CPSC 392: ASSIGNMENT 2

DIYA KULKARNI

INTRODUCTION

STREAMING:

In the first section, I am utilizing business churn data, “streaming.csv”, to predict the probability of whether customers of a streaming service will “churn”, i.e., stop being a customer. In the data, I was given the categorical data of the customer’s self-disclosed gender identity (“gender”), the plan the user is currently subscribed to (“plan”), most common genre of content the user watches (“topgenre”), and the second most common genre of content the user watches (“secondgenre”). I was also given the numeric data of their age (“age”), self-reported annual income (“income”), months subscribed to the service (“monthssubbed”), mean hours of content watched per month (“meanhourswatched”), whether or not the customer is subscribed to your competitor's streaming service (“competitorsub”), number of user profiles associated with the account (“numprofiles”), whether or not the user has cancelled the servce in the past (“cancelled”), whether or not the user has downgraded the service at some point in the past (“downgraded”), whether or not the user purchased their plan as a "bundle" with another service (“bundle”), whether or not the user has an active Kids profile on their account (“kids”), the length of the longest watch session from the user (“longest session”), and finally, whether or not the customer “churned” (“churn”).

By utilizing both Logistic Regression and Gradient Boosting Trees models, I will be able to determine which model is best for the data. If this model is successful, it can help the streaming service evaluate what the key variables are in customer churn. By determining these areas, it can implement solutions to minimize churn.

MUSHROOM:

In this section, I am utilizing mushroom data, “mushroom.csv” to predict the probability of a mushroom being poisonous. In the data, I was given the binary indicator of whether a mushroom was poisonous or not (“class”) followed by categorical data of physical attributes of the mushroom, including cap-shape, cap-surface, cap-color, bruises, odor, gill-attachment, gill-spacing, gill-size, gill-color, stalk-shape, stalk-root, stalk-surface-above-ring, stalk-surface-below-ring, stalk-color-above-ring, stalk-color-below-ring, veil-type, veil-color, ring-number, ring-type, spore-print-color, population, and habitat.

By utilizing Categorical Naive Bayes, K-Nearest Neighbors, and Logistic Regression models, I will be able to determine which model best fits the data. If this model is successful, I can use it to define a function to determine how safe a mushroom is. This would be incredibly useful for mushroom enthusiasts, foragers, and anyone interested in identifying wild mushrooms who can use it to mitigate risks associated with consuming poisonous mushrooms.

METHODS

STREAMING:

After loading the data and all the necessary imports and checking for missing values, the first thing I did was utilize dummy variables to convert 'gender', 'plan', 'topgenre', 'secondgenre', categorical variables, to numeric variables that could be used in my model. Following this, I split the data into 90% training data and 10% testing data. Finally, I Z-scored all of the continuous/interval variables, also converting them to a format that would work with my model, and allow me to compare the different data.

After the pre-processing, I fit my first model, logistic regression, using the data from both the train and test sets, and found the accuracy, recall, precision, and ROC AUC values for both the train and test sets. Finally, I assessed the calibration of the test set in order to understand how well the predicted probabilities of my model aligned with actual outcomes.

Following the logistic regression model, I created a gradient boosting tree model, which is a good way to capture nonlinear relationships between features. By using the GradientBoostingClassifier() function, I built a confusion matrix to assess the accuracy of the predicted probabilities to the true values. Then, similar to the previous model, I found the accuracy, recall, precision, and ROC AUC values for both the train and test sets, ending with the assessment of the calibration of the test set.

MUSHROOM:

After loading the data and all the necessary imports and checking for missing values, the first thing I did was utilize dummy variables to convert all the categorical variables to numeric variables that could be used in my model. Following this, I split the data into 80% training data and 20% testing data. Considering all of the variables were categorical, I did not Z-score the continuous variables.

After the pre-processing, I fit my first model, Categorical Naive Bayes, using the data from both the train and test sets, and found the accuracy, recall, precision, and ROC AUC values for both the train and test sets.

Following the logistic regression model, I created a K-Nearest Neighbors model, which predicts outcomes based on the closest data points in the training set. I used GridSearchCV() to test for the number of neighbors that would produce the best results. I ended up with 5-fold cross validation, then found the accuracy, recall, precision, and ROC AUC values for both the train and test sets.

My final model was a logistic regression model, where I once again found the accuracy, recall, precision, and ROC AUC values for both the train and test sets.

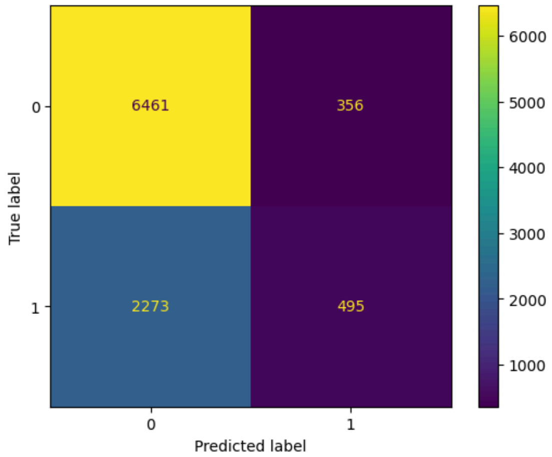
After building these models, I created a function to determine whether a mushroom was poisonous or not based on a list of predictors. The function takes in a list of input features then creates a dictionary to match these features to their names. The dictionary is then converted into a DataFrame which is encoded and reindexed to properly format the data. Finally, the function uses the model's prediction of 0 or 1 to determine if the mushroom is safe.

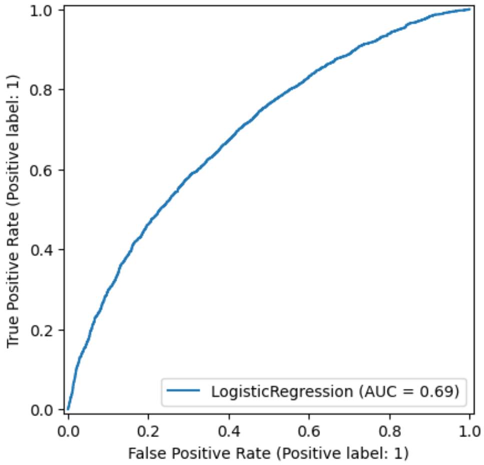
I used 3 calls to the function, one for a safe mushroom, one for a poisonous mushroom, and one for a mushroom with unique/ambiguous features. The function was able to accurately categorize the features for each of these calls.

RESULTS

STREAMING:

Logistic Regression Model:





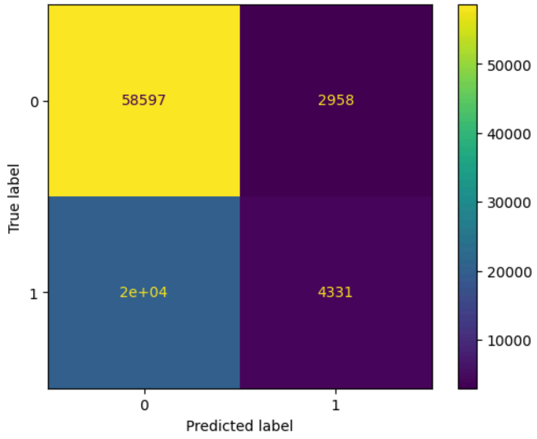
	Accuracy	Recall	Precision	ROC AUC
Train	0.7275878459059345	0.17883743523316062	0.5790301441677589	0.6942804984684492
Test	0.725717266562337	0.17882947976878613	0.581668625146886	0.6944069293783562

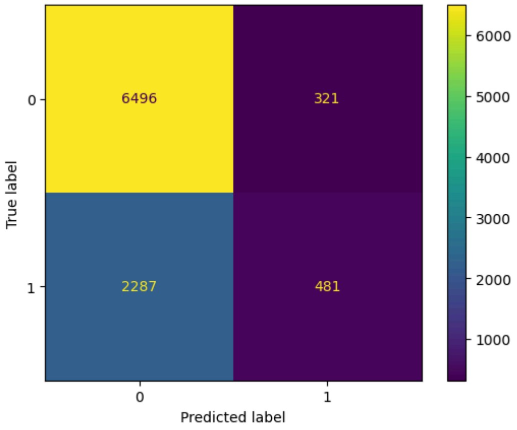
The accuracy of about 72.6% on both the training and test sets indicate that the model is correctly predicting the class about 72.6% of the time. The recall of 17.9% shows that the model is not identifying many poisonous mushrooms, which is a concerning factor. The precision of around 58% indicates that the model correctly predicts a positive class around 58% of the time. The ROC AUC score of 0.69 indicates that the model has a somewhat limited ability to differentiate between classes.

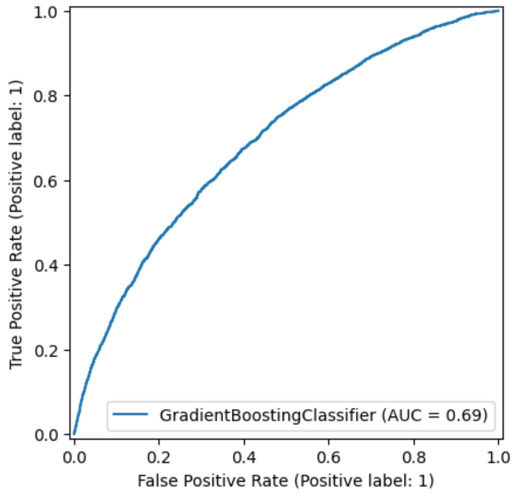
The model's AUC of 0.69 indicates that the model is not very well-calibrated, and the predicted probabilities might not accurately reflect the true likelihood of positive outcomes.

Overall, the model struggles to capture meaningful patterns in the data, and is **underfitting** the data. It is likely to not perform well on future data. I am unsure of how well I trust the results of this model as it is not well-calibrated. I would attempt to use a more well-calibrated model to accurately represent the data.

Gradient Boosting Trees:







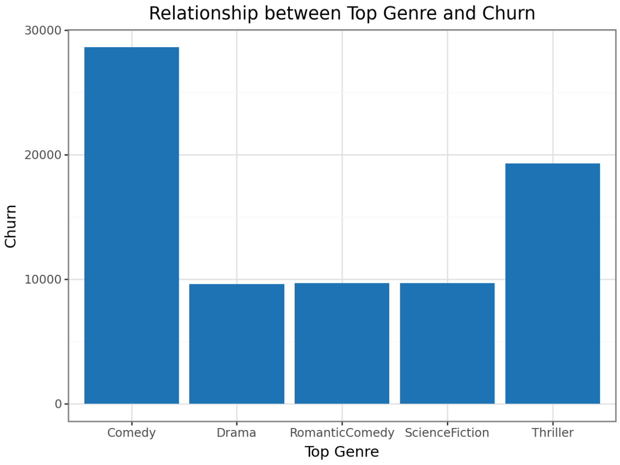
	Accuracy	Recall	Precision	ROC AUC
Train	0.729523875769485	0.17531573834196892	0.5941830155028125	0.6998139738783041
Test	0.7279081898800208	0.17377167630057805	0.5997506234413965	0.6927025082228125

The accuracy of about 72.9% on the training set and 72.8% on the test set indicates the rate that the model is correctly predicting the class. The recall of 17.5% on the training set and 17.4% on the test set shows that the model is not identifying many poisonous mushrooms, which is a concerning factor. The precision of around 59% indicates that the model correctly predicts a positive class around 59% of the time. The ROC AUC score of only 0.56 indicates that the model has a pretty limited ability to differentiate between classes.

Similar to the Logistic Regression model, the AUC of 0.69 indicates that the model is not very well-calibrated and the predicted probabilities might not accurately reflect the true likelihood of positive outcomes.

Overall, the model struggles to capture meaningful patterns in the data, and is **underfitting** the data. Even though both models are incredibly similar in their metrics, the **Gradient Boosting Tree** model is slightly better due to its higher test precision and consistency across training and test sets. This would be the model I would choose to put “in production”.

The pros of using the Gradient Boosting Tree Model are that it has a higher precision and is able to capture more complex patterns in the data. The cons are that it is more complex of a model and may be more expensive for the streaming service for a marginal difference.



In terms of how a customer's top genre impacts whether they are predicted to churn, the graph above shows that there is a higher number of churning amongst customers whose top genre is Comedy or Thriller. My guess is that they are easily bored and need more stimulation from movies. Therefore, if the streaming service is not providing new content consistently, the customer will churn.

My recommendations for the CEO are to try to increase the calibration of the model before using it. Additionally, he should focus more on getting comedy and thriller movies out on the platform to reduce the number of customers who get bored from the service.

MUSHROOM:

Categorical Naive Bayes:

	Accuracy	Recall	Precision	ROC AUC
Train	0.9401446376365594	0.8852459016393442	0.9885139985642498	0.9379004006425766
Test	0.9378461538461539	0.8807453416149068	0.9929971988795518	0.9373238903196485

The accuracy of 94% on the training set and 93.8% on the test set suggests that the model consistently makes correct predictions in many cases. The high recall of 88.5% on the training set and 88% on the test set indicates that the model is effectively identifying a large portion of actual positives. The high precision of almost 99% on both the training and test set means that almost every positive prediction made by the model is correct, avoiding most false positives. The ROC AUC of about 0.94 for both the training and test sets indicate that the model has a strong ability to differentiate between positive and negative cases.

Overall, the model is performing incredibly well and is generalizing well to unseen data, indicating that it is not over or underfitting. It is reliable to use.

K-Nearest Neighbors:

	Accuracy	Recall	Precision	ROC AUC
Train	1.0	1.0	1.0	1.0
Test	1.0	1.0	1.0	1.0

The perfect performance metrics indicate that the model is absolutely perfect for the data. While this may seem like a good thing, it is a strong sign that the model is **overfitting** the data - memorizing it rather than creating patterns to use for future data. As a result, the model is not reliable for new datasets.

Logistic Regression:

	Accuracy	Recall	Precision	ROC AUC
Train	0.9995383905216187	0.9990356798457087	1.0	0.9995178399228544
Test	0.9969230769230769	0.9937888198757764	1.0	0.9968944099378882

The high scores of about 99% for the accuracy, recall, and ROC AUC for both the training and test sets indicate that the model is consistently correct with its predictions and with its ability to identify false positives. Its perfect ROC AUC score shows that it has the ability to perfectly differentiate between classes and does not make a mistake in separating positive and negative cases.

Overall, the model is an almost perfect fit for the data. It contrasts from the K-Nearest Neighbors as it is not completely perfect, minimizing overfitting. This is the **most optimal model** for the data.

Function:
After building the models, I created a function to predict whether a mushroom is safe to consume. The function performed well across different feature sets, suggesting it could serve as a reliable indicator of mushroom safety. I personally believe the function is safe to use, however, it is never possible to be too safe. Especially in a situation where one wrong outcome could lead to fatality, it is important to perform multiple tests using the function before putting it to practice.

DISCUSSION/REFLECTION

Overall, these analyses were incredibly interesting to perform. It was interesting to see the ways that different models could represent the data. It was definitely difficult at points to understand how to best lay out the model but I believe I was able to capture key patterns within the data. I feel as though I have a better understanding of Logistic Regression, Categorical Naive Bayes, K-Nearest Neighbors, and Gradient Boosting Models and I was able to truly interpret performance metrics that indicate the model's performance. Something I would do next time is to maybe try out some of the newer models we have learned about to understand the differences in how they represent the data. I would also try to see if there is a way I could adjust the calibration of the models for those that were not as well-calibrated.