Jordan Tompkins

D209, Data mining I task 1

A1, Proposal of Question: The question that was looked at for this analysis was, can we predict whether or not a customer would discontinue their service, or “Churn”, within the last month? To answer this question, the K-Nearest Neighbors classification method was used.

A2, Defined Goal: One goal for this analysis and question is to use k-nearest neighbors, or k-NN, to create a model that can help the company predict which of their customers might churn or not. The goal is to create a model that the company might be able to use to help them accurately predict if customers will churn, so they can determine how and where to focus efforts on trying to keep customers.

B1, Explanation of Classification Method: K-Nearest neighbors, or KNN, “…is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point” (“What is K-Nearest neighbors algorithm?”). This can help predict a data point by looking at the closest “k” labeled data points. For example, if “k” was 3, this method will look at the 3 closest labeled data points and then assign that unlabeled data point the label that was the majority. So if there were 3 points being looked at, and 2 were yes the customer churned and 1 was no, that unlabeled data point would be labeled as yes the customer churned.

B2, Summary of Method Assumption: KNN has a few assumptions, but one of the biggest assumptions is that the data points that are close to each other are similar (Nelson, 2020). KNN uses distance to help determine similarity between the data points and to label the data points, so the further away a point is, the more dissimilar it is. The closer it is, the more similar it is to everything around it.

B3, Packages or Libraries List:

|  |  |
| --- | --- |
| Package and libraries | Usage |
| Pandas | This was used to import that data into the data frame and helped with any data preparation |
| SelectKBest, f\_classif from sklearn.feature\_selection | Using SelectKBest and f\_classif we were able to determine what the best features to use in our model would be. |
| StandardScaler from sklearn.preprocessing | This allows us to scale the features since KNN relies on distance. Because of this we needed to scale the features using StandardScaler |
| Train\_test\_split from sklearn.model\_selection | Train\_test\_split allows us to breakup the data into training sets and testing sets so the whole dataset was not being used for the analysis. It allowed for us to train the model on one set, the training set, and then test it on a separate set, the testing set. |
| Pipeline from sklearn.pipeline | In order to do both KNN and scale the data easier, a pipeline was created. This allowed for us to take our training and testing data, scale it first and then use KNN on the data in a more streamlined and easier to read code. |
| KNeighborsClassifier from sklearn.neighbors | This allows us to perform the KNN analysis. This was used in the pipeline created, along with StandardScaler |
| Roc\_auc\_score from sklearn.metrics | This allows for us to compute the AUC score, which was one of the metrics looked at to determine if the model created through KNN analysis was a good model or not |
| Confusion\_matrix from sklearn.metrics | This was used to create a confusion matrix which then helped determine the accuracy of the model to determine how well the model does at predicting “Churn”. |

C1, Data Preprocessing. One of the main data preprocessing goals related to KNN classification was to make sure that the data was prepped and ready for usage. To do this, we needed to clean the data and make sure that there were no nulls or missing values. The data was checked for any missing or null values, and categorical variables were encoded from the original responses to numeric responses of 1s and 0s.

C2, Data Set variables: The variables used to perform the analysis include “Tenure” which is a continuous variable, “MonthlyCharge” and “Bandwidth\_GB\_Year” which are also continuous variables. The categorical variables used in the analysis include “Multiple\_Yes”, “InternetService\_Fiber Optic”, “InternetService\_None”, and “Gender\_Male”. These were the categorical variables used in the prediction of “Churn\_Yes”, another categorical variable which we were predicting in the analysis.

C3, Steps for Analysis: After loading the dataset using “pd.read\_csv” we had to first ensure that there were no missing or N/A values in the data using “df.isnull().sum()”. After seeing that there were no nulls or N/As, the next step was to drop the initial variables that were deemed unnecessary for the performance of the analysis. To do this, we used “df.drop” and specified the columns that we wanted to initially drop. The majority of the columns dropped were customer demographic specific or survey responses. Those variables were deemed unnecessary so they were dropped. After dropping those columns, dummy variables were created on all the categorical variables in the data set using “pd.get\_dummies()”, setting drop\_first equal to “True”. The data was cleaned and prepped to use but in order to get the best variables for the analysis, the “SelectKBest” method was called on the data set which allowed for the data to then be used to determine which features would be best for KNN analysis. Checking for any features with p-values greater than 0.05, those features were selected for the final analysis. See jupyter notebook for a copy of the code described above.

C4, Cleaned Data Set: See copy of cleaned data set uploaded separately.

D1, Splitting the Data: After cleaning the data and ensuring that it is ready for analysis, the next step was to split the data into training and testing sets. Arrays were first created for the data, using “X = df[features\_to\_keep]” and ‘y = df[‘Churn\_Yes’]”. The X array was all of the predictor variables determined from the preprocessing steps using “SelectKBest”, while the y array was the target variable of “Churn\_Yes”. After creating the arrays, “train\_test\_split” was used, with the two arrays, a test size of 30% and a random seed set to 13 to ensure reproducibility. The test size was set to 30% which meant that we would test 30% of the data in the analysis, and that left 70% of the data to the train, or fit the model. See copies of training and testing sets in files uploaded separately.

D2, Output and Intermediate Calculations: The analysis technique used for this classification analysis was k-nearest Neighbors. The analysis was used to predict whether or not a customer would “Churn”, or discontinue their service within the last month. After all the data was cleaned and prepped for analysis, and the train and testing sets were created, KNeighborsClassifier was imported from sklearn.neighbors, to perform the analysis. To do the analysis, a pipeline was used to help streamline the process. To start off, the pipeline was created with “StandardScaler()” and “KNeighborsClassifer()” being apart of those steps. This allows for the train and test sets to be scaled for usage before being used in KNN analysis. The training sets, X\_Train and y\_train, were then fitted using the pipeline and X\_test was then used to create predictions and test the testing set. After the model was trained and predictions were. Using the testing and training sets, and the predictions created, the accuracy of the model and the area under the ROC, or AUC, were calculated. These scores show how the model does in terms of predictions. See calculations below:

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

D3, Code execution: See jupyter notebook uploaded separately for the execution of the code to perform the analysis.

E1, Accuracy and AUC: Two of the key metrics looked at for this analysis were the accuracy and the AUC. The accuracy represents the percentage of correct predictions compared to the total amount of predictions. To get this you can either use the “.score()” method or calculate it by looking at the confusion matrix and adding the numbers in the [0][0] and [1][1] positions then dividing by the total number of predictions. In this analysis, the accuracy calculated was .85066666, or 85.07%. To create the confusion matrix, confusion\_matrix was initially loaded in and then used on the “y\_test” and “y\_pred” data and it created the confusion matrix seen above. From there you can take the [0][0] and [1][1] spot, which are the top left and bottom right numbers and represent True Negative and True Positive predictions, and add those together and divide by the total of the matrix to get the accuracy of the analysis. This means that the model accurately predicts the “Churn\_Yes” response roughly 85% of the time. The other metric that was used for this analysis was AUC, or area under the curve. A model with an AUC of 0 is a model that 100% of the predictions being incorrect, while a model with 1.0 is a model that has 100% correct predictions. In the model created here, the AUC was .8798 which also shows that the model created was a good model in terms of predictions of “Churn\_Yes”.

E2, Results and Implications: K-nearest neighbors is a very simple algorithm that can help predict whether or not a customer will churn, yet it is highly effective. Using this algorithm, you can obtain certain metrics such as the accuracy of the model created, the best features to look at in order to create more accurate predictions, and so much more in such a simple algorithm. In our model, the accuracy of the model had an accuracy score of .85066, which means that the model accurately predicts whether or not a customer would churn a little over 85% of the time. If a company is looking for accurate predictions, 85% accuracy is a very good score for a model and is an indication that it is one that could be used. We also learned about the best features to look at by using SelectKBest approach for feature selection. This allowed us to simplify the model and ensure that the model was not overfitted by using too many variables, many of which could be highly unnecessary for the model. Looking at the AUC score also gave a clear indication that the model used was a good model. The AUC score was 0.8798 which is a good indication that the model was able to predict correctly. A model of 1.0 has 100% correct predictions, so the model created having a score of 0.8798 shows that it did well. While k-nearest neighbors does offer a simple algorithm to help with a classification problem, it could always be improved by doing things such as hyperparameter tuning to help with selecting the best value of k to use, or the right amount of n-neighbors, etc. There is always room for improvement, but KNN did well in this analysis.

E3, Limitation: While k-nearest neighbors classification is a very simple algorithm to use to help solve classification problems, it does have it’s limitations too. A few limitations of KNN include the fact that the accuracy, one of the key metrics looked at, all depends on the quality of the data (Chatterjee, 2022). If the data quality is poor, KNN might be hard to use. Predictions could be wrong, accuracy could be off, etc. That is why it is important to have good data, and also prepare the data to ensure that it is ready for use. Scaling is also important when using the algorithm. If the data is not scaled properly, you could end up getting incorrect results. There are many great things about k-nearest neighbors classification, but these limitations are a couple that need to be considered when deciding on what method to use to help solve the classification problem.

E4, Course of Action: There is a lot that the company could do with this information. The model has an accuracy of a little over 85%, so if the company decided to use this model, they would be able to predict if customers will churn pretty accurately. If they wanted to, they could use hyperparameter tuning to create an even better model, select more or better features, etc. If they just wanted to use the model created, they can do that too with little doubt that they can get accurate predictions. They can then focus their attention on improving certain aspects of the business if the notice any common features that tend to be associated with high churn rates, etc.

F, Panopto Recording: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=f9831c60-7bab-4a8b-9927-afda01748702>

G, Sources for Third-Party Code: No third-party code utilized.

H, Sources:

Chatterjee, M. (2022, December 13). *A quick introduction to KNN algorithm*. Great Learning Blog: Free Resources what Matters to shape your Career! Retrieved April 8, 2023, from https://www.mygreatlearning.com/blog/knn-algorithm-introduction/

Nelson, D. (2020, August 23). *What is a KNN (K-nearest neighbors)?* Unite.AI. Retrieved April 8, 2023, from https://www.unite.ai/what-is-k-nearest-neighbors/#:~:text=The%20primary%20assumption%20that%20a,two%20points%20on%20a%20graph.

*What is the K-nearest neighbors algorithm?* IBM. (n.d.). Retrieved April 8, 2023, from https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point.