H&M, balanced target classes ensure that the model's learning ability is not hindered by biases.

Data Modelling

Customer Segmentation

Framework for Customer Segmentation



RFM Analysis

1. Recency: Time since last transaction. A lower recency value indicates that a customer is potentially more receptive to brand communications or promotions.
2. Frequency: Frequency of transactions. A higher frequency indicates a higher level of loyalty, engagement and satisfaction.
3. Monetary: Total transaction value. It differentiates high-spending from low-spending customers.

RFM Implementation

Recency, frequency and monetary scores are calculated for each customer:

- *Recency*: The most recent transaction date of the entire dataset is 22/09/2020. Recency is calculated by subtracting each customer's last purchase date from this benchmark date. The result is the number of days since the customer's last purchase.
- *Frequency*: Count the number of transactions per customer.
- *Monetary*: Calculate the sum of prices (i.e., the aggregated total transaction value) for each customer

Sampled RFM values and summary statistics can be seen in Figure 2.1 and 2.2 respectively. After calculation, RFM values are ranked and normalised by linear scaling (the highest rank being 100). Normalisation brings RFM values to common scale, whilst still preserving the original distribution of the data. See ranking methodology below:

Recency	Descending order	Customers with more recent purchases have higher ranks.
Frequency	Ascending order	Customers with higher purchase frequency have higher ranks.
Monetary	Ascending order	Customers with higher aggregate monetary value have higher ranks.

Calculate CLV and determine score weightings according to business model: As discussed in the literature review, RFM scoring can be used as a measure of CLV. H&M is a retail business selling lower-medium priced fashion. Its business model is designed to get customers to shop frequently, even if the monetary value is not always high. Although accumulated monetary value is still important, less emphasis will be given to it in scoring. The CLV score is calculated by giving more weight to the frequency and recency scores in comparison to monetary. A scale of 0-5 is implemented to make results more interpretable (see CLV scores in Figure 2.3).

CLV score formula:

$$(0.4 * \text{Recency Score}) + (0.4 * \text{Frequency Score}) + (0.2 * \text{Monetary Score})$$

Implementation of K-means Clustering Algorithm

Customers are segmented using a K-means algorithm based on RFM values. K-means clustering is an unsupervised learning method that segments data into groups based on similarities or patterns. Initially, centroid positions are selected at random and then optimised through iterative calculations. This algorithm is chosen for its simplicity and interpretability.

Preparation for Modelling: K-means clustering gives optimal results when the data is not skewed and is standardised. After checking for skewness, RFM values are standardised. As the K-means clustering model calculates distances between data points, and such distances are influenced by the feature's scale, standardisation is important as it ensures all features contribute equally.

Modelling:

- The standardised RFM values are inputted into the model.

- Deploying Elbow Technique, the optimal number of clusters (K) is defined as four. On trial experimentation, both four and five clusters are run using a maximum number of ten iterations with different centroid seeds (see Figure 3.1).
- By visualising with cluster mapping and snake plots, four clusters segment customers most effectively (see Figures 3.2 and 3.3).
- Cluster labels are defined for each cluster in order to distinguish between customer behaviour and preferences (see Figure 3.4). Further details of these defined customer groups are given in the Testing and Results section below.

Customer Churn

Customer Churn Propensity Models

Modelling:

- The data is randomly split into a train-test split with a ratio of 75:25.
- A data modelling pipeline is created to determine the best baseline model from a range of classifiers: K-Nearest Neighbors, Support Vector (SVC), Decision Tree, Random Forest (RF), AdaBoost, Gradient Boosting and Logistic Regression. Cross validation is performed on the training set. The k-fold for cross-validation is set to ten, which averages model performance across ten different train-validation sets, thus providing a more robust estimate of performance in comparison to a single test-train split (see Figure 4.3).
- The ROC curve and AUC (area under the ROC curve) scores were compared for each baseline classifier. RF performs best with an AUC of 0.81, signifying that the model is better than random guessing.

Hyperparameter tuning of the best baseline model - Random Forest:

- By implementing grid search and k-fold cross validation, the RF model is refined to select the ideal hyperparameters based on recall score. The number of decision trees, alongside the maximum depth of trees is refined.
- The predictive power of features varies greatly (see Figure 4.2). To avoid the curse of dimensionality, scikit-learn's '*SelectKBest*' selector is also implemented to select only the most important features. The number of model features (k) is refined to the top six predictors (see Figure 4.4.1).
- Hyperparameter tuning is iterated until satisfactory results are achieved (see Figure 4.4.2). The RF model is then refitted with the hyperparameters (see Figure 4.5).

Customer Churn Propensity Models on RapidMiner AutoModel

A churn model is also run on RapidMiner AutoModel to validate model results and improve robustness. The one-hot-encoded dataset is uploaded to RapidMiner (see Figure 5.1). A classification task was selected to predict the probability that a customer will churn.

Feature Selection: See Figure 5.2

Feature Importance: See Figure 5.3

Evaluation

Customer Segmentation

Evaluation of K-Means Customer Segmentation Model

Model quality is evaluated based on the Davies-Bouldin Index (DBI), where lower values indicate superior clustering. A DBI of 0.8 is considered to be indicative of reasonably good clustering. It's not perfect, but typically acceptable in practical scenarios.

Customer Churn

Models performance is evaluated using unseen data (i.e., the test set) to assess the true predictive power. Using the predict method, test data is inputted and classified.

Evaluation of Hyperparameter Tuned RF Classifier

AUC summarises classification performance across different thresholds. This model results in an AUC score of 0.81, which is the same as the baseline model (see Figures 4.6 and 4.7 for evaluation metrics). The closer the AUC is to 1, the better the model performance. This indicates a good ability in distinguishing between positive and negative churning outcomes. There is a 82% probability that the model will correctly distinguish between a randomly chosen positive and negative instance. The confusion matrix plots the predicted and actual values of the classifier (see Figure 4.8 for interpretation).

Robustness Checks for Overfitting: This model does not show significant signs of overfitting, in accordance with the specified 0.2 threshold for differences in recall and AUC (see Figure 4.5). However, while such differences are deemed acceptable, a noticeable drop in precision and F1-score (overall balance) from training to test data still exists, possibly due to recall being prioritised in hyperparameter tuning (see Figure 4.6). This performance decrease should potentially be investigated further, possibly through additional feature selection or regularisation.

Evaluation of Customer Churn Propensity Models on RapidMiner

The Logistic Regression model was the best performing model when run on RapidMiner (see ROC Curve comparisons, performance evaluation and confusion matrix in Figure 5.4, 5.5 and 5.6 respectively). Given the trade-off between sensitivity (recall) and specificity, recall is deemed more important in customer churn prediction as the cost of false negatives outweighs the cost of false positives. However, it must be noted that the Logistic Regression Model is not robust to the target imbalance that exists. Further details of all model results will be discussed in Chapter 5.

Deployment

When deployed to the production environment, the clustering model is used to segment customers based on recent transactions and to target promotions for each specific segment. Additionally, the best classification model is deployed and implemented to provide real-time churn predictions for H&M's customers. Once models are deployed, it is imperative to continuously monitor performance and make suitable changes if necessary. At this phase, this final report and presentation of results are also produced.

Chapter 5: Testing and Results

Results of Customer Segmentation using K-Means:

H&M has a mixed customer base, with patron high-value customers being the largest segment (33.9%) and recent customers being the lowest (15.9%). The distribution of customers across each segment can be seen in Figure 3.5 below.

Cluster	Group	Profile
0	Recent Customers	Customers that have high recency, but relatively lower frequency and monetary values. There is still potential to create long-term value with this segment.
1	Patron, High-value Customers	Compared to all four clusters, cluster 1 has the highest RFM scores.
2	At-risk Customers	Customers that have the lowest RFM scores, compared to all other clusters. Higher risk of customer churn.
3	Loyal Customers	Customers that have moderate RFM scores. Such customers still purchase more frequently, at higher monetary values, in comparison to cluster 0 and 2.

Subsequently, marketing strategies can be tailored to each of the defined customer segments in the following ways in order to gain real business value:

<i>Patron, High-value customers</i>	In order to make customers feel appreciated, premium benefits could include personalised discounts on top-selling products or early access to H&M's designer collaborations.
<i>Loyal Customers</i>	Appreciation campaigns could include additional loyalty rewards or referral bonuses.
<i>Recent Customers</i>	Campaigns could focus on H&M brand education and time-sensitive discounts on subsequent purchases.
<i>At-Risk Customers</i>	Re-engagement campaigns through push-notifications or email could include targeted discounts or promotions to encourage browsing activity and purchases.

Results of Customer Churn Propensity Models

The best customer churn model is the hyperparameter tuned RF model with an AUC score of 0.81 and a test data recall of 0.74. Random Forest is an ensemble model that is robust to multicollinearity. Unlike the best RapidMiner model, the RF model considers the VIFs between variables and has increased robustness to target class imbalance. However, unlike the results of Khodabandehlou and Zivari Rahman (2017), this project finds a bagging technique superior in comparison to boosting algorithms.

Chapter 6: Conclusions and Future Work

In the increasingly competitive fast-fashion industry, analysing customer behaviour is imperative to maximise positive customer relationships and increase market dominance. The scope of this project can be split into two parts: customer segmentation and churn prediction. Firstly, H&M's customer base is segmented into clear and meaningful customer profiles using RFM values on a K-means clustering model. This segmentation gives meaningful insights to managers on how to refine effective marketing strategies for each segment in an effort to build retention and avoid high customer acquisition costs.

The second objective of this project is to build a robust classification model to predict customer churn using RFM values, customer demographic and historical transaction data. The hyperparameter-tuned Random Forest classifier is deemed to be the most sufficiently accurate in predicting customer churn, exhibiting the highest recall rate. From gaining a better understanding of the factors that may cause a customer to churn, H&M can take mitigating

actions to reduce churn rate. A key insight is that low frequency and a low average purchase value are primary predictors of a customer that has a high probability of churning.

Model Limitations

Although customers were reasonably segmented, there exists some additional limitations to the K-means clustering model that extend from the standard. Segmenting customers based on RFM values produces segments that are indicative of historical purchase behaviours and customer profitability. However, although it gives some indication, it cannot fully predict future customer development. This will be influenced by customer age, income, education etc. Furthermore, as customers are solely segmented based on RFM values, exclusive of demographic information, marketing implications are given at a collective level. Each customer still has individual preferences and needs.

Additionally, customers with the highest RFM scores were deemed the most valuable. However, additional factors also determine customer value, such as Customer Engagement Value (CEV). According to Appelbaum (2001), emotionally engaged and rationally loyal customers are the most valuable. Based on CEV, value is derived from customer referrals, feedback and social media influence. Therefore, the fact that customer value is derived solely from RFM values is limiting.

Secondly, limitations also exist for the churn models. By defining churn using *Recency* as a proxy, this introduces several limitations that can adversely affect model accuracy. The use of a single dimension could oversimplify customer behaviour and not fully represent the multiple underlying contextual factors that induce churn. Additionally, the available data is limited to two years. Forecasting model results must therefore take into account future market developments and changes in customer preferences. In consequence, further research is required to enhance the accuracy of findings and to improve the practical relevance of managerial recommendations.

Future Research

In segmentation, future research could combine RFM values with customers' demographic information to offer more precise and practical suggestions to H&M. For churn prediction, further research could focus on obtaining a well-defined binary churn variable based on historical data. This would improve robustness and validity of predictions, gaining more valuable managerial implications for the company.

References and Bibliography

- Anitha, P. and Patil, M.M. (2022) 'RFM model for customer purchase behavior using K-Means algorithm', *Journal of King Saud University - Computer and Information Sciences*, 34(5), pp. 1785–1792. Available at: <https://doi.org/10.1016/j.jksuci.2019.12.011>.
- Appelbaum, A. (2001) *The Constant Customer*, Gallup.com. Available at: <https://news.gallup.com/businessjournal/745/Constant-Customer.aspx> (Accessed: 18 May 2024).
- Bult, J.R. and Wansbeek, T. (1995) 'Optimal Selection for Direct Mail', *Marketing Science*, 14(4), pp. 378–394. Available at: <https://doi.org/10.1287/mksc.14.4.378>.
- Chapman, P. et al. (2000) 'CRISP-DM 1.0 step-by-step data mining guide', in Springer.
- Edelstein, H. (2001) 'Building profitable customer relationships with data mining', in *Customer Relationship Management: The Ultimate Guide to the Efficient Use of CRM*. Wiesbaden: Vieweg+Teubner Verlag, pp. 339–351. Available at: https://doi.org/10.1007/978-3-322-84961-8_26.
- Gupta, S., Lehmann, D.R. and Stuart, J.A. (2004) 'Valuing Customers', *Journal of Marketing Research*, 41(1), pp. 7–18.
- H&M Group (2024) *Sales of the H&M Group worldwide 2022*, Statista. Available at: <https://www.statista.com/statistics/252190/gross-sales-of-the-h-and-m-group-worldwide/> (Accessed: 23 February 2024).
- Joy, A. et al. (2012) 'Fast Fashion, Sustainability, and the Ethical Appeal of Luxury Brands', *Fashion Theory*, 16(3), pp. 273–295. Available at: <https://doi.org/10.2752/175174112X13340749707123>.
- Khajvand, M. et al. (2010) *Estimating Customer Lifetime Value Based on RFM Analysis of Customer Purchase Behavior: Case Study*, *Procedia Computer Science*. Available at: <https://doi.org/10.1016/j.procs.2010.12.011>.
- Khodabandehlou, S. and Zivari Rahman, M. (2017) 'Comparison of supervised machine learning techniques for customer churn prediction based on analysis of customer behavior', *Journal of Systems and Information Technology*, 19(1/2), pp. 65–93. Available at: <https://doi.org/10.1108/JSIT-10-2016-0061>.
- Kumar, V., Ramani, G. and Bohling, T. (2004) 'Customer lifetime value approaches and best practice applications', *Journal of Interactive Marketing*, 18(3), pp. 60–72. Available at: <https://doi.org/10.1002/dir.20014>.
- Reinartz, W.J. and Kumar, V. (2002) 'The Mismanagement of Customer Loyalty', *Harvard Business Review*, 1 July. Available at: <https://hbr.org/2002/07/the-mismanagement-of-customer-loyalty> (Accessed: 20 May 2024).
- Sommella, E. and Sorrentino, A. (2023) 'Digital Customer Relationship Management (e-CRM) in the Fashion Industry', in M. Brandstrup et al. (eds) *The Garment Economy: Understanding History, Developing Business Models, and Leveraging Digital Technologies*. Cham: Springer International Publishing (Springer Texts in Business and Economics), pp. 287–305. Available at: https://doi.org/10.1007/978-3-031-33302-6_15.

Song, M. *et al.* (2017) 'Statistics-based CRM approach via time series segmenting RFM on large scale data', *Knowledge-Based Systems*, 132, pp. 21–29. Available at: <https://doi.org/10.1016/j.knosys.2017.05.027>.

Zineb, E.F., Najat, R. and Jaafar, A. (2021) 'An Intelligent Approach for Data Analysis and Decision Making in Big Data: A Case Study on E-commerce Industry', *International Journal of Advanced Computer Science and Applications*, 12(7). Available at: <https://doi.org/10.14569/IJACSA.2021.0120783>.

Appendix

Tables

Table 1.1: Data Variables (before Data Preparation)

Column Name	Description	Data Type
t_dat	Date of Transaction in format YYYY-MM-DD (string)	Object
customer_id	Customer ID	Object - Numerical
article_id	Article ID	Integer - Numerical
price	Price of article	Float - Numerical
sales_channel_id	Sales Channel: Channel 1 - In-store Channel 2 - Online	Integer - Categorical
FN	Fashion News Status: Binary feature - 1.0 or NaN	Float - Categorical
Active	Is Active customer: Binary feature - 1.0 or NaN	Float - Categorical
club_member_status	Is an active H&M rewards club member	Object - Categorical
fashion_news_frequency	Frequency of sending marketing communication to customer via the 'Fashion News Subscription'	Object - Categorical
age	Age of Customer	Integer - Numerical
postal_code	Postal Code of Customer (anonymised)	Object

Figures

Figure 1.1: Correlation Matrix

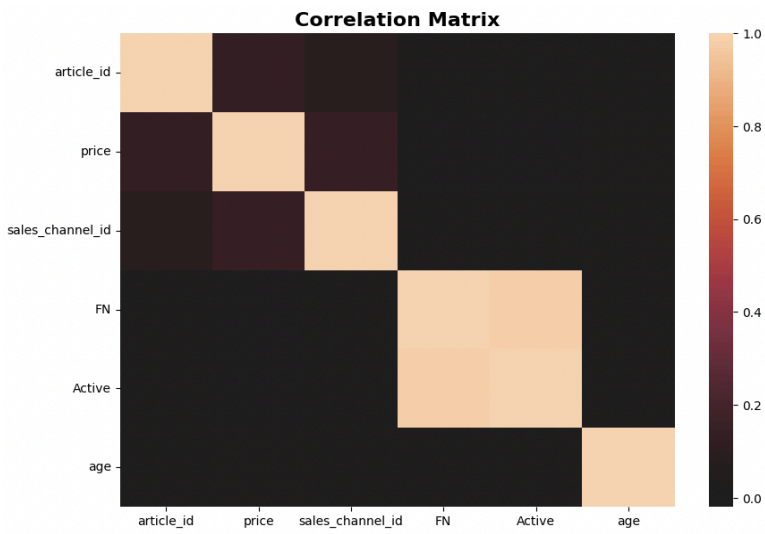


Figure 1.2: Missing Value Map

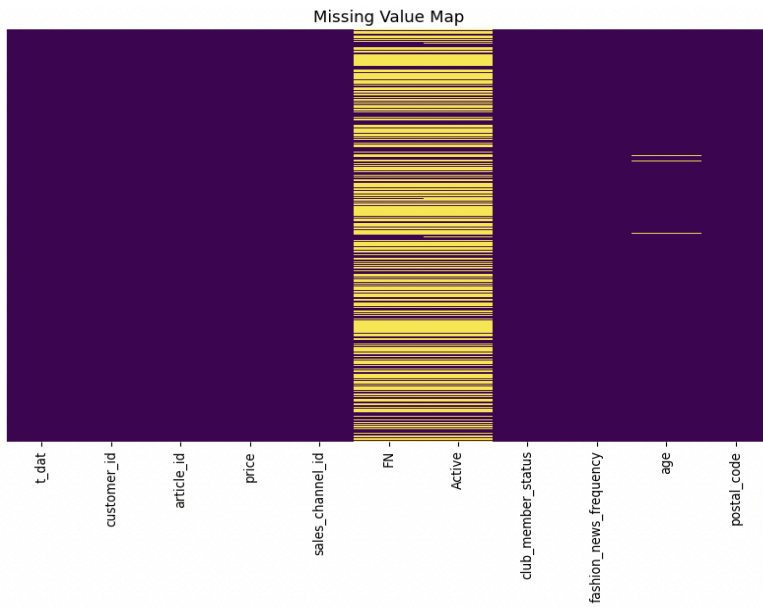
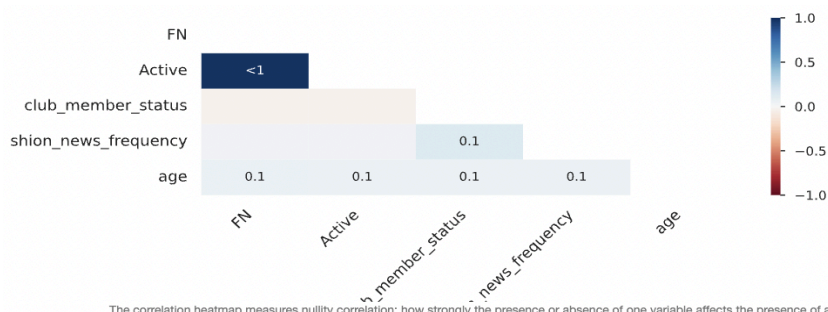


Figure 1.3: Missing Value Heatmap



The correlation heatmap measures nullity correlation: how strongly the presence or absence of one variable affects the presence of another.

Figure 2.1: RFM Values

```
# Combine RFM values into one new Dataframe
rfm_df = df_recency.merge(frequency_df, on='customer_id')
rfm_df = rfm_df.merge(monetary_df, on='customer_id').drop(
    columns='Last_Purchase_Date')
rfm_df.head()
```

	customer_id	Recency	Frequency	Monetary
0	001287c94ccec60e96adb79cb106c7c810b3cd9faa7e8b...	144	7	241.83
1	002faf80a68267264102e08eb4f1f21a59236773e4ab90...	0	113	3450.03
2	004c5c9ff1686fac7934d69a7a0abd33a5d111f2d894fb...	635	12	235.41
3	00549656241b634c8e0e6e5476b9b6ff1e3258b55c61c2...	630	3	25.37
4	0069ce62213d1232912ce510906a9b735d6fb08c54b7c2...	613	10	183.30

Figure 2.2: Summary Statistics of RFM Values

	Recency	Frequency	Monetary
count	997.000000	997.000000	997.000000
mean	236.424273	20.964895	559.859970
std	214.398878	38.326852	1068.265929
min	0.000000	1.000000	3.370000
25%	54.000000	3.000000	72.830000
50%	165.000000	9.000000	233.780000
75%	391.000000	22.000000	586.060000
max	729.000000	544.000000	14871.790000

The average recency is quite high (236 days), due to high standard deviation. Customers purchase 21 times on average and spend an average of €600 across all transactions. High variation across values exists, and therefore values are appropriately scaled before the K-means algorithm.

Figure 2.3: CLV Scores based on a weighted-average RFM Function

```
# Define RFM/CLV Function
rfm_df['CLV_Score'] = 0.4*rfm_df['R_rank_norm']+0.4 * \
    rfm_df['F_rank_norm']+0.2*rfm_df['M_rank_norm']

# Set scale to between 0-5
rfm_df['CLV_Score'] *= 0.05

# Round to two decimals and print head
rfm_df = rfm_df.round(2)
rfm_df[['customer_id', 'CLV_Score']].head()
```

	customer_id	CLV_Score
0	001287c94ccec60e96adb79cb106c7c810b3cd9faa7e8b...	2.47
1	002faf80a68267264102e08eb4f1f21a59236773e4ab90...	4.93
2	004c5c9ff1686fac7934d69a7a0abd33a5d111f2d894fb...	1.82
3	00549656241b634c8e0e6e5476b9b6ff1e3258b55c61c2...	0.71
4	0069ce62213d1232912ce510906a9b735d6fb08c54b7c2...	1.71

Figure 3.1: K-Means Clustering Model

```
# Build Model with 4 and 5 Clusters

# Define the number of clusters
num_clusters_4 = 4
num_clusters_5 = 5

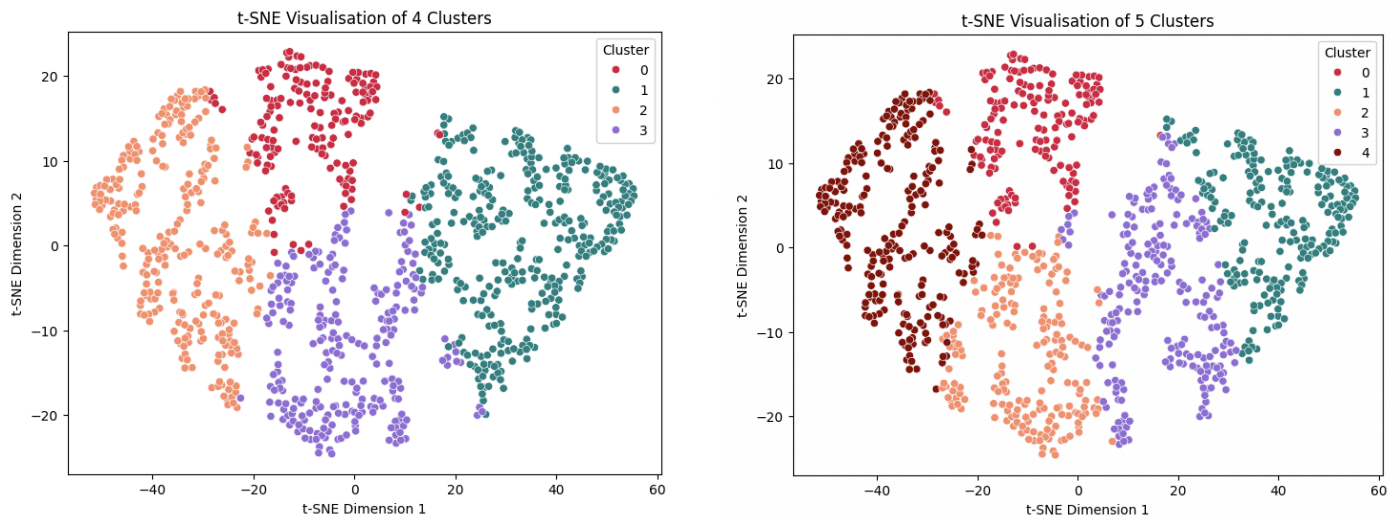
# Define the number of initialisations
n_init = 10 # You can adjust this value as needed

# Perform K-means clustering with 4 clusters
kmeans_4 = KMeans(n_clusters=num_clusters_4, n_init=10, random_state=1)
# Fit the model
df_rfm_kmeans_4 = kmeans_4.fit_predict(X)
# Retrieve centroids for 4 clusters
centroids_4 = kmeans_4.cluster_centers_

# Perform K-means clustering with 5 clusters
kmeans_5 = KMeans(n_clusters=num_clusters_5, n_init=10, random_state=1)
# Fit the model
df_rfm_kmeans_5 = kmeans_5.fit_predict(X)
# Retrieve centroids for 5 clusters
centroids_5 = kmeans_5.cluster_centers_

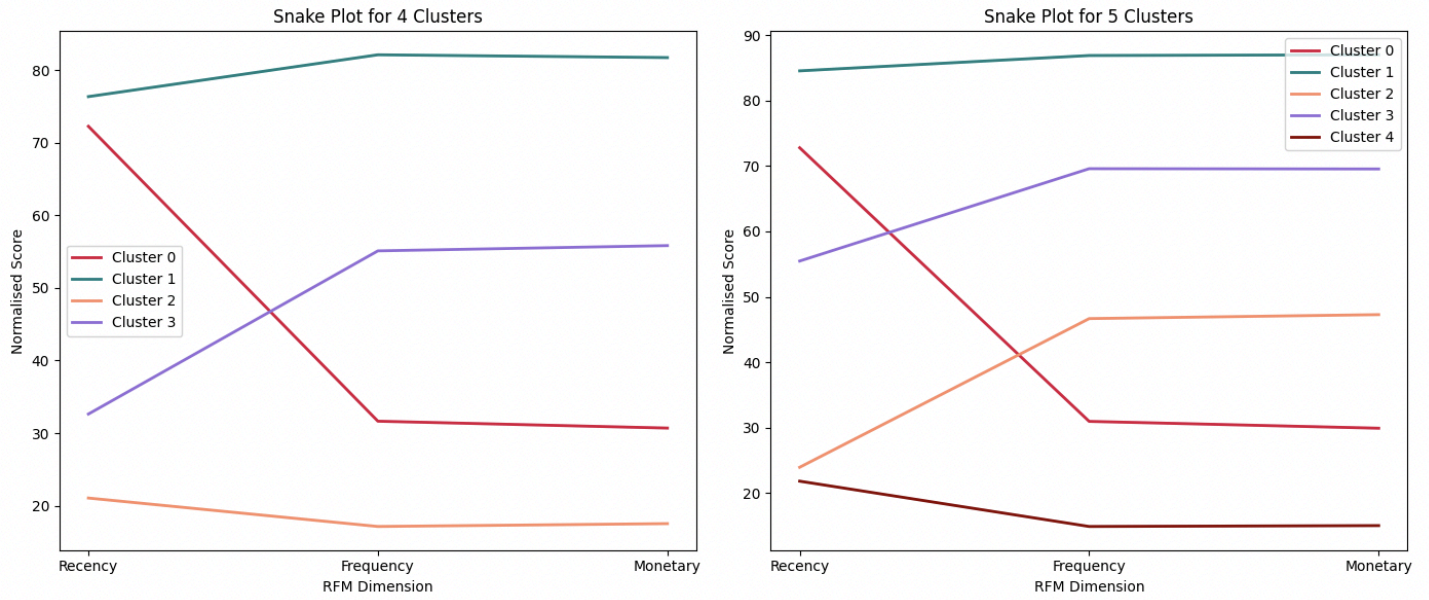
# The n_init parameter is set to 10, which determines that the algorithm will run with 10 different centroid seeds.
# Running the algorithm multiple times helps in avoiding it getting stuck on local minima, but instead finding the global minimum (i.e., the best solution).
# Random State = 1: setting the same seed value will ensure the same results from the K-means model on each run.
```

Figure 3.2 : K-Means Cluster Mapping



t-SNE Visualisation: t-SNE (t-distributed Stochastic Neighbour Embedding) visualises clusters on a 2D Map. It is a dimensionality reduction technique that can visualise the clusters (high-dimensional data) in a lower-dimensional space, whilst still preserving data structure. Even though the input of the clustering model (X) has three RFM dimensions, t-SNE reduces this to two dimensions (t-SNE Dimension 1 and t-SNE Dimension 2).

Figure 3.3: Snake-Plot of K-Means Clustering



Four clusters is optimal for customer segmentation.

Figure 3.4: RFM Values across K-means Clusters

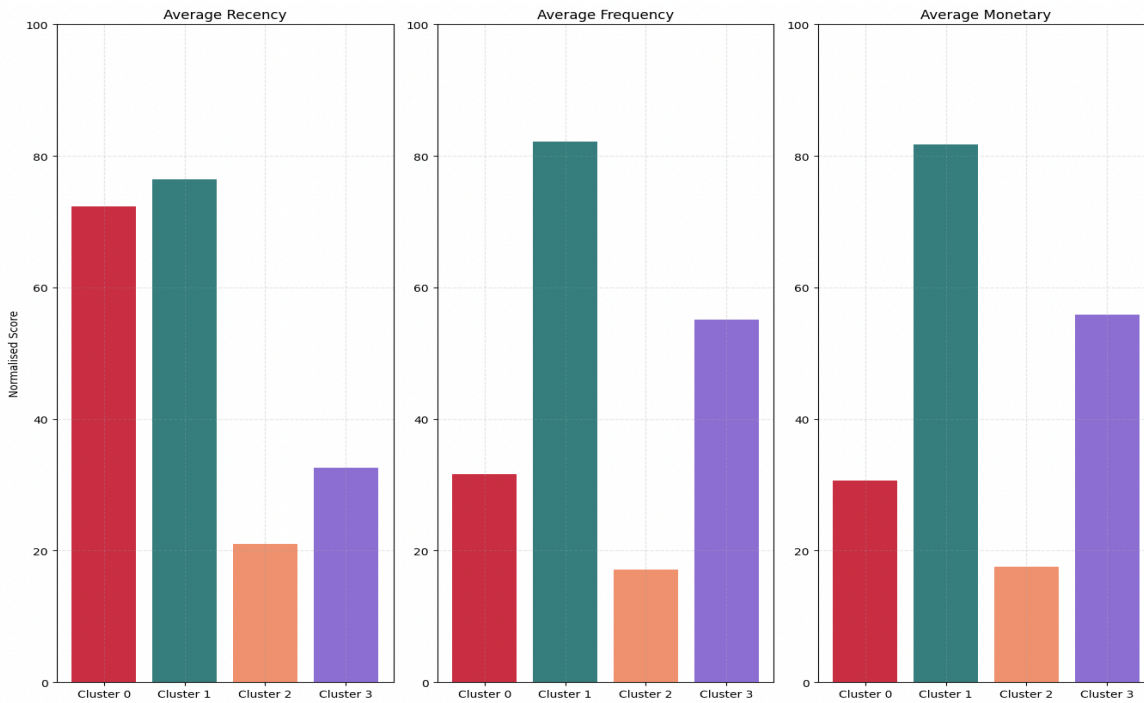


Figure 3.5: Customer Distribution across K-Means Clusters

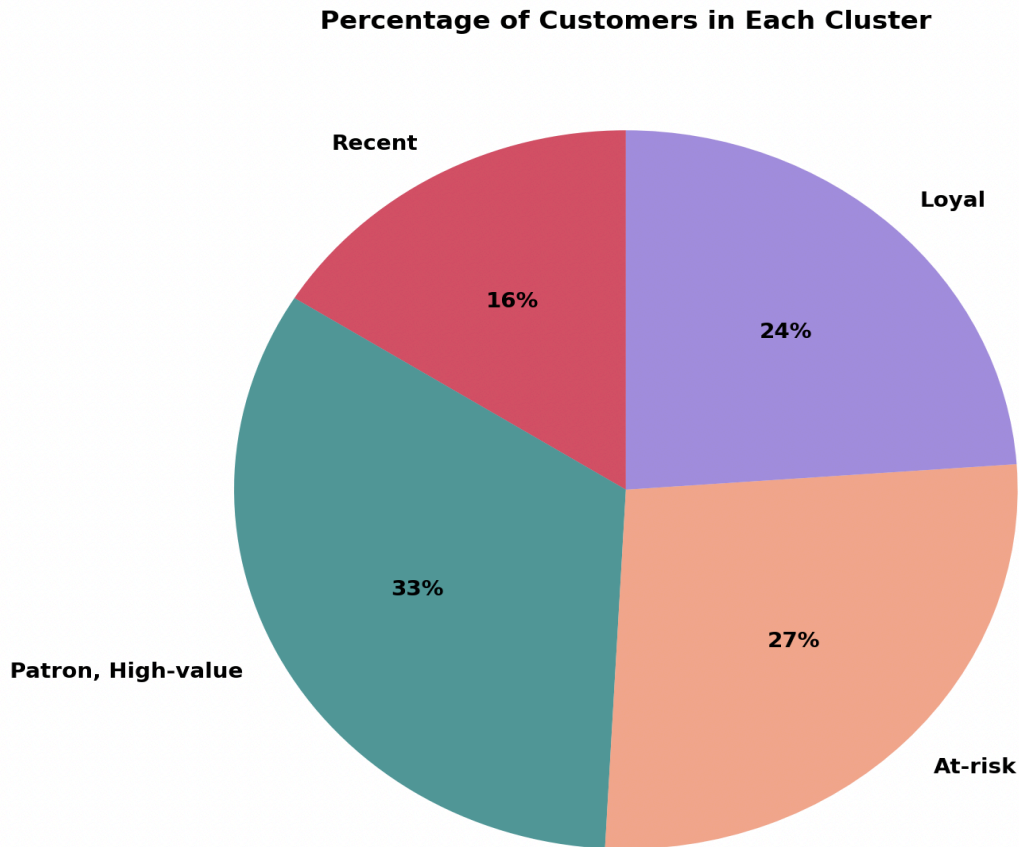
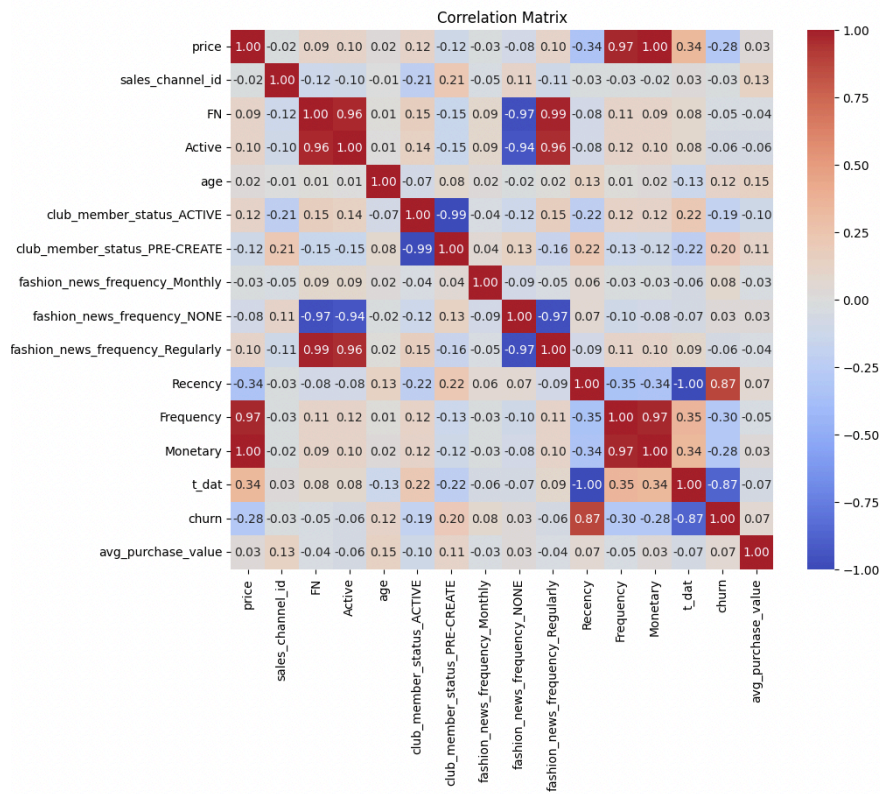


Figure 4.1: Correlation Matrix of Predictors and Target Variable



Removed predictors:

- *t_date*: High correlation with *Recency* and *Churn*
- *Recency*: High correlation with *Churn*. Due to the causality of the correlation (recency and thus transaction date is used as a proxy churn definition), both recency and transaction date must be excluded from the model.
- *Active*: High correlation with *FN*.
- *Price*: High correlation with *Frequency* and *Monetary*. Price is already reflected in the *Monetary* and *Average Purchase Value*.
- 'FN', 'fashion_news_frequency_Regularly', 'fashion_news_frequency_NONE' and 'Monetary' are also removed as such variables have VIFs over the 10 threshold.

Figure 4.2: Feature Importance in Churn Modelling

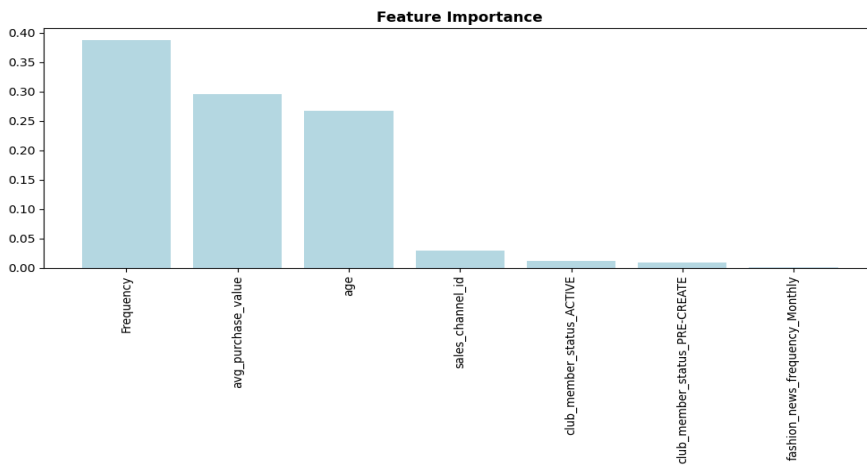


Figure 4.3: Determining the best baseline Model
(see next page)

```

from sklearn.pipeline import Pipeline
from sklearn.feature_selection import SelectKBest
from sklearn.model_selection import cross_val_predict

# Initialise an empty list to store the ROC AUC scores
model_scores = []

# Initialise an empty dictionary to store the predictions
predictions = {}

# Define list of classifiers with probability=True
classifiers = [
    KNeighborsClassifier(),
    SVC(random_state=123, probability=True),
    DecisionTreeClassifier(random_state=123),
    RandomForestClassifier(random_state=123),
    AdaBoostClassifier(random_state=123),
    GradientBoostingClassifier(random_state=123),
    LogisticRegression(random_state=123)
]

# Define classifier names in a list
classifier_names = [
    'KNeighborsClassifier',
    'SVC',
    'DecisionTreeClassifier',
    'RandomForestClassifier',
    'AdaBoostClassifier',
    'GradientBoostingClassifier',
    'LogisticRegressionClassifier'
]

# Loop through the classifiers
for classifier, name in zip(classifiers, classifier_names):
    pipe = Pipeline(steps=[
        ('selector', SelectKBest(k=len(X_train.columns))),
        ('classifier', classifier))

    # define the pipeline steps
    # Select the KBest features, first all of the features are selected until hyperparameter tuning later
    # Classify the target variable, iterating through each classifier model

    # Perform cross-validation (CV)
    y_pred = cross_val_predict(pipe, X_train, y_train, cv=10, method='predict_proba') # CV uses k-fold cross-validation and returns the predicted probabilities of the positive class (class 1) for each sample (y_pred)

    # Store the predictions
    predictions[name] = y_pred

    # Compute ROC curve and Area under Curve (AUC) for each classifier
    fpr, tpr, _ = roc_curve(y_train, y_pred[:, 1])
    roc_auc = auc(fpr, tpr)

    # Compute False Positive Rate (fpr) and True Positive Rate (tpr)
    # Compute AUC for each ROC curve

    # Plot ROC curve
    plt.figure()
    plt.plot(fpr, tpr, lw=2, label=f'{name} (AUC = {roc_auc:.2f})')
    plt.plot([0, 1], [0, 1], color='red', lw=2, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(f'ROC Curve - {name}')
    plt.legend(loc="lower right")
    plt.show()

    # Print legend label as the classifier name, and the AUC score to two decimal places
    # Print the random classifier diagonal line
    # Set axis limits

    # Store the AUC scores in the empty list created above
    model_scores.append(roc_auc)

```

Figure 4.4.1: Hyperparameter Tuning of the Random Forest Classifier I

```

:] # Build pipeline with best classifier only - Random Forest

# Define the pipeline
pipe = Pipeline([
    ('selector', SelectKBest()), # Feature selection using SelectKBest
    ('classifier', RandomForestClassifier(random_state=123)) # Random Forest classifier
])

# Define the range of k values for feature selection
k_range = [1, 2, 3, 4, 5, 6, 7] # Ensure it doesn't exceed the number of predictor features (7)

# Define the grid of hyperparameters to search over
param_grid = {
    "selector_k": k_range,
    "classifier_n_estimators": [50, 100, 200], # Number of trees in the forest
    "classifier_max_depth": [None, 10, 20], # Maximum depth of the trees. Aim is to increase accuracy, without overfitting the data.
    "classifier_min_samples_split": [2, 5, 10], # Minimum number of samples required to split a node
    "classifier_min_samples_leaf": [1, 2, 4] # Minimum number of samples required at each leaf node
}

# Perform grid search cross-validation
grid_search = GridSearchCV(estimator=pipe, param_grid=param_grid, scoring='recall', cv=5, n_jobs=-1) # Scores will be based on Recall. Cross validation will run in 5 different splits
grid_search.fit(X_train, y_train)

# Print the best hyperparameters and best score
print("Best Hyperparameters:", grid_search.best_params_)
print("Best ROC AUC Score:", grid_search.best_score_)
print("Best Estimator:", grid_search.best_estimator_)

Best Hyperparameters: {'classifier_max_depth': 10, 'classifier_min_samples_leaf': 4, 'classifier_min_samples_split': 2, 'classifier_n_estimators': 50, 'selector_k': 6}
Best ROC AUC Score: 0.8054945054945055
Best Estimator: Pipeline(steps=[('selector', SelectKBest(k=6)),
                                ('classifier',
                                 RandomForestClassifier(max_depth=10, min_samples_leaf=4,
                                                         n_estimators=50, random_state=123))])

```

This model takes the top six best predictive features (selector_k = 6).

Figure 4.4.2: Hyperparameter Tuning of the Random Forest Classifier II

```
# Add more hyperparameters to tune, in order to reduce overfitting the data

# import modules
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import randint

# Define the pipeline with fixed k=6
pipe = Pipeline([
    ('selector', SelectKBest(k=6)),          # Feature selection using SelectKBest with k=6, as previously defined
    ('classifier', RandomForestClassifier(random_state=123)) # Random Forest classifier
])

# Define the grid of hyperparameters to search over using distributions
param_dist = {
    "classifier__n_estimators": randint(50, 200), # Number of trees in the forest
    "classifier__max_depth": [None, 10, 20], # Maximum depth of the trees
    "classifier__min_samples_split": randint(2, 10), # Minimum number of samples required to split a node
    "classifier__min_samples_leaf": randint(1, 4), # Minimum number of samples required at each leaf node
    "classifier__max_features": ['auto', 'sqrt', 'log2'], # Number of features to consider when looking for the best split
    "classifier__bootstrap": [True, False] # Whether bootstrap samples are used when building trees
}

# Perform randomised search cross-validation, optimizing for recall          # 3 CV fold and randomised search speeds up the runtime of the cell
random_search = RandomizedSearchCV(estimator=pipe, param_distributions=param_dist, scoring='recall', n_iter=50, cv=3, n_jobs=-1, random_state=123)
random_search.fit(X_train, y_train)

# Print the best hyperparameters and best score
print("Best Hyperparameters:", random_search.best_params_)
print("Best Recall Score:", random_search.best_score_)
print("Best Estimator:", random_search.best_estimator_)
```

Figure 4.5: Hyperparameter Tuned RF Classifier - Robustness Checks for Overfitting

```
# Refitting the training data with the best parameters obtained from randomised search
best_pipe = random_search.best_estimator_

# Fit the pipeline on the training data
best_pipe.fit(X_train, y_train)

# Predict the target variable on the test data
y_test_pred = best_pipe.predict(X_test)

# Calculate evaluation metrics on test data
accuracy = accuracy_score(y_test, y_test_pred)
precision = precision_score(y_test, y_test_pred)
recall = recall_score(y_test, y_test_pred)
f1 = f1_score(y_test, y_test_pred)

# Calculate probabilities for the positive class (churners)
y_test_prob = best_pipe.predict_proba(X_test)[:, 1]

# Calculate ROC curve
fpr, tpr, thresholds = roc_curve(y_test, y_test_prob)

# Calculate AUC
auc = roc_auc_score(y_test, y_test_prob)

# Print evaluation metrics for test data to four decimal places
print("Evaluation Metrics:")
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
print(f"AUC: {auc:.4f}")

# Calculate evaluation metrics on the training data
y_train_pred = best_pipe.predict(X_train)
train_accuracy = accuracy_score(y_train, y_train_pred)
train_precision = precision_score(y_train, y_train_pred)
train_recall = recall_score(y_train, y_train_pred)
train_f1 = f1_score(y_train, y_train_pred)
train_auc = roc_auc_score(y_train, best_pipe.predict_proba(X_train)[:, 1])

# Print evaluation metrics for training data (to four decimal places)
print("\nTraining Data Evaluation Metrics:")
print(f"Accuracy: {train_accuracy:.4f}")
print(f"Precision: {train_precision:.4f}")
print(f"Recall: {train_recall:.4f}")
print(f"F1-score: {train_f1:.4f}")
print(f"AUC: {train_auc:.4f}")

# Compare with test data metrics
print("\nTest Data Evaluation Metrics:")
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
print(f"AUC: {auc:.4f}")

# Assess overfitting by comparing metrics
if (train_recall - test_recall > 0.2) or (train_auc - test_auc > 0.2):
    print("\nThe model might be overfitting. Consider regularisation or further tuning.")
else:
    print("\nThe model generalizes well between training and test data.")

# Second check -- more moderate tolerance to overfitting.
# Both recall and AUC scores are compared between the training and test sets. If the difference in recall or AUC exceeds 0.2 (20 percentage points), it indicates potential overfitting.
# The 0.2 threshold is a good starting point that signals a notable performance drop between the training and test data, suggesting overfitting.
```

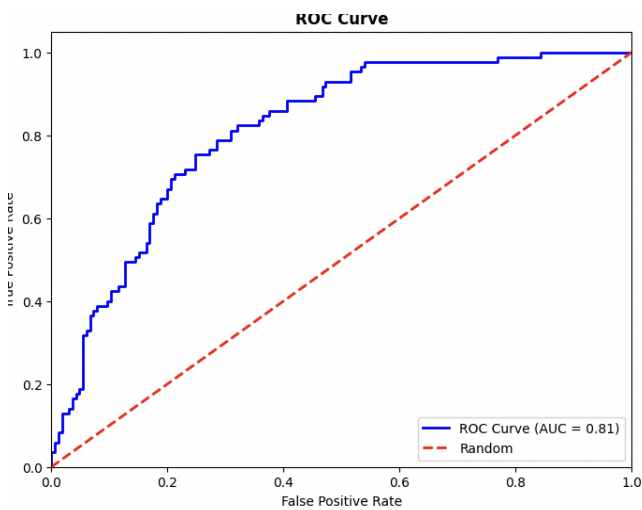
Figure 4.6: Hyperparameter Tuned RF Classifier - Comparison of Evaluation Metrics across Training and Test Data

Training Data Evaluation Metrics:
 Accuracy: 0.8632
 Precision: 0.8230
 Recall: 0.9256
 F1-score: 0.8713
 AUC: 0.9616

Test Data Evaluation Metrics:
 Accuracy: 0.7360
 Precision: 0.5888
 Recall: 0.7412
 F1-score: 0.6562
 AUC: 0.8157

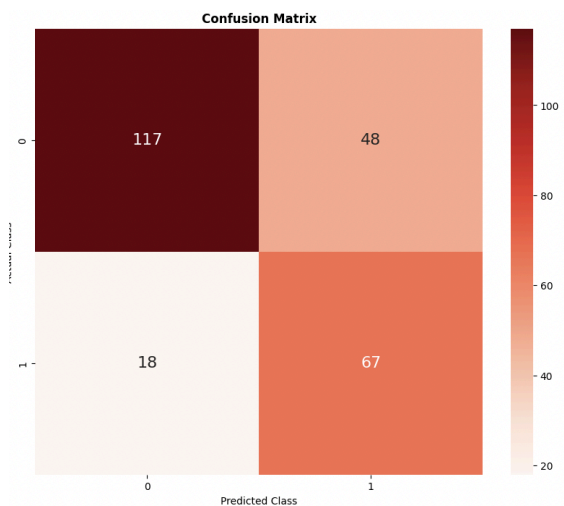
The model generalizes well between training and test data.

Figure 4.7: Hyperparameter Tuned RF Classifier ROC Curve and AUC Score



Recall is defined as the true positive rate. It is the rate at which the model correctly identifies customers that are going to churn. The true positive rate approaches 1 at a prediction threshold of 0.6.

Figure 4.8: Hyperparameter Tuned RF Classifier Confusion Matrix



TN: 117	FP: 48
FN: 18	TP: 67

TN	Non-churning customers who were correctly predicted.
TP	Churning customers who were correctly predicted.
FN	Churning customers who were incorrectly predicted as non-churning.
FP	Non-churning customers who were incorrectly predicted as churning.

Figure 5.1: Dataset for Customer Churn RapidMiner AutoModel

customer_id	price	sales_channel_id	FN	Active	age	club_member_status_ACTIVE	club_member_status_PRE_CREATE	fashion_news_frequency_Monthly	fashion_news_frequency_NONE	fashion_news_frequency_Regularly	Recency	Frequency	Monetary	t_dat	churn	avg_purchase_value	
b79cb106c7d810b3cd9faa7e8b...	241.83		1	1	1	20	1	0	0	0	1	144	7	241.83	2020-05-01	No	34.547143
e06eb4f1121a59236773e4ab90...	3450.03		0	0	0	49	1	0	0	0	0	0	113	3450.03	2020-09-22	No	30.531239
d69a7a0abd33a5d11102d894fb...	235.41		0	0	0	41	1	0	0	0	0	635	12	235.41	2018-12-27	Yes	19.617500
6e5476b9b6f1e3258b55c61c2...	25.37		1	0	0	41	0	1	0	0	0	630	3	25.37	2019-01-01	Yes	8.456667
i510906a9b735d6f0b0c54b7c2...	163.30		0	0	0	42	1	0	0	0	0	613	10	163.30	2019-01-18	Yes	18.330000

Figure 5.2: Feature Selection on RapidMiner AutoModel

Selected	Status ↑	Quality	Name	Correlation	ID-ness	Stability	Missing	Text-ness
<input type="checkbox"/>	●		customer_id	0.09%	100.00%	0.10%	0.00%	61.78%
<input type="checkbox"/>	●		club_member_status_AC...	3.62%	0.20%	92.18%	0.00%	0.00%
<input type="checkbox"/>	●		club_member_status_PRE...	3.93%	0.20%	92.38%	0.00%	0.00%
<input type="checkbox"/>	●		fashion_news_frequency_...	0.67%	0.20%	99.60%	0.00%	0.00%
<input type="checkbox"/>	●		Recency	76.52%	46.34%	1.30%	0.00%	0.00%
<input type="checkbox"/>	●		t_dat	76.53%	?	1.30%	0.00%	0.00%

9 out of a total of 16 predictors selected for the model. The following were removed:

- Customer_id: High identification.
- club_member_status_ACTIVE: high stability is not helpful for pattern identification.
- club_member_status_PRE_CREATE: high stability is not helpful for pattern identification.
- fashion_news_frequency_MONTHLY: high stability is not helpful for pattern identification.
- Recency and t_dat due to high correlation with the target variable

As Recency is used to define churn, it is removed from the predictor features. This avoids direct data leakage from the target variable into the predictors, and helps to build a more robust and generalised churn model. Transaction date is also removed.

Figure 5.3: Feature Importance of the best RapidMiner Model - Logistic Regression

Attribute	Weight
Frequency	0.495
price	0.185
Monetary	0.089
age	0.047
fashion_news_frequency_NONE	0.039
FN	0.021
sales_channel_id	0.010
Active	0.007
fashion_news_frequency_Regularly	0.005

Frequency, price and monetary are of top feature importance. However, the RapidMiner automodel may not have picked up on the high VIFs of these variables, which is something that must be refined.

Figure 5.4: ROC Curve Comparison of RapidMiner AutoModels

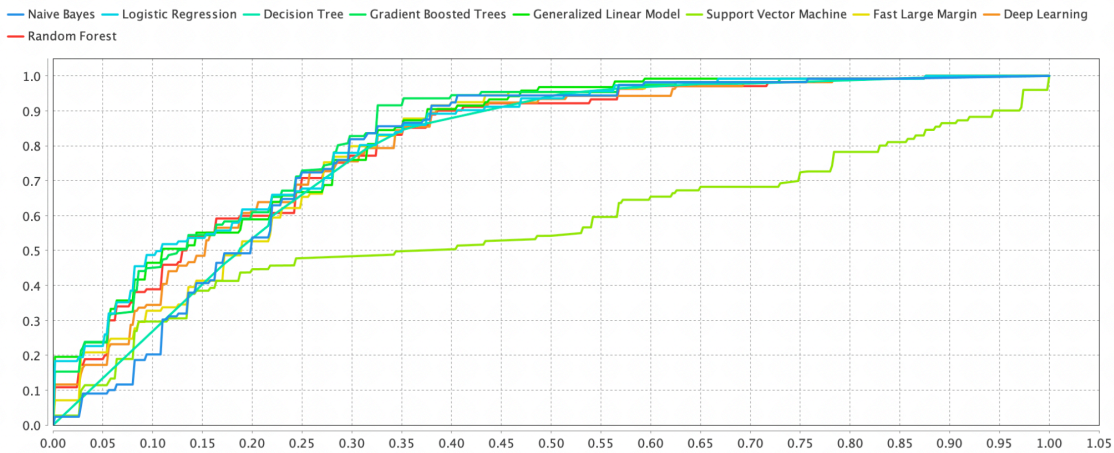


Figure 5.5: Performance Evaluation of best RapidMiner Model - Logistic Regression

Criterion	Value	Standard Deviation
Accuracy	71.9%	± 3.3%
Classification Error	28.1%	± 3.3%
AUC	82.1%	± 3.7%
Precision	60.2%	± 6.5%
Recall	79.2%	± 5.8%
F Measure	68.0%	± 2.7%
Sensitivity	79.2%	± 5.8%
Specificity	67.9%	± 6.9%

Figure 5.6: Confusion Matrix of best RapidMiner Model: Logistic Regression

Confusion Matrix

	true No	true Yes	class precision
pred. No	120	23	83.92%
pred. Yes	57	85	59.86%
class recall	67.80%	78.70%	

Management Progress Report

1. Project Preparation (Weeks 1 & 2)

- Finalise project scope, objectives and chosen dataset with supervisor
- Confirm access to necessary software and platforms (Spyder, R-studio, RapidMiner and Tableau)

2. Data Preparation (Weeks 3 & 4)

- EDA will be performed to gain a deeper understanding of the dataset
- Apply data pre-processing techniques to normalise and encode data (if needed) to ensure high data quality

3. Model Building (Weeks 5-8)

- Build models on chosen software
- Fine-tune the models using cross-validation techniques and further suitable feature engineering
- Experiment with different algorithms to determine the most successful approach to reach project objectives

4. Model Interpretation (Weeks 9-11)

- Interpret model results and evaluate their effectiveness in achieving project objectives
- Create visualisations (if appropriate) to effectively communicate key insights
- Prepare report to include findings and recommendations, in line with project guidelines
- Document all of these above steps, including challenges faced and effective solutions
- Incorporate feedback gained from supervisor to improve final results

5. Evaluation & Reflection (Week 12 & 13)

- Evaluate the project outcomes against the initial project objectives, alongside own strengths and weaknesses
- Identify aspects to improve
- Prepare and rehearse final presentation
- Submit the final project and deliver presentation