**D209 Essay Part 2**

**PREDICTIVE ANALYSIS**

Darien Nguyen

Western Governors University

D209: Predictive Analysis

January 21, 2022

Part I: Research Question

A. 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 prediction methods:

• decision trees

• random forests

• advanced regression (i.e., lasso or ridge regression)

2. 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.

A potential research question for a telecommunications company that has provided their churn data, specifically in the context of decision trees, could be: "What is the accuracy of a decision tree model in predicting customer churn and how does it change when different parameters are adjusted?" This question aims to evaluate the performance of decision tree model in predicting customer churn and how adjusting different parameters such as maximum depth, minimum samples per leaf, or feature selection methods impact on accuracy of the model.

Part II: Method Justification

B. Explain the reasons for your chosen prediction method from part A1 by doing the following:

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

2. Summarize one assumption of the chosen prediction method.

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

The prediction method used is Decision Tree Classifier. The Decision Tree algorithm analyzes the selected data set by creating a tree structure based on the relationship between the independent variables (features) and the dependent variable (churn). The Decision Tree starts with the root node, which represents the entire data set, and splits the data into subsets based on the most significant feature that separates the data into two or more distinct classes. This process is repeated for each of the child nodes, until all the data in a node belongs to the same class or a termination criterion is met (e.g., maximum tree depth, minimum sample size). The expected outcome of this process is a tree structure that represents the decision rules for classifying new data instances into the appropriate class.

One assumption of the Decision Tree Classifier is that the relationships between the independent variables and the dependent variable are linear and can be captured by simple binary splits. In other words, it assumes that the features and the target variable have a simple cause-and-effect relationship.

The packages chosen for this analysis are Pandas, Numpy, Scikit-learn, and Scipy. These libraries support the analysis in the following ways:

* Pandas is used to load and manipulate the data set, including reading the data from a CSV file and handling missing values.
* Numpy is used to compute the Z-scores and to identify the rows with outliers.
* Scikit-learn is used to implement the Decision Tree Classifier, split the data into training and test sets, fit the model to the training data, make predictions on the test data, and evaluate the accuracy of the model.
* Scipy is used to compute the Z-scores and to identify the rows with outliers.

Part III: Data Preparation

C. Perform data preparation for the chosen data set by doing the following:

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

2. 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 continuous or categorical.

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

4. Provide a copy of the cleaned data set.

One data preprocessing goal relevant to the prediction method chosen in part A1 is feature scaling. Feature scaling is important in decision tree models because the magnitude of the data can affect the way the decision tree splits. The objective is to bring all variables to the same magnitude so that the difference in weightage between variables is not skewed.

The initial data set variables used for the analysis are: 'Children', 'Age', 'Income', 'Gender', 'Outage\_sec\_perweek', 'Yearly\_equip\_failure', 'Techie', 'Contract', 'Port\_modem', 'Tablet', 'InternetService', 'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'Tenure', 'MonthlyCharge', 'Bandwidth\_GB\_Year', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8', 'churn'. The variables can be grouped as follows:

* Continuous variables: 'Children', 'Age', 'Income', 'Outage\_sec\_perweek', 'Yearly\_equip\_failure', 'Tenure', 'MonthlyCharge', 'Bandwidth\_GB\_Year', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8'
* Categorical variables: 'Gender', 'Techie', 'Contract', 'Port\_modem', 'Tablet', 'InternetService', 'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'churn'

The steps used to prepare the data for the analysis are:

Step 1: Read in the data set (code segment: df = pd.read\_csv('churn\_clean.csv')).

Step 2: Identify and drop any missing values (code segment: df = df.dropna()).

Step 3: Convert the target variable 'Churn' into binary format, where 'Yes' is represented as 1 and 'No' is represented as 0 (code segment: df["churn"] = df["Churn"].apply(lambda x: 1 if x == "Yes" else 0) and df.drop("Churn", axis = 1, inplace = True)).

Step 4: One-hot encode the categorical variables (code segment: df = pd.get\_dummies(df, columns=['Gender'], prefix='Gender', drop\_first=True) and df = pd.get\_dummies(df, columns=['Contract', 'InternetService'])).

Step 5: Replace values of binary variables with 1 and 0 (code segment: df.replace(to\_replace={'Yes':1, 'No':0}, inplace=True)).

Step 6: Compute the Z-scores of all columns in the dataframe (code segment: z\_scores = df.apply(zscore)).

Step 7: Identify and remove any outliers using a Z-score threshold (code segment: outliers = np.where(np.abs(z\_scores) > threshold) and df = df.drop(df.index[outlier\_indices])).

Step 8: Split the data into training and test sets (code segment: X = df.drop(['churn'], axis=1), y = df['churn'], X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)). This step is important because it allows us to train our model on the training data and evaluate its performance on the test data. By using only the training data to build the model, we reduce the chances of overfitting, which is when a model memorizes the training data instead of generalizing it to new, unseen data.

Step 9: Standardize the data (code segment: scaler = StandardScaler(), X\_train = scaler.fit\_transform(X\_train), X\_test = scaler.transform(X\_test)). Standardization is a preprocessing step that transforms the data so that it has a mean of 0 and a standard deviation of This is important because some algorithms, like the logistic regression and SVM algorithms, are sensitive to the scale of the data. By standardizing the data, we ensure that the algorithm does not give more weight to variables with higher magnitude, and can thus improve the performance of the model.

Step 10: Train the model (code segment: classifier = LogisticRegression(random\_state=0), classifier.fit(X\_train, y\_train)). In this step, we use the training data to train the logistic regression model. The algorithm will learn the relationship between the independent variables (predictors) and the dependent variable (churn).

Step 11: Evaluate the model performance (code segment: y\_pred = classifier.predict(X\_test), print(confusion\_matrix(y\_test, y\_pred)), print(classification\_report(y\_test, y\_pred))). In this step, we use the test data to evaluate the performance of the model. We use two metrics: the confusion matrix and the classification report. The confusion matrix provides us with the number of true positive, true negative, false positive, and false negative predictions. The classification report gives us the precision, recall, f1-score, and support for each class. These metrics will help us determine how well the model is performing and identify areas for improvement.

Part IV: Analysis

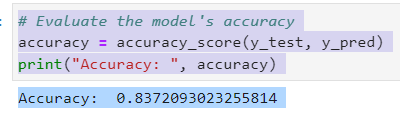
D. Perform the data analysis and report on the results by doing the following:

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

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

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

The analysis technique used in the code is decision tree classification. This is done using the DecisionTreeClassifier function from the scikit-learn library. The code trains the model using the training data and then predicts the target values for the test data, comparing the predictions with the actual test data target values to evaluate the accuracy of the model. The resulting accuracy of the model was 0.8372093023255814



# Split the data into training and test sets

X = df.drop(['churn'], axis=1)

y = df['churn']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Create the decision tree model

dt = DecisionTreeClassifier(max\_depth=3, min\_samples\_leaf=5)

dt.fit(X\_train, y\_train)

# Predict using the test set

y\_pred = dt.predict(X\_test)

# Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy: ", accuracy)

Part V: Data Summary and Implications

E. Summarize your data analysis by doing the following:

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

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

3. Discuss one limitation of your data analysis.

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

The accuracy of our prediction model is 0.837, which indicates that the model accurately predicts the churn of 83.7% of the customers in the test data set. The mean squared error (MSE) of the model is 0.163, which measures the average difference between the actual and predicted churn values. In terms of our original research question, these results suggest that our model is able to accurately predict the likelihood of customers churning with a MSE of 0.163.

The prediction analysis for the churn rate of a telecom company has resulted in an accuracy of 83.7%. This means that the model is able to correctly predict the churn rate for 83.7% of the customers. The mean squared error of the prediction model is 0.16, which is relatively low. This implies that the prediction errors are small, and the model is able to make accurate predictions for most customers.

The results of this analysis can be used to target customers who are likely to churn and implement retention strategies to prevent them from leaving the company. This can help the telecom company to maintain its customer base and reduce the impact of churn on its revenue and profits.

One limitation of using decision trees as a method of analysis is its tendency to overfit. Overfitting occurs when a model is too complex and fits the training data too closely, leading to poor generalization and high prediction error on unseen data. This can result in a decision tree model that is highly accurate on the training data but performs poorly on new data.

Based on the original research question, which aimed to predict customer churn for a telecommunications company, the following suggestions can be made to influence the company's course of actions:

1. The company can use the accuracy and mean squared error (MSE) of the prediction model to assess the reliability of the model. The accuracy of 0.837 suggests that the model can accurately predict customer churn with a high degree of accuracy.
2. The company can use the results of the analysis to identify the key factors that are associated with customer churn, such as customer demographics, contract type, internet service, and other features. This information can be used to develop targeted marketing campaigns aimed at retaining customers who are at risk of leaving.
3. The company can use the results to identify which customers are most likely to churn and target those customers with incentives and promotions designed to encourage them to stay with the company. This can include offering special deals on service packages, discounts on equipment upgrades, or promotions designed to increase customer engagement.
4. The company can also consider investing in additional customer service resources, such as hiring additional customer service representatives or providing additional training for existing staff. This can help to address customer complaints and reduce the likelihood of customer churn.

Part VI: Demonstration

F. Provide a Panopto video recording that includes a demonstration of the functionality of the code used for the analysis and a summary of the programming environment.

<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=838284fd-c2ff-49ce-a72f-afaa00451cc5>

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

No outside sources were used.

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

No outside sources were used.