Data Mining D209
Task 2
Eric D. Otten
Student ID: 011399183

A1. Propose one question relevant to a real-world organizational situation that you will answer using one of the following prediction methods: decision trees, random forests, advanced regression (i.e., lasso or ridge regression)

I will use the same research question as Task 1 of D209, "Which customer factors contribute most to a customer's decision to churn?" This question is highly relevant to any organization, particularly in competitive industries such as telecommunications. Understanding the key drivers of customer churn can help a company implement targeted interventions to improve customer retention, which is often more cost-effective than acquiring new customers. I will be using the random forests prediction model due to its predictive accuracy and ability to handle a complex dataset with many variables. I decided against using lasso regression due to its bias for high-dimensional data. I also decided against using decision trees as they can be prone to overfitting, especially with complex trees.

I will be using the telecom churn dataset provided with this course.

A2. Define one goal of the data analysis. Ensure that your goal is reasonable within the scope of the scenario and is represented in the available data.

My goal with this data analysis is to improve upon the KNN model constructed in D209 Task 1. Random forests are known for their high accuracy in classification problems and work well on large datasets such as the telecom company's churn dataset.

One of the most valuable outputs of random forests is the ranking of features by importance. This is achieved by looking at how much each feature decreases the impurity of the split (e.g., Gini impurity for classification tasks). The more a feature decreases the impurity, the more important it is. This output should provide a clear answer to my research question, "Which customer factors contribute most to a customer's decision to churn?"

By identifying which features are most important, I can understand what drives customers to churn. This might include factors like service quality, pricing, usage patterns, customer support interactions, contract types, etc. Then this information will be used to propose a recommendation for a real-world organization to use in Part E4.

B1. Explain how the prediction method you chose analyzes the selected data set. Include expected outcomes.

Random forest has a wide range of use cases involving banking, stock trading, medicine, and e-commerce. As previously mentioned in Part B1, one of the most valuable outputs of random forests is the ranking of features by importance.

This will be used with the churn data to identify which customer factors most influence churn, which allows an organization to do three things: 1) predict which current customers are likely to churn, 2) identify which demographic or psychographic market segments are likely to churn, and 3) identify which service offerings involve a higher rate of churn.

The sklearn Python library implements random forest through the method RandomForestRegressor. RandomForestRegressor builds multiple decision trees and merges them together to get more accurate and stable predictions. The fundamental technique behind RandomForestRegressor is bagging, which stands for Bootstrap Aggregating and involves sampling and training.

When constructing each tree in the forest, only a randomly selected subset of features is considered at each split. This decreases the correlation between individual trees in the forest, especially if there are a few strong predictors in

the data set. In traditional decision tree learning, the most dominant features (e.g., purchase frequency) might always be chosen early in the tree splits, making many trees similar or correlated. In RandomForestRegressor, when building each tree, only a random subset of features might be considered at each split. For example, even if purchase frequency is a strong predictor, it might not be considered in some trees at all. By leveraging multiple trees, the model averages out many of the errors. If some trees give predictions that are way off, they are likely to be balanced out by more accurate predictions from other trees.

The random forest method also makes the forest more robust to noise in the data. For example, one tree might use purchase frequency and average spend to predict churn, while another might use time on site and customer service interactions. This variety ensures that the random forest does not overly rely on possibly noisy or outlier-affected features. The use of bagging and random feature selection means the model is less sensitive to the specifics of any single training dataset and to overfitting, making it robust across different datasets and scenarios.

The output of random forest regression is an easily interpreted model. Although the ensemble itself is complex, feature importance scores derived from the forest provide clear insights into which features are driving predictions, aiding interpretability for decision-makers. The output provides insights into the importance of each feature in predicting the target variable. This is calculated based on how much the MSE decreases due to splits over a given feature, averaged over all trees.

B2. Summarize one assumption of the chosen prediction method.

Random forest is a non-parametric model and as such doesn't assume an explicit form or functional relationship between the dependent and independent variables, which results in fewer assumptions about data distribution.

Random forest relies on the idea that with a sufficiently large number of trees, the ensemble can converge to a stable prediction. Thus, it operates under the assumption that **more trees in the forest will generally lead to better and more robust predictions** by averaging out biases and variances across individual trees. As Sirikulviriya et al. notes, "Ensemble method is a popular machine learning technique which has been interested in data mining communities" (Sirikulviriya et al., 2011)

Aside from this assumption, one goal of the random forest algorithm is that the predictions from each tree should have very low correlations. While random forest aims to minimize the correlation between trees, the trees are not strictly independent because they are derived from the same original dataset through bootstrap sampling. The key here is not complete independence, but low correlation.

At each split in the construction of the tree, random forest randomly selects a subset of features to consider. This method, known as feature bagging, is crucial because it ensures that trees do not always split on the same features, even if some features are very strong predictors. This reduces the likelihood that trees will be similar, hence reducing their correlation.

The primary motivation behind ensuring low correlation among trees in a random forest is to reduce the variance of the ensemble prediction. When trees are less correlated, the ensemble can average out their individual errors, leading to a more accurate and stable estimate than any single tree could provide.

B3. List the packages or libraries you have chosen for Python or R and justify how each item on the list supports the analysis.

For this task, I chose to use Python due to its flexibility and extensive libraries, of which I will be using the following:

- The Pandas library. Pandas is a Python library used for data manipulation and analysis that provides data structures and operations for manipulating numerical tables and time series. The most prominent feature of this package is the `pandas.DataFrame` constructor which allows for the storage and manipulation of data with rows and columns, similar to a spreadsheet or SQL table.

- The NumPy library. Numpy is a Python library that adds support for multi-dimensional arrays and matrices and high-level mathematical functions like Fourier transforms and matrix multiplications to operate on these arrays.

- The MatPlotLib library. MatPlotLib is a visualization library in Python. It is a base library that provides a lot of flexibility but can be complex when attempting to create advanced visualizations.

- The Seaborn library. Seaborn is another visualization library in Python built on top of MatPlotLib that provides a high-level interface for creating graphs such as bar charts, line charts, histograms, scatter plots, etc and simplifies many aspects of visualization.

- SciPy's statsmodels API. Statsmodels is a Python module that provides classes and functions to estimate many different statistical models and conduct statistical tests and statistical data exploration. It includes various tools for modeling statistics, including linear regression, time series analysis, etc.

- The Scikit-Learn/sklearn library. The sklearn library offers a broad array of machine learning algorithms including classification, regression, clustering, and dimensionality reduction. Additionally, it provides utilities for model fitting, data preprocessing, model selection, and evaluation, making it extremely versatile. One of the core strengths of sklearn is its consistent and intuitive API and integration with other libraries in the Python data science stack, such as NumPy and SciPy. While primarily designed for small to medium data sets, sklearn can handle much larger data sets by integrating it with other tools in the Python ecosystem. Specifically, the following functions will be used from the sklearn library:

C1. Describe one data preprocessing goal relevant to the prediction method from part A1.

My goal with regards to cleaning the sample data is to create a uniform DataFrame to which the random forest method can be applied and from which useful business insights can be drawn. This is similar to my data cleaning goal in D209 Task 1, with the exception of the classification method used.

The provided dataset is cleaned yet contains several data anomalies which should be corrected:

- Zip codes are provided in int28 format instead of as a string and as a result have lost their leading zeroes. These rows will be converted to strings.
- Time zone categories are redundant. For example, separate time zones exist for EST: New York, Detroit, and others. These categories will be reduced to the standard US time zones. The following mappings will be used:
    - America/New_York will be mapped to EST.
    - America/Detroit will be mapped to EST.
    - America/Indiana/Indianapolis will be mapped to EST.
    - America/Kentucky/Louisville will be mapped to EST.
    - America/Indiana/Vincennes will be mapped to EST.
    - America/Indiana/Tell_City will be mapped to EST.
    - America/Indiana/Petersburg will be mapped to EST.
    - America/Indiana/Knox will be mapped to EST.
    - America/Indiana/Winamac will be mapped to EST.
    - America/Indiana/Marengo will be mapped to EST.
    - America/Toronto will be mapped to EST.
    - America/Chicago will be mapped to CST.
    - America/Menominee will be mapped to CST.
    - America/North_Dakota/New_Salem will be mapped to CST.
    - America/Denver will be mapped to MST.
    - America/Phoenix will be mapped to MST.
    - America/Boise will be mapped to MST.
    - America/Los_Angeles will be mapped to PST.
    - America/Anchorage will be mapped to AKST.
    - America/Nome will be mapped to AKST.

    - America/Sitka will be mapped to AKST.
    - America/Juneau will be mapped to AKST.
    - Pacific/Honolulu will be mapped to HAST.
    - America/Puerto_Rico will be mapped to AST.
    - America/Ojinaga will be mapped to MST.

For nominal categorical data, one hot encoding will be used as it is the most widespread approach. This approach involves creating a new column for each category, which contains a binary encoding of 0 or 1 to denote whether a particular row belongs to this category. This can be achieved using the get_dummies() method within the pandas library. Additionally, the `drop_first=True` argument will be used to avoid the dummy variable trap.

C2. Identify the initial data set variables that you will use to perform the analysis for the prediction question from part A1 and group each variable as numeric or categorical.

The following variables will be used to perform my data analysis for my classification question, “Which customer factors contribute most to a customer's decision to churn?”:

- **Population**: Population within a mile radius of the customer, based on census data. (Numeric)
- **Area**: Classification of the customer's area (rural, urban, suburban). (Categorical)
- **TimeZone**: Time zone of the customer's residence. (Categorical)
- **Children**: Number of children in the customer’s household as reported during sign-up. (Numeric)
- **Age**: Age of the customer as reported during sign-up. (Numeric)
- **Income**: Annual income of the customer as reported at the time of sign-up. (Numeric)
- **Marital**: Marital status of the customer. (Categorical)
- **Gender**: Gender of the customer as they self-identify. (Categorical)
- **Churn:** Whether the customer discontinued service within the last month (yes, no) (Categorical)
- **Outage_sec_perweek**: Average number of seconds per week of system outages experienced by the customer. (Numeric)
- **Email**: Number of emails sent to the customer in the last year. (Numeric)
- **Contacts**: Number of times the customer contacted technical support. (Numeric)
- **Yearly_equip_failure**: Number of times the customer’s equipment failed and needed replacement or reset in the past year. (Numeric)
- **Techie**: Indicates whether the customer considers themselves technically inclined. (Categorical)
- **Contract**: The type of service contract the customer has (month-to-month, one year, two years). (Categorical)
- **Port_modem**: Whether the customer uses a portable modem. (Categorical)
- **Tablet**: Whether the customer owns a tablet device. (Categorical)
- **InternetService**: Type of internet service the customer has (DSL, fiber optic, none). (Categorical)
- **Phone**: Whether the customer has a phone service. (Categorical)
- **Multiple**: Whether the customer has multiple lines. (Categorical)
- **OnlineSecurity**: Whether the customer has an online security service. (Categorical)
- **OnlineBackup**: Whether the customer uses an online backup service. (Categorical)
- **DeviceProtection**: Whether the customer uses a device protection service. (Categorical)
- **TechSupport**: Whether the customer has technical support service. (Categorical)
- **StreamingTV**: Whether the customer uses streaming TV service. (Categorical)
- **StreamingMovies**: Whether the customer uses streaming movies service. (Categorical)

- **PaperlessBilling**: Whether the customer has opted for paperless billing. (Categorical)
- **PaymentMethod**: Method by which the customer makes payments (e.g., electronic check, mailed check, bank transfer, credit card). (Categorical)
- **Tenure**: Number of months the customer has been with the service provider. (Numeric)
- **MonthlyCharge**: Average monthly charge billed to the customer. (Numeric)
- **Bandwidth_GB_Year**: Average amount of data in GB used by the customer per year. (Numeric)
- **Item1**: Timely response - This survey item evaluates how quickly the company responds to customer inquiries and requests, which can significantly influence customer satisfaction and perception of the company's service efficiency. (Numeric)
- **Item2**: Timely fixes - This item measures the promptness of the company in addressing and resolving technical or service-related issues reported by customers. (Numeric)
- **Item3**: Timely replacements - This question assesses how swiftly the company manages to replace faulty or inadequate equipment or services that do not meet the customer's expectations or needs. (Numeric)
- **Item4**: Reliability - This survey question gauges the consistency and dependability of the company's services, reflecting how often customers face issues or disruptions. (Numeric)
- **Item5**: Options - This item looks at the variety and flexibility of service options available to customers, allowing them to choose services that best fit their needs and preferences. (Numeric)
- **Item6**: Respectful response - This question assesses the respectfulness and professionalism of the company's responses to customer interactions, which can affect the customer's overall service experience. (Numeric)
- **Item7**: Courteous exchange - Similar to respectful responses, this survey item evaluates the courtesy extended by the company during customer interactions, including politeness and positive communication. (Numeric)
- **Item8**: Evidence of active listening - This item measures the extent to which customers feel that the company listens to and understands their concerns or requests, which is crucial for effective service and support. (Numeric)

C3. Explain the steps used to prepare the data for the analysis. Identify the code segment for each step.

My goal with regards to cleaning the sample data is to create a uniform DataFrame to which the random forest method can be applied and from which useful business insights can be drawn. To achieve this, the following data transformations will be performed:

1. Select only the columns that are relevant to the research question.
2. Standardize timezones using a mapping function.
3. Convert columns intended to represent boolean values to booleans with “Yes” being mapped to True and “No” being mapped to False.
4. Convert nominal variables that include repeating values to categories.
5. Use one-hot encoding to create dummy columns, with the first category of each categorical variable being dropped to avoid the dummy variable trap.

The annotated code below achieves these objectives:

1. **Select only the columns that are relevant to the research question.**

Code

```
# Remove unused columns
df = df[[
    "Population", "Area", "TimeZone", "Children", "Age", "Income", "Marital", "Gender",
"Churn",
    "Outage_sec_perweek", "Email", "Contacts", "Yearly_equip_failure", "Techie",
    "Contract", "Port_modem", "Tablet", "InternetService", "Phone", "Multiple",
    "OnlineSecurity", "OnlineBackup", "DeviceProtection", "TechSupport", "StreamingTV",
    "StreamingMovies", "PaperlessBilling", "PaymentMethod", "Tenure", "MonthlyCharge",
    "Bandwidth_GB_Year", "Item1", "Item2", "Item3", "Item4", "Item5", "Item6", "Item7",
"Item8"
]]
```

2. **Standardize timezones using a mapping function.**

Code

```
# Mapping of locations to time zones
time_zone_map = {
    "America/New_York": "EST",
    "America/Detroit": "EST",
    "America/Indiana/Indianapolis": "EST",
    "America/Kentucky/Louisville": "EST",
    "America/Indiana/Vincennes": "EST",
    "America/Indiana/Tell_City": "EST",
    "America/Indiana/Petersburg": "EST",
    "America/Indiana/Knox": "EST",
    "America/Indiana/Winamac": "EST",
    "America/Indiana/Marengo": "EST",
    "America/Toronto": "EST",
    "America/Chicago": "CST",
    "America/Menominee": "CST",
    "America/North_Dakota/New_Salem": "CST",
    "America/Denver": "MST",
    "America/Phoenix": "MST",
    "America/Boise": "MST",
    "America/Los_Angeles": "PST",
    "America/Anchorage": "AKST",
    "America/Nome": "AKST",
    "America/Sitka": "AKST",
    "America/Juneau": "AKST",
    "Pacific/Honolulu": "HAST",
    "America/Puerto_Rico": "AST",
    "America/Ojinaga": "MST",
}

# Replace the TimeZone column with the mapped values
df["TimeZone"] = df["TimeZone"].map(time_zone_map)
```

3. **Convert columns intended to represent boolean values to booleans with “Yes” being mapped to True and “No” being mapped to False.**

Code

```
# Convert boolean columns to actual boolean types
boolean_columns = [
    "Churn",
    "Techie",
    "Port_modem",
    "Tablet",
    "Phone",
    "Multiple",
    "OnlineSecurity",
    "OnlineBackup",
    "DeviceProtection",
    "TechSupport",
    "StreamingTV",
    "StreamingMovies",
    "PaperlessBilling",
]
for column in boolean_columns:
    df[column] = df[column].map({"Yes": True, "No": False})
```

4. **Convert nominal variables that include repeating values to categories.**

Code

```
# Convert remaining categorical data to category dtype
nominal_columns = [
    "Area",
    "TimeZone",
    "Marital",
    "Gender",
    "Contract",
    "InternetService",
    "PaymentMethod",
]
for column in nominal_columns:
    df[column] = df[column].astype("category")
```

5. **Use one-hot encoding to create dummy columns, with the first category of each categorical variable being dropped to avoid the dummy variable trap.**

Code

```
# Get dummy columns
df = pd.get_dummies(df, columns = nominal_columns, drop_first = True)
```

C4. Provide a copy of the cleaned data set.

See churn_encoded.csv

D1. Split the data into training and test data sets and provide the file(s).

The annotated code below splits the data into training and test datasets and outputs them to .csv files.

Code

```
# Assuming 'df' is your DataFrame and 'Churn' is the target variable
X = df.drop('Churn', axis=1)
y = df['Churn']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Save the training and testing datasets to CSV files for reference
X_train.to_csv("churn_Xtrain.csv", index = False)
X_test.to_csv("churn_Xtest.csv", index = False)
y_train.to_csv("churn_ytrain.csv", index = False)
y_test.to_csv("churn_ytest.csv", index = False)

# Assuming preprocessing has been done to handle categorical variables etc.
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

See attached churn_Xtrain.csv, churn_Xtest.csv, churn_ytrain.csv, and churn_ytest.csv.

D2. Describe the analysis technique you used to appropriately analyze the data. Include screenshots of the intermediate calculations you performed.

After fitting the cleaned data set using the RandomForestClassifier method, I implemented several analysis techniques on the resulting model. The annotated code below implements the following analysis techniques:

- Accuracy calculation
- Classification report
- MSE calculation
- Feature importance visualization
- Model accuracy by number of trees
- Confusion matrix visualization

Code

```
# D2. Describe the analysis technique you used to appropriately analyze the data. Include
screenshots of the intermediate calculations you performed.

# Calculate accuracy and classification report
accuracy = (y_pred == y_test).mean()
```

```
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))

# Calculate Mean Squared Error (MSE)
mse = mean_squared_error(y_test, y_pred.astype(int))
print("Mean Squared Error:", mse)



importances = model.feature_importances_
indices = np.argsort(importances)[::-1]
plt.figure()
plt.title("Feature importances")
plt.bar(range(X_train.shape[1]), importances[indices])
plt.xticks(range(X_train.shape[1]), X_train.columns[indices], rotation=90)
plt.show()

accuracies = []
estimators_range = range(1, 101, 10)
for n_estimators in estimators_range:
    model = RandomForestClassifier(n_estimators=n_estimators, random_state=42)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracies.append(accuracy_score(y_test, y_pred))

plt.plot(estimators_range, accuracies)
plt.xlabel('Number of Trees')
plt.ylabel('Accuracy')
plt.title('Model Accuracy by Number of Trees')
plt.show()

cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
plt.show()
```

Result

```
Accuracy: 0.898
              precision    recall  f1-score   support

       False       0.91      0.96      0.93      1456
        True       0.86      0.74      0.80       544

    accuracy                           0.90      2000
   macro avg       0.89      0.85      0.87      2000
weighted avg       0.90      0.90      0.90      2000

Mean Squared Error: 0.102
```

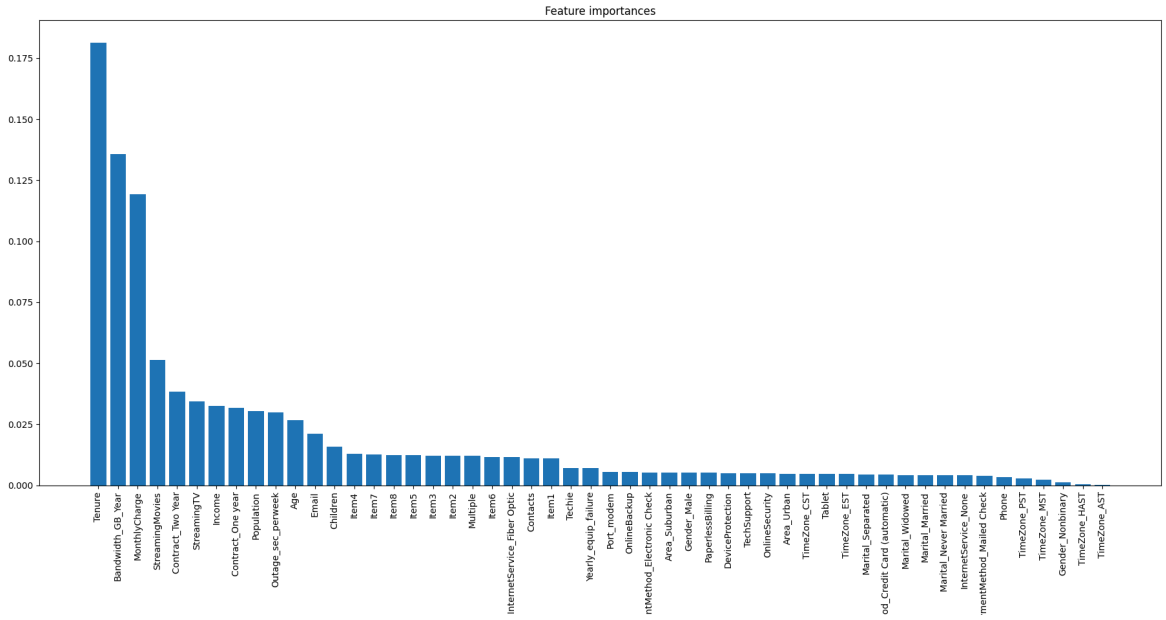
The classification report tells us that:

- Among Non-Churners (False class,)

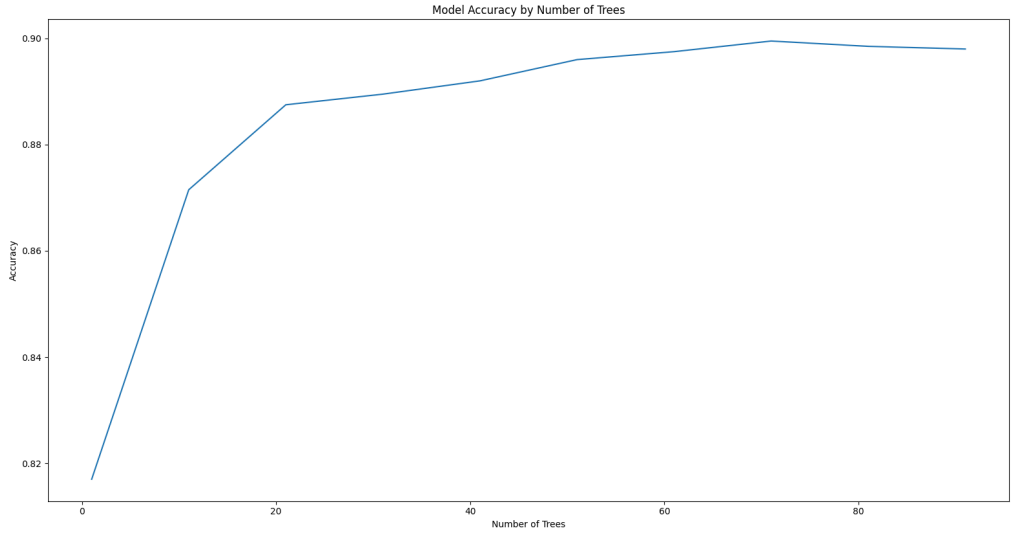
- Precision (0.91): When the model predicts that a customer will not churn, it is correct 91% of the time.
- Recall (0.96): Of the actual non-churners, the model correctly identifies 96% of them.
- F1-Score (0.93): A high F1-score for non-churners indicates a strong balance between precision and recall for this class.

- Among Churners (True class,)
  - Precision (0.86): When the model predicts that a customer will churn, it is correct 86% of the time.
  - Recall (0.74): The model correctly identifies 74% of the actual churners. This suggests some room for improvement, as about 26% of actual churners are being missed.
  - F1-Score (0.80): The lower F1-score for churners compared to non-churners suggests that it is more challenging for the model to predict churn accurately compared to predicting non-churn.

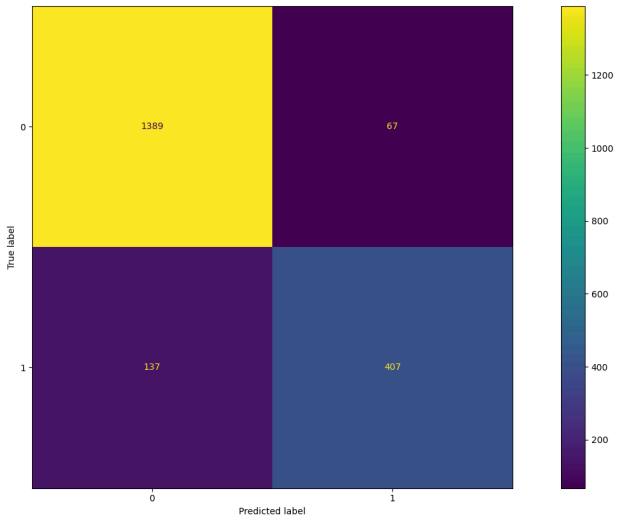
Additionally, this code outputs several graphs. First is a bar graph identifying feature importance which show that the following features have the greatest importance: Tenure, Bandwidth_GB_Year, MonthlyCharge, StreamingMovies, Contract_Two Year. See Figure 1 below.



Next, a line graph titled “Model Accuracy by Number of Trees,” represents how the accuracy of your Random Forest model changes as you vary the number of decision trees (estimators) in the ensemble. See Figure 2 below.



Lastly, a confusion matrix display is produced and included in Figure 3 below.



D3. Provide the code used to perform the prediction analysis from part D2.

See main.py

E1. Explain the accuracy and the mean squared error (MSE) of your prediction model.

The accuracy of 0.898 indicates that my model correctly predicts whether a customer will churn or not about 89.8% of the time and is therefore effective in identifying the underlying patterns of churn in your dataset.

The mean squared error (MSE) of my model is 0.102, which indicates that, on average, the square of the error between predicted and actual values is 0.102 and suggests a low error rate.

E2. Discuss the results and implications of your prediction analysis.

In Part D2, I describe the analysis techniques performed. Listed below are the analysis techniques along with their results and implications:

- Accuracy calculation – Described in Part E1
- Classification report – Described in Part E1
- MSE calculation – Described in Part E1
- Feature importance visualization – The dominant features are Tenure, Bandwidth_GB_Year, and MonthlyCharge. This tells us that Tenure is the most significant predictor of churn, implying that the length of time customers stay with the service strongly influences their likelihood to continue or discontinue the service. Additionally, Bandwidth_GB_Year and MonthlyCharge are highly influential, suggesting that how much data a customer uses and how much they are charged each month are critical in their decision to churn. An organization should therefore focus on customers with shorter tenure by implementing loyalty programs or offering incentives to increase their stickiness.
- Model accuracy by number of trees – After approximately 20 trees, the rate of increase in accuracy slows down significantly. The curve begins to flatten, indicating that each additional tree contributes less to improving the model's accuracy. From a resource and efficiency perspective, adding more trees beyond 40 does not seem to provide substantial benefits in terms of accuracy. Therefore, setting the number of trees around 40-60 could be optimal for balancing model performance and computational cost.
- Confusion matrix visualization – The model is particularly strong in identifying non-churn cases (high specificity) and has respectable accuracy overall. There is room for improvement in minimizing false negatives, as these are particularly costly in a churn prediction context. Reducing false negatives could help in retaining customers who are at risk of churning. The matrix is divided into four quadrants:
    - True Negative (TN): The top-left quadrant (Yellow - 1389) represents the number of true negative predictions. These are instances where the model correctly predicted the negative class (non-churn).
    - False Positive (FP): The top-right quadrant (Purple - 67) represents the false positives. These are instances where the model incorrectly predicted the positive class (churn) when the actual class was negative (non-churn).
    - False Negative (FN): The bottom-left quadrant (Purple - 137) shows the false negatives. These are instances where the model incorrectly predicted the negative class (non-churn) when the actual class was positive (churn).
    - True Positive (TP): The bottom-right quadrant (Blue - 407) represents the true positives. These are instances where the model correctly predicted the positive class (churn).

E3. Discuss one limitation of your data analysis.

One limitation of my data analysis involves the class imbalance issue. In many customer datasets, the number of customers who do not churn (non-churn) is usually much higher than those who do (churn). When this happens, the model might learn to predict the majority class (non-churn) very well because there is more data available for this class, thus optimizing the accuracy metric, which might be misleading.

Because there are fewer examples of the churn class, the model may not learn enough about the characteristics of churning customers. As a result, it might fail to identify churn instances accurately, leading to a high number of false negatives (i.e., customers who are predicted not to churn but do).

Relying on such a model might lead businesses to make decisions based on incomplete or skewed insights, potentially missing out on opportunities to retain customers at risk of churning.

Techniques such as oversampling the minority class, undersampling the majority class, or synthetically generating data for the minority class can help provide a more balanced dataset.

E4. Recommend a course of action for the real-world organizational situation from part A1 based on your results and implications discussed in part E2.

My research question described in part A1 is "Which customer factors contribute most to a customer's decision to churn?"

As previously described in Part E2, the dominant features that result in Chrun are Tenure, Bandwidth_GB_Year, and MonthlyCharge. This tells us that Tenure is the most significant predictor of churn, implying that the length of time customers stay with the service strongly influences their likelihood to continue or discontinue the service. This correlation might stem from various factors such as accumulated benefits, greater satisfaction, or the inertia that comes with longer-term engagements.

An organization should therefore focus on customers with shorter tenure by implementing loyalty programs or offering incentives to increase their stickiness.

Additionally, Bandwidth_GB_Year is highly influential, suggesting that how much data a customer uses and how much they are charged each month are critical in their decision to churn. The importance of Bandwidth_GB_Year indicates that a customer's service usage, particularly data consumption, is a key indicator of their likelihood to continue or discontinue the service. High usage likely correlates with greater reliance on or satisfaction with the service.

This insight suggests that companies should segment customers based on usage and tailor communications and promotions accordingly.

Lastly, MonthlyCharge is another strong predictor of Customer Churn. The perception of getting good value for money plays a critical role in retention, pointing to the need for competitive and transparent pricing strategies. Regularly analyzing competitors' pricing and adjusting offers accordingly can keep an organization competitive.

G. Web Sources

What Is Random Forest? – https://careerfoundry.com/en/blog/data-analytics/what-is-random-forest/

Random Forest in Python – https://towardsdatascience.com/random-forest-in-python-24d0893d51c0

Introduction to Random Forest in R – https://www.simplilearn.com/tutorials/data-science-tutorial/random-forest-in-r

H. Works Consulted

Sirikulviriya, N., & Sinthupinyo, S. (2011, May). Integration of rules from a random forest. In International Conference on Information and Electronics Engineering (Vol. 6, pp. 194-198).