**Research Question**

How accurately can a Naive Bayes model predict whether a customer has discontinued service in the last month based on service usage, satisfaction, and demographic data?

**Data Analysis Goal**

The goal is to develop a Naive Bayes model that accurately classifies customers as having discontinued service or not within the last month. By analyzing service usage, customer satisfaction, and demographic data, this model aims to identify key factors contributing to customer churn and provide actionable insights to improve customer retention efforts.

**Method Justification**

The Naive Bayes classification method analyzes the dataset by applying Bayes' theorem, which calculates the probability of a customer churning based on the features in the dataset. It assumes that the features are conditionally independent given the class label, in this case, whether a customer will churn. The model takes input features such as customer demographics, usage patterns, and account details. It calculates the prior probabilities of each class churn or not churn, and the likelihood of the features given the classes and for each customer, it combines these probabilities to predict the class with the highest posterior probability. The expected outcome is a set of probabilities indicating whether each customer is likely to churn. Customers can be classified as "likely to churn" or "not likely to churn," enabling targeted interventions by the organization.

A key assumption of Naive Bayes is that all features are conditionally independent given the class label. This means that the presence or absence of a particular feature does not affect the presence or absence of other features when predicting the class. While this assumption simplifies the computation, it may not hold true in all real-world scenarios, potentially affecting the model's accuracy. The Scikit-learn library provides a robust implementation of the Naive Bayes algorithm, allowing easy model training and evaluation. It includes tools for splitting the dataset into training and test sets, which helps in validating the model's performance. The Pandas library is essential for data manipulation and analysis. It allows for easy loading, cleaning, and preprocessing of the dataset, including handling missing values and encoding categorical variables.

The NumPy library supports numerical operations and is used alongside Pandas for efficient data manipulation, especially for our calculations related to probabilities and statistics.

Matplotlib/Seaborn visualization libraries help in understanding the data and the results of the model. They are used to visualize feature distributions, correlations, and the performance metrics of the classification model and the Statsmodels, is beneficial for evaluating model assumptions and understanding the statistical significance of predictors. These libraries collectively provide the necessary tools for data preprocessing, modeling, evaluation, and visualization, ensuring a comprehensive analysis of customer churn using Naive Bayes.

**Data Preparation**

The primary goal of data preprocessing for the Naive Bayes classification method is to ensure that the data is clean, well-structured, and in a suitable format for analysis. This involves handling missing values, converting categorical variables to numerical representations, and normalizing or scaling features if necessary. The objective is to enhance the quality of the dataset to improve the accuracy of the predictive model. The initial data set variables that we will use to perform the analysis for our classification question are our numeric variables MonthlyCharge(float64), Bandwidth\_GB\_Year(float64), Age(int64), Tenure(float64), Item1(string), Item2(string), Item3(string), Item4(string), Item5(string), Item6(string), Item7(string), Item8(string) and our categorical variables Contract(string), InternetService(string), OnlineSecurity(string), PaymentMethod(string).

We bring in our data with the pd.read\_csv() to read in our data file and pd.DataFrame(df) turns the data into a data frame for us, df.head() gives us a quick view of the top five rows to make sure our data was pulled in correctly. The df.info() allows us to get insight into the data, showing the index, column names, non-null count, and data types for each feature. The df.describe() allows us to see the statistical descriptions of our dataset, df.dtypes shows us the data type of each feature in the dataset. We then move on to checking for null values using df.isnull().sum() to get a count of missing values for each feature column. We have no missing values to account for, so we move on to checking for duplicates with df.duplicated().sum().

We will be using customers service data and customer satisfaction data in our model so to improve understanding; satisfaction survey column names were changed using df.rename(). Instead of using Item1- Item8 as column names, each item has been updated to represent what the survey question refers to. We will be using the satisfaction variables as categories based on customer responses, so we convert the columns to strings using df[column].astype(dtype). We then move into addressing outliers in the data. We are going to replace numerical outliers with the median value of each column, and we used the quartile range lower and upper bounds to identify outliers df[column] = np.where((df[column] < lower\_bound) | (df[column] > upper\_bound), median\_value, df[column]). Finally, we encode our categorical variables X\_encoded = pd.get\_dummies(X, columns=[‘Contract’, ‘InternetService’, 'TimelyResponse','TimelyFixes', 'TimelyReplacements', 'Reliability', 'Options', 'RespectfulResponse', 'CourteousExchange', 'EvidenceOfActiveListening', 'OnlineSecurity', 'PaymentMethod'], drop\_first=True). Now we have a cleaned dataset with no missing values, no duplicates, addressed outliers, and encoded values ready for modeling.

**Analysis**

In our analysis, we employed the Naive Bayes classification technique to predict customer churn based on various features extracted from the dataset. Naive Bayes is a probabilistic classifier that applies Bayes' theorem, assuming that the features are conditionally independent given the class label. This assumption allows us to calculate the posterior probability of each class (churn or no churn) efficiently. We utilized the Gaussian Naive Bayes variant, which is particularly effective for continuous features, as it assumes that the features follow a Gaussian (normal) distribution.

Before fitting the model, we preprocessed the data by splitting it into training and test sets, ensuring a balanced representation of churn classes in both subsets. This step is critical to avoid bias in model evaluation. We also encoded categorical variables using label encoding to convert them into a format suitable for model training. After fitting the model on the training data, we assessed its performance using the test dataset, calculating key metrics such as accuracy, precision, recall, and F1-score. These metrics provide insights into the model's ability to correctly classify customers as likely to churn or not. The confusion matrix was also utilized to visualize the performance and identify misclassifications. Overall, Naive Bayes is particularly advantageous in this context due to its simplicity and efficiency with large datasets, and effectiveness in handling categorical data, making it suitable for our churn prediction analysis.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph of a curve

Description automatically generated

**Data Summary and Implications**

The accuracy of 82% indicates that the model correctly predicts churn status for 82% of the customers in the test dataset. This is a relatively strong performance, suggesting that the model effectively captures the patterns related to customer churn. The classification report provides several key metrics for both non-churn(0) and churn(1). Non-churn(0) had a precision score of  0.94.This means that 94% of the customers predicted as non-churn did not churn. Churn(1) had a precision of 0.62 indicating that 62% of the customers predicted to churn did churn. This lower precision suggests that the model may be misclassifying some churn cases.

Our models recall score for non-churn(0) is 80% meaning 80% of actual non-churn customers were correctly predicted. The recall score for churn(1) customers was 73%. This reflects a good balance, but the lower score compared to non-churn indicates some room for improvement in accurately predicting churn. Our confusion matrix results are as follows

True Negatives (1165): Correctly predicted non-churn customers.

False Positives (291): Non-churn customers incorrectly predicted as churn.

False Negatives (71): Churn customers incorrectly predicted as non-churn.

True Positives (473): Correctly predicted churn customers.

The model shows strong overall accuracy, particularly in identifying non-churn customers, which is crucial for retaining customers. However, the lower precision for the churn class indicates that there may be several false positives, which can lead to unnecessary retention efforts or resource allocation.

Our models AUC(Area Under the Curve) value of 0.9109 indicates the model has an excellent ability to distinguish between the customers who churn and the customers who do not churn. Specifically, the AUC represents the probability that a randomly selected positive instance will have a higher predicted probability of being positive than a randomly selected negative instance. Our high AUC score combined with the accuracy score suggests that our Naive Bayes model is well-suited for predicting customer churn. This performance indicates that the model can effectively identify customers at risk of churning, allowing the organization to take proactive measures to improve customer retention.

**Analysis Limitations**

One limitation of the model is the imbalance in the target variable, where the number of non-churn customers significantly exceeds the number of churn customers. This imbalance can affect precision and may cause the model to favor predicting non-churn over churn, leading to potential misclassifications, also our analysis assumes that the data is representative of future behavior. However, if there are significant changes in the service offerings or external factors affecting customer decisions, the model's predictions may become less relevant. For instance, promotional offers or policy changes could alter churn behavior significantly.

**Recommended Course of Action**

Given the strong recall for churn prediction (87%), it would be beneficial for the organization to focus on targeted retention strategies for customers predicted to churn. We have a dataset of predicted to churn customers obtain from our model. I recommend developing personalized communication strategies for these customers. This may include targeted marketing campaigns, special offers, or exclusive promotions aimed at incentivizing them to stay. Collecting feedback and increasing engagement with these customers can provide more insight in how to improve the customers odds of not churning. Regularly monitoring model performance and updating the model with new data will also help maintain accuracy over time.

**References**

W3schools.com accessed, 07 October 2024, https://www.w3schools.com/python/pandas/default.asp

W3schools.com accessed, 07 October 2024, https://www.w3schools.com/python/numpy/numpy\_array\_iterating.asp.

W3schools.com accessed, 07 October 2024, https://www.w3schools.com/python/matplotlib\_intro.asp

Seaborn.pydata.org accessed, 07 October 2024, https://seaborn.pydata.org/tutorial/function\_overview.html

Scikit-learn.org accessed, 07 October 2024, https://scikit-learn.org/stable/modules/impute.html

Scikit-learn.org accessed, 07 October 2024, https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html

Scikit-learn.org accessed, 07 October 2024, https://scikit-learn.org/stable/modules/generated/sklearn.model\_selection.train\_test\_split.html#sklearn.model\_selection.train\_test\_split

Scikit-learn.org accessed, 07 October 2024, https://scikit-learn.org/stable/api/sklearn.metrics.html#module-sklearn.metrics

Scikit-learn.org accessed 07 October 2024, https://scikit-learn.org/stable/modules/naive\_bayes.html

Scikit-learn.org accessed 07 October 2024, https://scikit-learn.org/stable/modules/generated/sklearn.feature\_selection.chi2.html

docs.scipy.org accessed 07 October 2024, https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest\_ind.html#ttest-ind

Scikit-learn.org accessed 07 October 2024, https://scikit-learn.org/dev/modules/generated/sklearn.metrics.auc.html