Project Report

Data Mining Project: Customer Churn Prediction

Group 13: Dexiang Cao, Melvin Cubrovic, Anna Sergeeva, Nik Yakovlev, Jianguo Zhang

December 8, 2024

1 Application Area, Goals, and Business Understanding

In subscription-based industries, customer churn prediction is vital, particularly in telecommunications, where competition is intense, and customer loyalty is difficult to secure. Customer churn, defined as the discontinuation of a service, leads to significant revenue loss and increased costs for new customer acquisition. Therefore, telecom companies benefit greatly from predictive models that can identify customers at risk of churning, allowing them to implement targeted retention strategies like personalized offers, discounts, or enhanced customer service.

The primary objective of this project is to develop a machine learning model that accurately forecasts customer churn. To achieve this, we will compare several models—Logistic Regression, Decison Tree, Random Forest, and Naive Bayes—to identify the one with the highest predictive accuracy.

Additionally, an experimental clustering will be applied to segment customers based on shared attributes and behaviors, allowing for more tailored retention efforts across different customer profiles.

From a business perspective, this project addresses the problem of identifying customers likely to discontinue their subscription in a timely manner. The target variable in this project is the "churn label," a binary variable indicating whether a customer will churn ("yes") or stay ("no"). This is fundamentally a classification problem, as the goal is to classify customers into one of two categories based on their likelihood of churning.

2 Profiling

For this analysis, we utilized the publicly available "Telco Customer Churn (11.1.3+)" dataset [1], hosted on Kaggle and originally provided by IBM blogs. It includes 50 features across 7043 observations, providing a sufficient foundation for analysis. These features span in three cathegories: customer demographics ("Age"), contractual information ("Internet type"), and service usage metrics ("Average monthly GB downloads"). The dataset consists of text (2), categorical (12), numeric (18), and boolean (18) variable types.

Exploratory data analysis (EDA) was performed to ensure the data was suitable for analysis. The dataset contains 15,751 missing values, accounting for 4.5% of all entries, spread across four columns: Offer with 3877 (55.0%), Internet Type with 1526 (21.7%), Churn Category with 5174 (73.5%), and Churn Reason with 5174 missing cells (73.5%). There are no duplicate rows, confirming each observation corresponds to a unique customer record.

The target variable selected for this analysis is "Churn Label," a boolean variable indicating whether a customer has left the company ("True") or stayed ("False"). The dataset shows a churn imbalance, with 5174 customers (73.5%) remaining and 1869 (26.5%) having churned (see 1). This imbalance highlights that while the telecom company retains the majority of its customers, approximately one in four customers leaves the company in a given quarter. This indicates the need for targeted churn reduction

strategies.

Churn is observed across all age groups (19-80 years). However, there are fewer customers over the age of 65, suggesting that older individuals are less likely to use the services. Further, churn is evenly distributed between genders, indicating no difference in churn likelihood between males and females (see 1).

The satisfaction score, a customer's rating of their overall experience with the company on a scale of 1 (Very Dissatisfied) to 5 (Very Satisfied), is strongly correlated with churn (correlation = 0.859). Customers with scores of 1 or 2 are most likely to churn, while those with scores of 4 or 5 are predominantly retained. For customers with a satisfaction score of 3, the churn likelihood is not clear, though the majority tend to stay. These findings highlight satisfaction score as a critical factor in predicting churn (see 1).

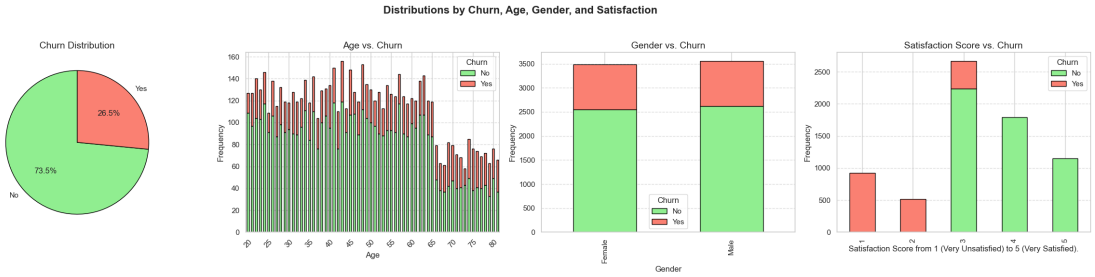

Figure 1: Data Profiling Plots: Distribution of Churn and No Churn on Age, Gender and Satisfaction Score

3 Preprocessing

To ensure the dataset was properly prepared for machine learning analysis, a series of systematic preprocessing steps were performed to address missing values, transform features, and optimize the dataset for model compatibility and interpretability.

The Customer ID column, being 100% unique across all entries, was removed as it provides no predictive value. Similarly, the Quarter column, containing only values for "Q3," was removed due to the lack of variability. Features such as Under 30 and Senior Citizen were excluded because they correlate directly with Age and thus offer redundant information. Additionally, State and Country were removed, as they contained only a single value ("California" and "USA," respectively) and provided no variation for model training.

Several features that directly or indirectly indicate churn, including Churn Score, Churn Category, Churn Reason, and Customer Status, were removed to avoid redundancy and potential data leakage. The Zip Code and City columns were excluded because Latitude and Longitude were deemed sufficient substitutes for geographic information. The Offer column was also removed due to its high correlation with Contract and the fact that over 50% of its values were missing.

Further, the Referred a Friend feature was dropped as the Number of Referrals column

already captures whether a customer was referred and how many referrals were made. Similarly, Internet Service was removed because its information is already represented in the Internet Type column, making it redundant.

To address missing values in the Internet Type column, all None values were replaced with the categorical label "No Internet," providing a meaningful representation for customers without internet services. Additionally, numerical features such as Total Extra Data Charges and Total Refunds were converted into boolean indicators. The new columns, Has Extra Data Charges and Has Refunds, represented whether a customer incurred extra data charges or received refunds. Values greater than zero were assigned as True (1), and zero values were marked as False (0). The original numerical columns were subsequently dropped, as their binary representation was deemed sufficient for analysis.

Binary categorical features, such as those with Yes/No or Male/Female values, were encoded into numeric representations. Specifically, "Yes" and "Male" were mapped to 1, while "No" and "Female" were mapped to 0, ensuring the model could process these variables without interpreting them as ordinal features. For multi-category variables, including Internet Type, Contract, and Payment Method, one-hot encoding was applied. This transformation created separate binary columns for each category, allowing the model to interpret categorical variables without assuming an ordinal relationship. The original columns were removed after the one-hot encoding was applied.

Finally, the dataset was split into features (X) and the target label (y). The target label, Churn Label, represented whether a customer had churned (1) or not (0), while the feature set consisted of all other processed columns. This structured dataset provided a clean and well-defined input for subsequent machine learning analysis.

By handling missing values, optimizing feature representation, and eliminating redundancy, the preprocessing pipeline ensured that the dataset was clean, consistent, and tailored for efficient and accurate churn prediction. These steps minimized redundancy and improved the interpretability of the resulting models, providing a robust foundation for predictive analysis.

Finally, we introduce the 35 remaining variables we will use to train our models: 'Satisfaction Score', 'Number of Referrals', 'Monthly Charge', 'Tenure in Months', 'Online Security', 'Internet Type', 'Dependents', 'Contract', 'Payment Method', 'Gender', 'Age', 'Married', 'Number of Dependents', 'Latitude', 'Longitude', 'Population', 'Phone Service', 'Avg Monthly Long Distance Charges', 'Multiple Lines', 'Avg Monthly GB Download', 'Online Backup', 'Device Protection Plan', 'Premium Tech Support', 'Streaming TV', 'Streaming Movies', 'Streaming Music', 'Unlimited Data', 'Paperless Billing', 'Total Charges', 'Has Refunds', 'Has Extra Data Charges', 'Total Long Distance Charges', 'Total Revenue', 'CLTV'.

4 Baseline

As a starting point for our analysis, we implemented a baseline model to evaluate churn prediction using a simple, rule-based approach. The baseline leverages the feature with the highest correlation to the target variable Churn Label, which was identified as the

Satisfaction Score. This approach is designed to provide a straightforward benchmark against which more complex models can be compared.

The rule-based logic predicts that a customer will churn (Predicted Churn = 1) if their Satisfaction Score is less than 3. This decision rule is based on the assumption that low satisfaction is a strong indicator of churn. The model's predictions were then compared to the true Churn Label values in the dataset.

After applying this rule, the performance of the baseline model was evaluated using a classification report and a confusion matrix. The classification report 1 provides detailed metrics, including precision, recall, and F1-score, for both the "Churn" and "No Churn" classes, as well as the overall model accuracy.

Table 1: Classification Report for Baseline Model

Class	**Precision**	**Recall**	**F1-Score**	**Support**
No Churn (0)	0.92	1.00	0.96	5174
Churn (1)	1.00	0.77	0.87	1869
Accuracy		0.94		7043
Macro Avg	0.96	0.89	0.92	7043
Weighted Avg	0.94	0.94	0.94	7043

5 Training the Models

5.1 Logistic Regression

After completing the data preprocessing steps, which were described earlier, we proceeded with data splitting, hyperparameter optimization and model training.

The dataset was divided into features (X) and labels (y), with an 80-20 train-test split applied to ensure that the model's performance could be evaluated on unseen data. To address differences in feature scales, a MinMaxScaler was employed separately on the training and testing sets. This scaling procedure normalized all feature values to a range between 0 and 1, thereby facilitating efficient optimization during logistic regression.

To optimize the logistic regression model, we implemented nested cross-validation combined with a random search for hyperparameter tuning. Nested cross-validation, a robust evaluation framework, isolates the data used for hyperparameter tuning from the data used for model evaluation, thereby preventing overfitting and ensuring unbiased performance estimates. This process involved two levels of cross-validation. The inner cross-validation loop performed a randomized search over a predefined set of hyperparameters using five stratified folds. The hyperparameters considered included the penalty type (L2 regularization), the inverse regularization strength (C, sampled from a uniform distribution between 0.01 and 10), the optimization solver (liblinear and lbfgs), and the maximum number of iterations (sampled from a range between 100 and 5000). RandomizedSearchCV was configured to explore 50 random combinations of these hyperparameters. Two scoring metrics were used for optimization: accuracy and recall for

the "Churn" class. Accuracy was used to evaluate the overall performance of the model across both classes, while recall for "Churn" emphasized the model's ability to correctly identify customers at risk of leaving.

The outer cross-validation loop, also using five stratified folds, evaluated the generalization performance of the model with the best hyperparameters identified from the inner loop. The mean of the respective scoring metric (accuracy or churn recall) from the outer loop were reported as the nested cross-validation result. Following hyperparameter tuning, the final logistic regression model was trained on the entire training set using the optimal hyperparameters determined by RandomizedSearchCV. This model was subsequently evaluated on the test set.

5.2 Naive Bayes

Both the Multinomial Naive Bayes (MultinomialNB) and Gaussian Naive Bayes (GaussianNB) variants were implemented to assess their effectiveness in the churn prediction task. The dataset was first split into features (X) and the target variable (y), followed by a train-test split with an 80-20 ratio. To prepare the features for the Naive Bayes models, normalization was applied using a MinMaxScaler to scale feature values between 0 and 1. This scaling step is critical for MultinomialNB, which relies on feature values being positive. For MultinomialNB, a random search was performed over a hyperparameter grid using nested cross-validation. The key hyperparameter optimized was the smoothing parameter alpha, sampled from a uniform distribution between 0.01 and 1.0. RandomizedSearchCV was used within the inner cross-validation loop to identify the optimal value of alpha. GaussianNB, on the other hand, does not require hyperparameter tuning, as its implementation is parameter-free.

The models were evaluated using accuracy and recall for the 'Churn' class as the scoring metric for the outer cross-validation loops. The nested cross-validation process for MultinomialNB reported a mean accuracy. Similarly, GaussianNB's performance was assessed using just the outer cross-validation.

5.3 Decision Tree

In the Decision Tree approach, we implemented a supervised machine learning model that learns a tree-like structure based on the input features to make predictions about churn. To ensure robust model evaluation and optimal hyperparameter tuning, we utilized nested cross-validation.

The dataset was split into training and testing subsets using an 80-20 stratified split. The Decision Tree classifier was configured with a hyperparameter grid that included the criterion (gini or entropy), maximum tree depth (max_depth), and a threshold for splitting (min_impurity_decrease). To optimize these parameters, we employed a randomized search strategy with 50 iterations. The inner cross-validation loop, comprising 5 stratified folds, was used to select the best hyperparameters, while the outer cross-validation loop, also with 5 stratified folds, provided an unbiased estimate of model performance.

5.4 Random Forest

The Random Forest approach extended the Decision Tree model by aggregating multiple decision trees through a process called bagging (bootstrap aggregation). Each tree in the forest was trained on a random subset of the data, and the final prediction was made by majority voting. Random Forest improves over Decision Tree by reducing variance and increasing robustness to overfitting.

Similar to the Decision Tree approach, we utilized nested cross-validation for hyperparameter tuning and performance evaluation. The parameter grid for the Random Forest model included the number of estimators (n_estimators), criterion (gini or entropy), maximum tree depth (max_depth), and a threshold for splitting (min_impurity_decrease). Randomized search with 50 iterations was employed in the inner cross-validation loop to identify the best combination of hyperparameters. The outer cross-validation loop provided a reliable estimate of the model's generalization performance.

6 Clustering

Using the within-cluster sum of squares (WCSS) and the elbow method, we attempted to determine the optimal number of clusters for the original dataset. However, the results were contradictory: the elbow method suggested the best number of clusters was 1, while WCSS indicated 3 clusters as the optimal choice. After weighing both approaches, we chose to proceed with 3 clusters (see 2).

Additionally, considering the high dimensionality of the original dataset (48 features), we decided to apply Principal Component Analysis (PCA) to reduce the dimensionality. This allowed us to condense the 48 features into 9 principal components, with PC1 and PC2 capturing the most variance from the original data. However, when clustering the reduced data, the results were not satisfactory, as many clusters were heavily mixed. This suggested that the original dataset contained a significant amount of noise (see 2).

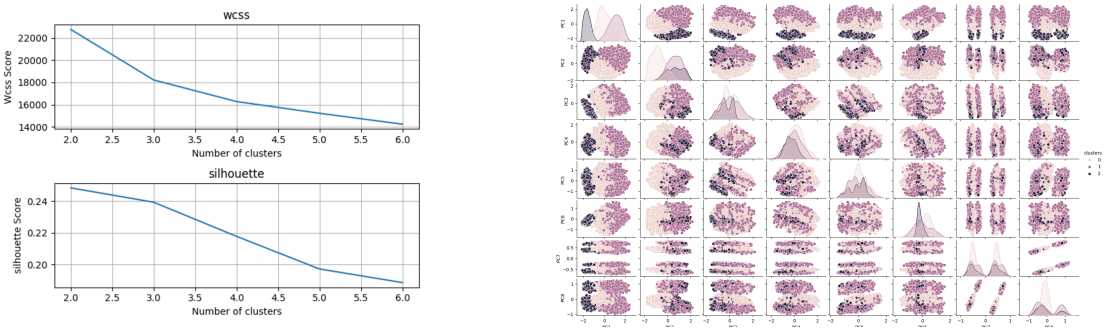

Figure 2: WCSS-Plot, Silhouette-Plot and PCA-Cluster-Plot

To address this, we applied Singular Value Decomposition (SVD) for noise reduction. We decomposed the original matrix into three matrices: the left singular vectors matrix, the singular value matrix, and the right singular vectors matrix. By selecting the top K

singular vectors, we reconstructed the original matrix using the first K singular vectors and subtracted the reconstruction from the original matrix to obtain the residual matrix. We then introduced random factors by multiplying the residual matrix by either +1 or -1, resulting in a noisy matrix. We computed the Frobenius norm of the residual and noisy matrices. If the Frobenius norms of both matrices were equal, it meant the residual matrix was purely noise, and the reconstructed matrix was a denoised version. We plotted the differences as a function of the residual and noisy matrices and determined that the optimal value of K was 12. This indicated that using the first 12 singular vectors to reconstruct the original matrix provided the best denoised dataset (see 3).

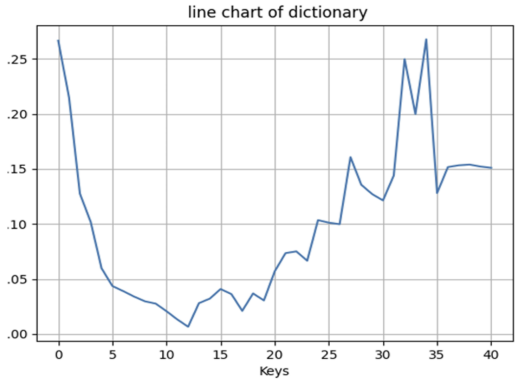

Figure 3: Residual and Noisy Matrix Differences vs. Number of Singular Vectors (K)

(Note that the noise reduction was applied only to the training dataset and did not include the test dataset.) Afterward, we performed clustering again using the denoised dataset and applied both WCSS and the elbow method to determine the optimal number of clusters. This time, both methods pointed to 3 clusters as the best choice. Subsequently, we performed PCA on the denoised data, where the clustering results on PC1 and PC2 were excellent, with very clear classification. This confirmed that the dataset had been significantly optimized (see 4).

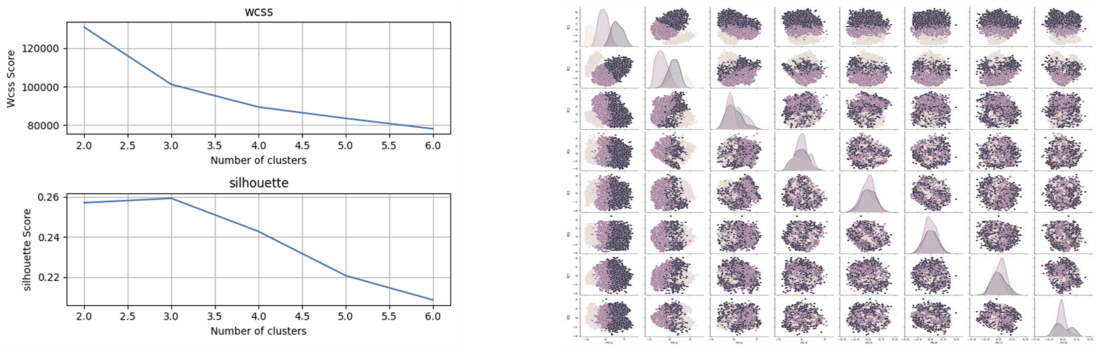

Figure 4: WCSS-Plot, Silhouette-Plot and PCA-Cluster-Plot after noise reduction

7 Results

7.1 Evaluation Metrics

In this analysis, we aim to evaluate model performance using metrics that align with both the overall prediction quality and the business goal of identifying customers likely to churn.

For the overall evaluation of our models, accuracy serves as a primary metric. Accuracy measures the proportion of correctly classified instances (both churned and non-churned customers) out of the test dataset. This metric provides a holistic view of model performance and ensures that both classes are considered during evaluation. High accuracy indicates that the model performs well in classifying the majority class (non-churned customers) as well as the minority class (churned customers).

Given that the business objective of churn prediction is to identify customers at risk of leaving, recall for the churned class (Churn = 1) is of particular importance. Recall measures the proportion of actual churned customers that the model correctly identifies. A high recall for the churn class ensures that most at-risk customers are flagged, enabling targeted interventions to retain these customers. This focus on recall directly supports the business goal of minimizing customer attrition and prioritizing resources for retention strategies.

It is particularly crucial to emphasize recall for the churned class due to the imbalanced nature of the target variable distribution, where 26.5% of the customers churn, and 73.5% do not. In imbalanced datasets, accuracy alone can be misleading, as a model that predicts the majority class (no churn) with high frequency can achieve a seemingly good accuracy score without effectively identifying churned customers. By incorporating recall as a key metric, we ensure that the model is evaluated on its ability to identify the minority class, addressing the imbalance in the target variable and aligning the analysis with the business objective.

7.2 Best Models

The following section presents the best model identified through hyperparameter tuning for each algorithm we tested. Both accuracy and recall for 'Churn' were used as scoring metrics in the RandomizedSearchCV. From the tested grid, the model that achieved the best accuracy also yielded the highest recall for 'Churn.' Therefore, we list only one model, even though two separate runs were performed with varying scoring metrics.

The best hyperparameters for each model were identified through hyperparameter optimization and are summarized as follows. For the **Decision Tree** model, the optimal parameters included the use of the `entropy` criterion, a maximum depth of 10, and a minimum impurity decrease of 0.0044. The **Random Forest** model also used the `entropy` criterion, with a maximum depth of 16, a minimum impurity decrease of 0.0047, and 75 estimators.

The **Multinomial Naive Bayes** model performed best with an alpha smoothing parameter of 0.3845. Finally, for **Logistic Regression**, the optimal parameters included a regularization strength (C) of 9.7776, a maximum iteration count of 2123, the `l2`

penalty, and the `liblinear` solver. These hyperparameters were determined through randomized cross-validation to maximize model performance on the given dataset.

7.3 Scores

To asses the performance of the different models we compared the resulting accuracy and recall of 'Churn' from the cross-validations explained in the training section.

Table 2: Accuracy and Recall for Each Model

Model	**Accuracy**	**Recall (Churn)**
Logistic Regression	**0.9560**	**0.8997**
Multinomial Naive Bayes	0.8486	0.5019
Gaussian Naive Bayes	0.8523	0.8700
Decision Tree	0.9526	0.8957
Random Forest	0.9476	0.8095
Baseline	0.9400	0.7700

The results presented in Table 2 highlight the performance of different machine learning models evaluated based on overall accuracy and recall for churn prediction. Logistic Regression achieved the highest accuracy (0.9560) and also exhibited strong recall for the churn class (0.8997), demonstrating its ability to balance general classification performance with a focus on identifying at-risk customers. This makes Logistic Regression particularly well-suited for the given business objective.

Among the Naive Bayes models, Gaussian Naive Bayes performed significantly better than Multinomial Naive Bayes in terms of both accuracy (0.8523) and recall (0.8700). The Multinomial Naive Bayes model struggled to correctly identify churned customers, achieving a recall of only 0.5019, which limits its utility for churn prediction despite its simplicity.

The Decision Tree model delivered competitive results, achieving an accuracy of 0.9526 and a recall of 0.8957. Its performance is close to that of Logistic Regression, making it another viable option for churn prediction while providing interpretability in decision-making processes.

The Random Forest model, while robust, did not outperform Logistic Regression or Decision Tree in this context. It achieved an accuracy of 0.9476 but fell short in recall (0.8095), suggesting that while it is good at overall classification, it may miss a significant number of churned customers, which is a critical focus of this analysis.

In conclusion, Logistic Regression stands out as the best-performing model in this evaluation due to its strong balance of accuracy and recall, while Decision Tree offers a competitive and interpretable alternative. Gaussian Naive Bayes also demonstrated reasonable performance, but the Random Forest model's lower recall and the underwhelming performance of Multinomial Naive Bayes suggest these models may be less effective for this specific business goal.

8 State of the Art Performance Comparison

In order to validate our model's performance, we conducted a State of the Art (SotA) comparison by examining the approaches documented in relevant Kaggle notebooks. Since Kaggle was the only platform for finding well-documented work on this dataset, we adopted a straightforward method for identifying the most endorsed notebooks by the Kaggle community—evaluating their upvotes. Furthermore, we specifically focused on notebooks that reported key performance metrics such as accuracy and recall for churn prediction, as these metrics are critical for assessing customer churn effectively and aligning with the objectives of our study. This approach resulted in filtering out notebooks that, despite having a high number of upvotes, did not provide the relevant metrics needed for our comparison.

Ultimately, we identified two notebooks that offered extensive results suitable for benchmarking. One employed the CatBoost algorithm for predicting customer churn, while the other utilized a variety of algorithms, with its best model being a stacked ensemble of these methods.

Table 3 summarizes the comparative performance metrics of our model alongside the top-performing models from those notebooks.

Table 3: Comparison of Churn Prediction Models

Model	**Accuracy**	**Recall (Churn)**
Logistic Regression	0.96	0.90
Notebook 1 (CatBoost) [2]	0.96	0.88
Notebook 2 (Stack)[1] [3]	0.91	0.85
Notebook 2 (Decision Tree) [3]	0.87	0.84

The comparison highlights that our best model, Logistic Regression, is superior, matching the accuracy of CatBoost in Notebook 1 while achieving a higher recall for predicting churn. In contrast, Notebook 2's stacking model and Decision Tree performed worse in both metrics, with our Decision Tree model even outperforming theirs. This confirms the robustness of our approach, particularly in achieving higher recall, which is crucial for better identification of potential churners.

[1] Combination of XGBoost, LightGBM, Random Forest, and Decision Tree classifiers.

References

[1] A. Terry, "Telco customer churn dataset," https://www.kaggle.com/datasets/alfathterry/telco-customer-churn-11-1-3, 2024, accessed: 2024-12-01. [Online]. Available: https://www.kaggle.com/datasets/alfathterry/telco-customer-churn-11-1-3

[2] ——, "Telco customer churn analysis," https://www.kaggle.com/code/alfathterry/telco-customer-churn-analysis/notebook, 2024.

[3] G. Tanmay, "Telco churn eda - cv score 85, f1 score 80," https://www.kaggle.com/code/tanmay111999/telco-churn-eda-cv-score-85-f1-score-80/notebook, 2024.

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Bachelor-, Master-, Seminar-, oder Projektarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und in der untenstehenden Tabelle angegebenen Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Declaration of Used AI Tools

Tool	Purpose	Where?	Useful?
ChatGPT	Rephrasing	Throughout	+
ChatGPT	Help with Coding	Throughout	++
Google	Search for Dataset and comparable Notebooks	Throughout	+

Mannheim, den 8. Dezember 2024