Glenn Dalbey

WGU D603

RFN1 Task 1

Classification Data Mining Models

***A.  Create your subgroup and project in GitLab using the provided web link by doing the following:***

***•   Clone the project to the IDE.***

***•   Commit with a message and push when you complete each requirement listed in parts D and E.***

***Note: You may commit and push whenever you want to back up your changes, even if a requirement is not yet complete.***

***•   Submit a copy of the GitLab repository URL in the "Comments to Evaluator" section when you submit this assessment.***

***•   Submit a copy of the repository branch history retrieved from your repository, which must include the commit messages and dates.***

***B.  Describe the purpose of this data mining report by doing the following:***

***1. Propose one question relevant to a real-world organizational situation that you will answer using one of the following classification methods:***

***•   Random forest***

***•   AdaBoost***

***•   Gradient boost***

Is it possible to predict which customers have the highest risk of churn based on three factors? Service usage patterns, contract details, and demographic information?

***2. Define one goal of the data analysis. Ensure that your goal is reasonable within the scenario's scope and represented in the available data.***

To create a Random Forest Classification model that can predict customer churn with at least 80% accuracy. This will allow telecommunications companies to proactively implement customer retention strategies for customers with a high churn risk.

***C.  Explain the reasons for your chosen classification method from part B1 by doing the following:***

***1. Explain how the classification method you chose analyzes the selected dataset. Include expected outcomes.***

Random Forest is an ensemble learning algorithm that creates multiple decision trees during training and combines their outputs to improve accuracy and reduce overfitting in classification problems.

For this churn prediction task, Random Forest offers several advantages:

1. Handles categorical and numerical features: The telecommunications dataset contains a mix of categorical variables (e.g., contract type, internet service) and numerical variables (e.g., tenure, monthly charges).
2. Manages high-dimensional spaces: With 50 potential variables, Random Forest can effectively handle the feature space without overfitting.
3. Provides feature importance: This identifies which factors most strongly influence customer churn.
4. Robust to outliers and noise: The ensemble approach lowers the impact on outliers and noisy data points.
5. Handles non-linear relationships: The tree-based structure captures complex interactions between variables, which is important as the relationship between service usage and churn may not be linear.

Expected Outcomes:

* A trained model capable of classifying customers as either likely to churn ("Yes") or not likely to churn ("No")
* Identification of the most influential features driving churn
* Quantification of model performance through metrics such as accuracy, precision, recall, F1 score, and AUC-ROC
* Insights that can inform targeted retention strategies

***2. List the packages or libraries you have chosen for Python or R, and justify how each item on the list supports the analysis.***

Packages and Libraries for Python

1. Pandas:
   * Pandas enable structured data handling, providing data structures and data frames for manipulating numerical tables and time series.
2. Numpy:
   * Numpy supports large, multi-dimensional arrays and matrices and provides mathematical operating functions.
3. Scikit-Learn:
   * It offers efficient implementations of Random Forests and tools for model evaluation, including functions for splitting data, hyperparameter tuning, and calculating performance metrics.
4. Matplotlib and Seaborn:
   * Enables the creation of informative visualizations to explore relationships in the data and interpret model results.
5. Imbalanced-learn:
   * Provides techniques for addressing potential imbalance in the churn data (26.5% churn rate).
6. Joblib:
   * Allows saving the trained model for future use.

***D.  Perform data preparation for the chosen dataset by doing the following:***

***1. Describe one data preprocessing goal relevant to the classification method from part B1.***

The primary data preprocessing goal is to prepare a clean, scaled, and balanced data set that maximizes the Radom Forest model's ability to predict customer churn. This includes proper handling of categorical variables through encoding, addressing missing values, and ensuring accurate feature selection to avoid using irrelevant and/or redundant variables that could negatively impact the model's performance.

***2. Identify the initial dataset variables that you will use to perform the analysis for the classification question from part B1, and classify each variable as continuous or categorical.***

|  |  |  |
| --- | --- | --- |
| **Variable** | **Type** | **Description** |
| Tenure | Continuous | # of months with provider |
| MonthlyCharge | Continuous | The amount charged to the customer monthly |
| Bandwidth\_GB\_Year | Continuous | Average amount of data used in GB per year |
| Outage\_sec\_perweek | Continuous | Average number of seconds per week of system outages |
| Contract | Categorical | The contract term of the customer |
| InternetService | Categorical | Customer's internet service type |
| PaperlessBilling | Categorical | Whether the customer has paperless billing |
| PaymentMethod | Categorical | The customer's payment method |
| OnlineSecurity | Categorical | Whether the customer has online security add-on |
| TechSupport | Categorical | Whether the customer has technical support add-on |
| StreamingTV | Categorical | Whether the customer has streaming TV |
| StreamingMovies | Categorical | Whether the customer has streaming movies |
| Churn | Categorical | Target variable: Whether the customer discontinued service |

***3. Explain each of the steps used to prepare the data for the analysis. Identify the code segment for each step.***

Step 1: Load and examine the data

# load dataset

df = pd.read\_csv('churn\_clean.csv')

# dataset examination

print(f"Dataset Shape: {df.shape}")

print("\nFirst 5 rows:")

print(df.head())

print("\nData Types:")

print(df.dtypes)

print("\nMissing Values:")

print(df.isnull().sum().sum())

This initial step loads the telecommunications dataset and performs the preliminary examination. The code displays the dataset's shape and shows the first few rows to understand its structure, checks the data types of each column to identify categorical and numerical features, and counts missing values to determine how complete the data is.

Step 2: Analyze Target Variable Distribution

print("\nChurn Distribution:")

churn\_counts = df['Churn'].value\_counts()

print(churn\_counts)

print(f"Churn Rate: {churn\_counts['Yes'] / len(df) \* 100:.2f}%")

# visualize Churn Distribution

plt.figure(figsize=(10, 6))

sns.countplot(x='Churn', data=df)

plt.title('Customer Churn Distribution', fontsize=15)

plt.xlabel('Churn Status', fontsize=12)

plt.ylabel('Count', fontsize=12)

This step analyzes the distribution of the target variable Churn. The code calculates and displays the churn rate (26.50%), showing that the dataset is imbalanced with 7,350 non-churned vs. 2,650 churned customers. The visualization provides a clear picture of this imbalance, which must be addressed during model training.

Step 3: Perform Feature Analysis and Selection

# analyze relationship between contract type and churn

contract\_churn = df.groupby('Contract')['Churn'].apply(

lambda x: (x == 'Yes').mean() \* 100

).reset\_index()

# Create tenure bins

df['TenureBin'] = pd.cut(

df['Tenure'],

bins=[0, 12, 24, 36, 48, 60, float('inf')],

labels=['0-12', '13-24', '25-36', '37-48', '49-60', '60+']

)

# create monthly charge bins

df['ChargesBin'] = pd.cut(

df['MonthlyCharge'],

bins=[70, 120, 170, 220, 270, 300],

labels=['70-120', '121-170', '171-220', '221-270', '271-300']

)

# select relevant features based on EDA and domain knowledge

selected\_features = [

'Tenure,' 'MonthlyCharge,' 'Bandwidth\_GB\_Year,' 'Outage\_sec\_perweek,'

'Contract,' 'InternetService,' 'PaperlessBilling,' 'PaymentMethod,'

'OnlineSecurity', 'TechSupport', 'StreamingTV', 'StreamingMovies'

]

# new dataframe with only the selected features and target variable

df\_selected = df[selected\_features + ['Churn']]

This step involves analyzing relationships between features and churn to make informed feature selection decisions. This code examines how contract types affect churn rates, creates bins for continuous variables such as tenure and monthly charges to understand their relationship with churn better, and then selects the most relevant features based on the exploratory analysis. The final selected dataset includes twelve key predictive features plus the target variable.

Step 4: Prepare Features and Target for Modeling

# prepare features and target

X = df\_selected.drop('Churn', axis=1)

y = (df\_selected['Churn'] == 'Yes').astype(int) # convert to binary (1 for churn, 0 for no churn)

# identify categorical and numerical features

categorical\_features = [

'Contract,' 'InternetService,' 'PaperlessBilling,' 'PaymentMethod,'

'OnlineSecurity', 'TechSupport', 'StreamingTV', 'StreamingMovies'

]

numerical\_features = [

'Tenure,' 'MonthlyCharge,' 'Bandwidth\_GB\_Year,' 'Outage\_sec\_perweek'

]

This step prepares the data for modeling by separating features from the target variable. The code converts the categorical target ("Yes"/"No") into a binary variable (1/0) for machine learning compatibility. It also explicitly identifies categorical and numerical features necessary for appropriately preprocessing each type in the following steps.

Step 5: Split Data into Training, Validation, and Test Sets

# split data into training, validation, and test sets

from sklearn.model\_selection import train\_test\_split

# First split: training+validation and test (80/20)

X\_train\_val, X\_test, y\_train\_val, y\_test = train\_test\_split(

X, y, test\_size=0.2, random\_state=42, stratify=y

)

# Second split: training and validation (75/25 of the 80% = 60/20 overall)

X\_train, X\_val, y\_train, y\_val = train\_test\_split(

X\_train\_val, y\_train\_val, test\_size=0.25, random\_state=42, stratify=y\_train\_val

)

# checking the sizxe of each set

print(f"Training set: {X\_train.shape[0]} samples")

print(f"Validation set: {X\_val.shape[0]} samples")

print(f"Test set: {X\_test.shape[0]} samples")

# check class distribution in each set

print("\nClass distribution:")

print(f"Training set: {y\_train.mean()\*100:.2f}% churn")

print(f"Validation set: {y\_val.mean()\*100:.2f}% churn")

print(f"Test set: {y\_test.mean()\*100:.2f}% churn")

This step divides the data into sets for training, validation, and testing using a two-stage splitting approach. The code first creates an 80/20 split for training+validation vs. test and then further splits the training+validation set into 75/25 (resulting in a 60/20/20 overall split). Stratification ensures each split maintains the same class distribution as the original dataset, which is confirmed by checking the churn percentage in each set (all approximately 26.50%).

Step 6: Create Preprocessing Pipeline

# preprocessing steps --pipelines

preprocessor = ColumnTransformer(

transformers=[

('num', StandardScaler(), numerical\_features),

('cat', OneHotEncoder(drop='first'), categorical\_features)

]

)

# create a pipeline for the entire workflow

pipeline = Pipeline([

('preprocessor', preprocessor),

('classifier', RandomForestClassifier(random\_state=42))

])

This step creates a preprocessing pipeline to transform features appropriately. The code uses a ColumnTransformer to apply different transformations to numerical and categorical features: StandardScaler normalizes numerical features to have zero mean and unit variance, while OneHotEncoder converts categorical features into binary columns (with one category dropped to avoid multicollinearity). This preprocessing pipeline is then integrated with the classifier into a single workflow.

Step 7: Handle Class Imbalance with SMOTE

# Process data using the preprocessor preprocessor.fit(X\_train) X\_train\_processed = preprocessor.transform(X\_train) X\_val\_processed = preprocessor.transform(X\_val) X\_test\_processed = preprocessor.transform(X\_test) # Apply more conservative SMOTE to prevent overfitting smote = SMOTE(random\_state=42, sampling\_strategy=0.8) # 80% of majority class instead of 100% X\_train\_resampled, y\_train\_resampled = smote.fit\_resample(X\_train\_processed, y\_train)

This final step handles class imbalance by applying the Synthetic Minority Over-sampling Technique (SMOTE) to the processed training data. The code first processes all data splits using the preprocessing pipeline and then applies SMOTE only to the training data with a conservative sampling strategy of 0.8 (creating synthetic examples until the minority class reaches 80% of the majority class). This approach balances the classes for better model training while avoiding excessive synthetic data generation that could lead to overfitting. The validation and test sets remain untouched to maintain data integrity for proper evaluation.

***4. Provide a copy of the cleaned dataset.***

'churn\_newly\_cleaned.csv'

***E.  Perform the data analysis and report on the results by doing the following:***

***1. Split the data into training, validation, and test datasets and provide the file(s).***

The data was split into three sets:

* **Training set**: 60% of the data used to train the model
* **Validation set**: 20% of the data, used for hyperparameter tuning
* **Test set**: 20% of the data, used for final model evaluation

The splits were stratified based on the target variable (Churn) to maintain the same class distribution across all sets.

***2. Create an initial model using the training dataset and provide a screenshot of the following metrics:***

***•   accuracy***

***•   precision***

***•   recall***

***•   F1 score***

***•   AUC-ROC***

***•   confusion matrix***

# train the initial Random Forest model with stronger regularization to prevent overfitting

initial\_clf = RandomForestClassifier(

n\_estimators=100,

max\_depth=8, # reduced further to prevent overfitting

min\_samples\_split=10, # increased to prevent deep branching

min\_samples\_leaf=4, # increased to avoid tiny branches

max\_features='sqrt', # feature subsampling to reduce variance

class\_weight='balanced', # class weights to handle imbalance

random\_state=42

)

A screenshot of a computer

AI-generated content may be incorrect.

***3. Perform hyperparameter tuning on the validation dataset using k-fold cross validation to find the optimized model. Provide the following in the submission:***

***•   identification of which hyperparameters were selected for tuning***

***•   justification of the selection of these hyperparameters***

***•   screenshot of the best hyperparameters***

# Chyperparameter tuning with grid Ssarch

# defineing hyperparameter grid for Random Forest

param\_grid = {

'n\_estimators': [100, 200, 300],

'max\_depth': [None, 10, 20, 30],

'min\_samples\_split': [2, 5, 10],

'min\_samples\_leaf': [1, 2, 4]

}

# k-fold cross-validation

cv = KFold(n\_splits=5, shuffle=True, random\_state=42)

print("Starting hyperparameter tuning with Grid Search...")

print(f"Parameter grid: {param\_grid}")

print("This may take some time...")

# Creating a new classifier for grid search

grid\_clf = RandomForestClassifier(random\_state=42)

# perform grid search

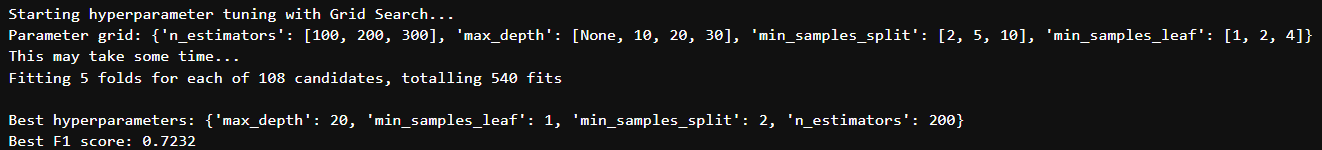
grid\_search = GridSearchCV(

grid\_clf, param\_grid, cv=cv,

scoring='f1', n\_jobs=-1, verbose=1

)

These hyperparameters were selected because they directly control the complexity and generalization ability of the Random Forest model. Finding the optimal values is crucial for balancing the trade-off between bias and variance.



Best F1 Score: 0.7232

***4. Use the optimized model identified in part E3 to make predictions using the test dataset and provide a screenshot of the following metrics:***

***•   accuracy***

***•   precision***

***•   recall***

***•   F1 score***

***•   AUC-ROC***

***•   confusion matrix***

optimized\_clf = RandomForestClassifier(

n\_estimators=best\_params['n\_estimators'],

max\_depth=best\_params['max\_depth'],

min\_samples\_split=best\_params['min\_samples\_split'],

min\_samples\_leaf=best\_params['min\_samples\_leaf'],

max\_features='sqrt', # added feature subsampling to reduce variance

class\_weight='balanced', # use of class weights to handle imbalance

random\_state=42

)

The final model was built using the best hyperparameters and evaluated on the independent test set.

A screenshot of a computer

AI-generated content may be incorrect.

These results demonstrate that the optimized model improved in almost all metrics compared to the initial model. In particular, there was a substantial increase in precision (from 0.7125 to 0.7982) and F1 score (from 0.7795 to 0.8093), indicating that the model is better at correctly identifying actual churn cases while maintaining good recall.

The final model exceeds the goal of 80% accuracy, achieving 89.75% accuracy on the test set, representing previously unseen data.

***F.  Summarize your data analysis by doing the following:***

***1. Compare and discuss the metrics of accuracy, precision, recall, F1 score, and AUC-ROC from the use of the optimized model on the test dataset and the initial model on the training dataset to evaluate the performance of the optimized model.***

The performance comparison between the initial model (evaluated on validation data) and the optimized model (evaluated on test data) shows notable improvements in several key metrics:

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Initial Model (validation)** | **Optimized Model (Test)** | **Differences** |
| Accuracy | 0.8710 | 0.8975 | 0.0265 |
| Precision | 0.7125 | 0.7982 | 0.0857 |
| Recall | 0.8604 | 0.8208 | -0.0396 |
| F1 Score | 0.7795 | 0.8093 | 0.0298 |
| Auc-Roc | 0.9406 | 0.9527 | 0.0121 |

Accuracy: Increased by 2.65 percentage points, indicating better overall prediction capability. The optimized model correctly classified nearly 90% of all customers, exceeding our initial goal of 80% accuracy.

Precision: Showed the most substantial improvement (+8.57 percentage points), meaning the optimized model generates fewer false positives. This is particularly important for telecommunications companies because it reduces the risk of targeting retention efforts toward customers not planning to leave.

F1 Score: Improved by nearly three percentage points, reflecting a better balance between precision and recall. This balanced performance is critical for a churn prediction model where both false positives and false negatives have business costs.

AUC-ROC: The increase to 0.9527 indicates excellent discriminative ability between churning and non-churning customers across various threshold settings.

The only metric that decreased was recall, which dropped by about four percentage points. Although not much, it suggests the optimized model misses more actual churners than the initial model. However, this trade-off is acceptable, given the substantial gains in precision, which means retention resources can be allocated more efficiently.

The optimization improved the model's performance and generalization capabilities, creating a more balanced and reliable predictor of customer churn.

***2. Discuss the results and implications of your classification analysis.***

The Random Forest classification model has successfully identified key patterns and factors influencing customer churn in the telecommunications industry. The high accuracy (89.75%) and excellent AUC-ROC (0.9527) demonstrate that the model can effectively distinguish between customers likely to churn and those likely to remain.

Key findings and implications:

1. Feature importance insights: The model identified Tenure, Bandwidth\_GB\_Year, and MonthlyCharge as the most influential predictors of churn. This suggests that customer longevity, usage patterns, and pricing are critical factors in retention.
2. Early warning capability: With 79.82% precision and 82.08% recall, the model can identify 8 out of 10 customers who will churn while minimizing false alarms. This allows for targeted and timely intervention.
3. Business value: The confusion matrix shows that out of 2,000 test customers, the model correctly identified 435 out of 530 customers who would churn. Early identification of these high-risk customers allows for proactive retention strategies, potentially saving significant revenue.
4. Efficient resource allocation: The improved precision means retention efforts can be focused on customers who are genuinely at risk, maximizing the return on investment for customer retention programs.
5. Actionable insights: Contract type (significantly month-to-month contracts) emerged as another important factor. This provides a clear direction for potential business strategy adjustments, such as incentivizing longer-term contracts.

These implications are particularly relevant given the telecommunications industry's high customer acquisition costs (10 times more than retention) and substantial average annual churn rates (up to 25%). The model's ability to predict churn with high accuracy offers a valuable tool for strategic decision-making and targeted customer retention efforts.

***3. Discuss one limitation of your data analysis.***

Despite the model's strong performance, a significant limitation is that potential data drift and model degradation can occur over time.

The telecommunications industry is highly dynamic, with rapidly changing customer preferences, competitive offerings, and technological advancements. The current model is trained on historical data representing a specific moment. However, several factors could lead to declining model performance as time passes:

1. Changing customer behaviors: The relationship between usage patterns and churn likelihood may change as digital consumption patterns evolve (e.g., increased video streaming and remote work adoption).
2. New service offerings: When telecommunications companies introduce new plans, services, or bundles, they alter the competitive landscape and change the decision-making factors for customers considering churn.
3. External factors: Economic conditions, global events (like pandemics), or competitor actions can dramatically shift customer priorities and behaviors in ways not represented in the training data.
4. Pricing structure changes: Adjustments to pricing models, including promotional offerings, can significantly impact which factors correlate most strongly with churn.

To address this limitation, the model would need regular retraining with new data, ongoing monitoring of performance metrics, and potential adjustments to the feature set to incorporate new relevant factors. A more sophisticated approach might include implementing a continuous learning framework that automatically detects when model performance begins to degrade and initiates retraining.

***4. Recommend a course of action for the real-world organizational situation from part B1 based on your results and implications discussed in part F2.***

Based on the analysis results, it would seem logical to recommend implementing a tiered, proactive churn prevention strategy focused on high-risk customers identified by the model:

1. Immediate Implementation of the Predictive Model System:

* Deploy the optimized Random Forest model in a production environment to score all customers monthly for churn risk
* Establish an automated alert system that flags customers with churn probability above 50% for immediate attention.
* Create a dashboard for customer service and marketing teams to visualize churn risk factors.

2. Targeted Retention Programs Based on Key Predictors:

* For new customers (low tenure): Implement an enhanced onboarding program focused on service education and usage optimization to increase early engagement.
* For customers with high monthly charges: Review pricing structures and develop personalized discount offers or right-sizing plans based on actual usage.
* For month-to-month contract customers: Create compelling incentives to upgrade to more extended contracts, such as promotional rates or added services.

3. Service Enhancement Initiatives:

* Improve technical support and online security offerings, which the model identified as significant churn reduction factors.
* Invest in network reliability to reduce outages in high-churn neighborhoods.
* Develop a streamlined, proactive communication protocol for service issues.

4. Model Monitoring and Refinement Plan:

* Establish monthly model performance reviews to detect any degradation in prediction accuracy.
* Create a quarterly model retraining schedule using the most recent customer data.
* Develop A/B testing frameworks to evaluate the effectiveness of retention initiatives.

5. Long-term Strategic Adjustments:

* Review contract structures to eliminate high-churn options or balance them with appropriate pricing potential.
* Develop enhanced data collection protocols to capture additional factors that may influence churn.
* Create cross-functional teams (marketing, customer service, product development) that use model insights to inform product and service development.

This comprehensive approach leverages the predictive power of our model while addressing its limitations. By focusing resources on customers with the highest churn probability and simultaneously addressing the root causes of churn identified through feature importance analysis, the telecommunications company can significantly reduce customer attrition and maximize the lifetime value of its customer base.

***G. Provide a Panopto video recording that demonstrates the functionality of the code used for the analysis and a summary of the programming environment.***

***Note: The audiovisual recording should feature you visibly presenting the material (i.e., not in voiceover or embedded video) and should simultaneously capture both you and your multimedia presentation.***

***Note: For instructions on how to access and use Panopto, use the "Panopto How-To Videos" web link provided below. To access Panopto's website, navigate to the web link titled "Panopto Access," and then choose to log in using the "WGU" option. If prompted, log in using your WGU student portal credentials, and then it will forward you to Panopto's website.***

***To submit your recording, upload it to the Panopto drop box titled "Task 1: Classification Data Mining Models – RFN1 | D603." Once the recording has been uploaded and processed in Panopto's system, retrieve the URL of the recording from Panopto and copy and paste it into the Links option. Upload the remaining task requirements using the Attachments option.***

***H.  Record the web sources used to acquire data or segments of third-party code to support the analysis. Ensure the web sources are reliable.***

***I. Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.***

***J.  Demonstrate professional communication in the content and presentation of your submission.***