1 Introduction [15 points]

- Group members: Brendan Flaherty, Sujit Iyer, Sayuj Choudhari
- Kaggle team name: Beavers
- Ranking on the private leaderboard:
- AUC score on the private leaderboard:
- Colab link: beaver_mp1.ipynb
- Piazza link: Beaver's MP1 Piazza Post
- Division of labor: Worked together on everything in person.

2 Overview [15 points]

Models and techniques attempted

During the project, we used Logistic Regression and Random Forest Classifiers. Along with these models, we also experimented with Lasso Regression to find the subset of features that were most important for predictive modeling. We combined Lasso Regression with biserial correlation to confirm which features we would use to train the data. This made our predictions more accurate and we noticed a distinct increase in test set accuracy. Finally, we experimented with hyper-parameters for regularization such as max depth and minimum leaf size stopping criterion. We found the ideal hyper-parameters by implementing a grid search.

Work timeline

We first started with logistic regression. Once we realized that logistic regression performed poorly on the test set, we pivoted to using RandomTreeClassifier. We then performed feature engineering to determine the most important features, and adjusted parameters on the RandomTreeClassifier using grid search to achieve the best testing classifications. We spent three separate nights perfecting our model and a total of around 12 hours each.

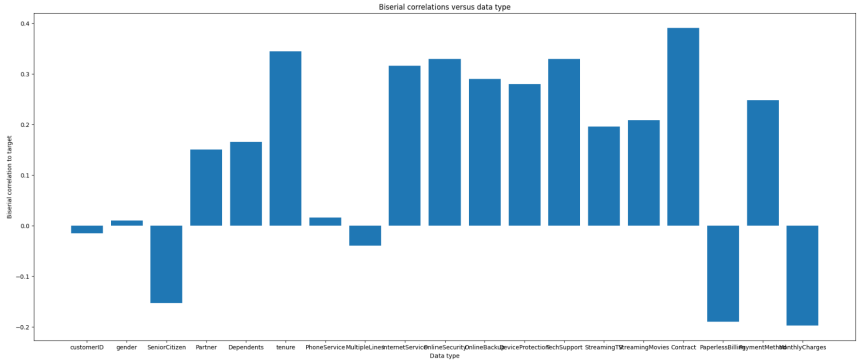
3 Approach [20 points]

Data exploration, processing and manipulation

We noticed that the data given was in-homogeneous, presenting an issue for models that expected numerical data. Columns such as 'Senior Citizen', 'tenure', 'MonthlyCharges', and 'TotalCharges' were numerical. However, the rest of the columns were string data. Using the pandas factorize function, we factorized the data to be entirely comprised of integers. Throughout the project, we used pandas data frame objects to handle training and test data representations. This made it convenient to quickly extract the feature set (X) and the binary target (Y). After we had sufficiently transformed the data into a suitable form, we attempted to fit a model to it and predict on the test set. However, we were still running into errors. We discovered that the training data column 'MonthlyCharges' had NaN values in it which threw errors when we tried to fit the data to it. We considered using pd.dropna() to remove NaN rows but ultimately removed 'MonthlyCharges' from our feature subset as we ran biserial correlation and Lasso regression tests for feature selection. From this, we found that 'MonthlyCharges' and several other features were not needed to model discontinued service. This concluded our data manipulation.

Details of models and techniques

The two models we used for feature selection were biserial correlations and a lasso (L1 regularized) regression. Given the output was binary, researching various correlation coefficients, the biserial correlation was the most fitting for this problem as it performs well on correlations against binary output data both for when input data is categorical/ binary and continuous. The graphic below shows the bar chart of correlations for all features and validated our use of biserial correlation in that it the results of features such as gender, CustomerID, Streaming movies and tv shows, and use of phone service all had low correlations with the output of discontinuing also make sense logically for the problem. We combined this analysis with an L1 regression of $\alpha = .01$ (larger α's pushed all weights to zero) so the sparsity of the L1 regularization will push weights that may not be needed to 0 (results showed use of features such as Tech Support, Contract, and Internet service type to be most important). We used a combination of both results to determine the set of features for the model, cutting feature space size from 20 to 10 and reducing risk of overfitting on less meaningful features.



The graph above shows biserial correlation coefficients against binary data where output = 1 denotes not cancelling and 0 denotes cancelling so negative correlations imply the feature is correlated with cancelling service.

4 Model Selection [20 points]

Scoring

We first started the project by implementing Logistic regression which has a cross-entropy optimization objective. We started with logistic regression because the project specification said to submit probabilities from the models instead of $0/1$ predictions. Due to the nature of probability prediction, we found it hard to gauge the success of the training data because the 'Discontinued' labels were presented as binary data. We projected these probabilities to their respective [0, 1] classifications to determine training accuracy. We observed a high training accuracy of 0.967, and so we submitted the test set predictions (with probabilities instead of binary classifications) into Kaggle. We ended up with a 0.657 test score, which was far below the TA benchmark. After discussion, we pivoted to using Decision Trees, which are far better with categorical data. Decision Trees use 'Gini impurity' as an optimization objective instead of a continuous objective function that methods such as Logistic Regression and Neural Networks use. We specifically implemented a Random Forest Classifier because using ensemble methods helps improve model generality and implicitly 'validates' the data by selecting random subsets of the training data for decision tree generation. We immediately saw that the Random Forest Classifier was achieving 100% training accuracy given no regularization parameters, a strong sign of overfitting. Therefore to regularize the model, we modified hyper-parameters such as the number of estimators, max depth, and minimum leaf size. This resulted in a 0.82 training accuracy (using binary classification), which presented a middle ground between underfitting and overfitting. We submitted our probability predictions for the test set, resulting in a 0.85 accuracy score.

Validation and test

Although we did not explicitly separate our data into a training and validation set, we implicitly validated the results of our model and prevented overfitting by using a Random Forest Classifier. The Random Forest created random subsets of the data to train on and ensembled the classifiers at the end to make predictions. Random Forests also introduce randomness in the feature selection process, by choosing a random subset of features to train on at every split. This helped reduce the correlation between the trees without significantly increasing the bias. For testing purposes, we implemented a grid search to find the ideal hyperparameters. This helped us find a balance between overfitting and underfitting the data. We verified our findings for optimal hyperparameters with the public Kaggle score. Given more time, we would have split the training set into a validation set to verify our findings without using one of our 10 submissions. This also would have prevented overfitting to the public test set.

5 Conclusion [20 points]

Insights

Based on the analysis conducted for feature selection, contract type and tenure were most correlated according to the biserial correlation coefficient to the binary output target value of 0 or discontinuing service (see graph for visual). Furthermore, the Lasso regression also kept these feature weights at non-zero supporting keeping them in the model's feature space. This logically makes the most sense as customers with longer term contracts (e.g. yearly or 2-years) are less likely to discontinue the service since they've probably signed up for the longer term service with the intention of keeping it for a longer time. Similarly, longer tenured customers are probably most loyal to the service and thus unlikely to discontinue it as well. For negative correlation the paperless billing binary category and monthly charges numerical data are most heavily correlated with discontinuing subscription according to the biserial correlation and further validated by the Lasso regression. These features correlations also make sense with the context of the problem since paperless billing allows for online cancellation/ quicker cancellation process and higher monthly charges will make customers more likely to find alternatives and discontinue the service. A learning point from this part of the process was that a combination of both negatively and positively correlated variables produced the best results rather than cutting off data to just positive or just negatively correlated features. This is likely due to negatively correlated features providing additional information that is not explained by positively correlated features (e.g. the length of contract and use of paperless billing are unrelated to each other and thus the combination of both should expand the information the model is able to use/ decide on.). In terms of the 10 most important features selected, it was 'SeniorCitizen', 'tenure', 'InternetService', 'OnlineSecurity', 'TechSupport', 'StreamingTV', 'Contract', 'PaperlessBilling', 'PaymentMethod', and 'MonthlyCharges' and based on the graph above we can see the predicted direction of each feature's influence. Some learning outcomes of this project were finding aspects of a problem that pointed model selection towards decision tree methods, mainly in that we noticed the binary and categorical data fits with the model logic of decision trees. We also learned a great deal of feature engineering and analyzing the results of feature analyses. Overall, we feel that we have learned the overall process of addressing a machine learning problem through various steps from feature selection, manipulation, and model selection.

Challenges

The main obstacles during this project were processing the data in the way we best saw fit, but these were mainly library function knowledge challenges that were fixed through research. Other obstacles included finding a good fit of parameters to use in the RandomForest classifier which we addressed using a grid search across parameter combinations. While the grid search was not difficult to code, we had to familiarize ourselves with the seaborn library to build a heatmap of the accuracy scores in training so we could better visualize and analyze the grid search. In terms of changes in our process, we would extend our current work to test on further parameters of the RandomForestClassifier and/ or extend our predictions to be a conclusion from multiple methods to reduce errors from certain classifiers and create a better 'average prediction'. Another addition we wanted to our solution process was feature building, where we use given data to build some formula of our own to create our own feature that considers multiple of the features given (e.g. a measure of combining tenure and monthly charges so high tenure customers with high monthly charges are not put in a risk of discontinuity bucket since high tenure is more strongly influential). These notes on obstacles and future directions of work conclude this section.

6 Extra Credit [5 points]

7 LLM Usage

This section has an example of LLM usage reporting below. Please follow this format to report LLM usage. Indicate each usage clearly.

- Name of LLM(s) Used: ChatGPT
- Components of Project involving LLM: Data processing.

Data preprocessing (example)

Interaction (example)

Only include the parts of the response that you integrated into your project. You may submit screenshots. For example, Figure 1 illustrates the interaction with LLM for data preprocessing. Alternatively, you can also write out the prompt and response in text.

Usage/Integration of Responses (example)

As shown in Figure 2, We used the response as follows, modifying the name of the encoder object and applying it to our training data frame data, in column some_column.

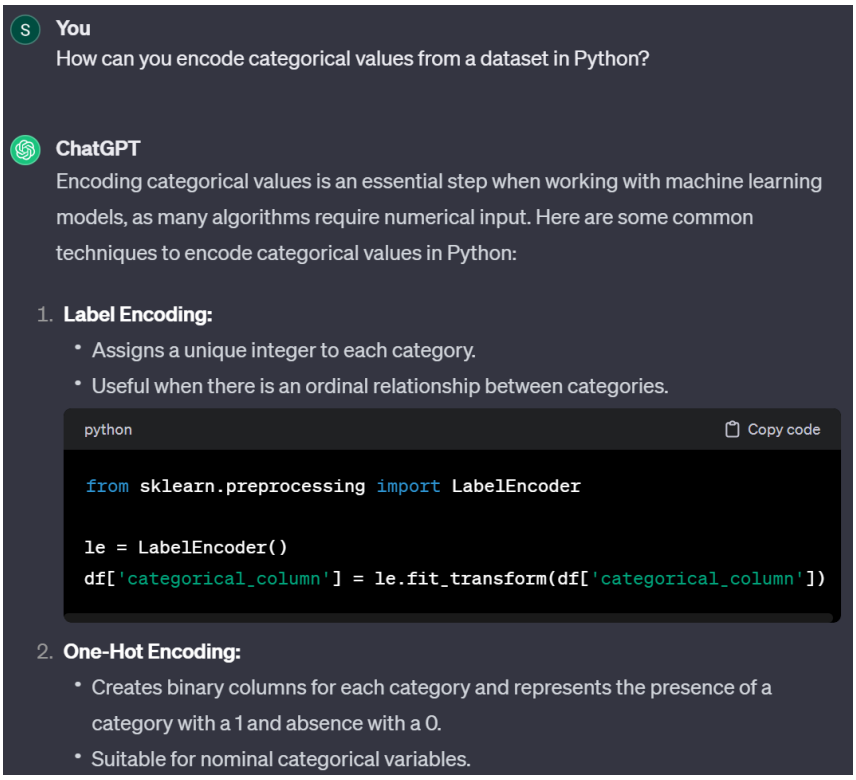


Figure 1: Interaction with LLM for data preprocessing.

```
from sklearn.preprocessing import LabelEncoder

enc = LabelEncoder()
data['some_column'] = enc.fit_transform(data['some_column'])
```

Figure 2: Integration of LLM's output for data preprocessing