**Naïve Bayes Classification of Customer Churn**

Kelseyann Wright

Student ID: 00565814

College of Information Technology, Western Governors University

Data Mining I - D209 Task 1

Dr. Festus Elleh

June 6, 2023

**Naïve Bayes Classification of Customer Churn**

**A1. PROPOSAL OF QUESTION**

The question that will be addressed is, "Can a Naïve Bayes classifier be used to predict telecom customers at risk of churn using categorical feature data?". This research question is important because predicting if a customer is at risk of churn will allow the company to take steps to retain those customers.

**A2. DEFINED GOAL**

The main goal of this analysis is to develop a machine learning model using Naïve Bayes classification to predict the likelihood of customer churn given appropriate categorical feature data.

**B1. EXPLANATION OF CLASSIFICATION METHOD**

Naïve Bayes classification is a technique based on Bayes Theorem, which describes the probability of an event given prior knowledge of features related to the event (Awan, n.d.). Naïve Bayes classification is a simple supervised learning algorithm with high speed and accuracy when dealing with large datasets (Awan, n.d.). Classification using Naïve Bayes will result in probability predictions for a categorical variable (i.e., customer churn) based on the likelihood probabilities of feature variables in each class of the categorical variable (Awan, n.d.). Naïve Bayes can be used with multiple features while being fast and accurate, making it an appropriate tool for the posed research question.

**B2. SUMMARY OF METHOD ASSUMPTION**

One assumption of the Naïve Bayes classification method is that the effect of each feature in the model is independent of the other features (Awan, n.d.). This means that all features are treated as independent of one another in the analysis, even if they are codependent, resulting in the term “naïve” to describe the model (Awan, n.d.). If features are not independent of each other, meaning they are based on the same or related information, the algorithm will duplicate the effect of the information and may reach an incorrect prediction. This assumption should be considered when identifying features to include in Naïve Bayes classification models.

**B3. PACKAGES OR LIBRARIES LIST**

The libraries used in the analysis are described in Table 1.

***Table 1: Justification of python libraries used in the analysis.***

|  |  |  |
| --- | --- | --- |
| Library | Justification / Use | Source |
| Pandas | Data frame manipulation and summarization | McKinney, 2010 |
| numpy | Data manipulation and formatting for statistical analyses | Harris et al., 2020 |
| matplotlib | Data visualization | Hunter, 2007 |
| seaborn | Data visualization | Wascom, 2021 |
| imblearn | Balancing data using SMOTE algorithm | Lemaître, Nogueira, & Aridas, 2017 |
| Sci-kit learn | Includes a function for splitting data into train and test sets and a function for executing the Naïve Bayes algorithm. This library also includes multiple functions for testing model accuracy. | Pedregosa et al., 2011 |
| scipy | Chi-square test for independence | Virtanen et al., 2020 |

**C1. DATA PREPROCESSING**

One data preprocessing goal is to identify related (codependent) features and remove them from the dataset to ensure that variables met the independence assumption of the Naïve Bayes classifier. To achieve this goal, chi square tests for independence will be used to compare all features and identify statistically significant relationships according to a P value less than 0.05. Removal of features that do not meet the Naïve Bayes independent assumption is important to ensure model prediction accuracy.

**C2. DATA SET VARIABLES**

The dataset variables used in the analysis are Area, Marital, Gender, Churn, Techie, Contract, Port\_modem, InternetService, Phone, Multiple, OnlineSecurity, StreamingTV, StreamingMovies, PaperlessBilling, and PaymentMethod. All variables are categorical. The dependent variable is ‘Churn’, while all other variables are predictor (independent) variables.

**C3. STEPS FOR ANALYSIS**

Continuous variables were removed from the dataset, as the research question is focused on the use of categorical variables. In addition, categorical variables with high cardinality, specifically more than five groups, were dropped from the dataset. Variables with high cardinality can result in the “curse of dimensionality”, which results in sparse data in each unique group. This can reduce overall model performance because there are not enough data points in each unique group. In addition, high cardinality features can take up a lot of memory during model fit and cause computation issues. Therefore, those variables with high cardinality were removed to avoid issues with dimensionality and computer memory.

Chi-square tests for independence were run for all combinations of independent variables to identify correlated variables. The variables that were removed due to significant chi square statistics (P<0.05) were ‘OnlineBackup’, ‘DeviceProtection’, ‘Tablet’, and ‘TechSupport’. The removal of related variables is necessary to meet the independence assumption of the Naïve Bayes classifier discussed in section C1.

The remaining variables were transformed into dummy variables for the analysis. Dummy variables are binary values of 0 or 1, where 0 represents "no" or not being a member of that group and 1 represents "yes" or being a member of the group. Using dummy variables prevents us from having to make multiple models to represent different groups (Garavaglia and Sharma, 1998).

Finally. The initial dataset was imbalanced, with there being about one third as many “churn” customers as “no churn” customers (Figure 1). The data was balanced using the **A picture containing text, screenshot, rectangle, diagram

Description automatically generated**synthetic minority oversampling technique (SMOTE), as balanced data is important for logistic regression (Brownlee, 2020).

Figure 1: Countplot of customer churn, where 0 is “no” and 1 is “yes”.

The annotated Python code containing all data preprocessing is included in the attached python script titled “D209\_T1.py”.

**C4. CLEANED DATA SET**

The final cleaned dataset is attached in the file “churn\_data\_processed\_T1.csv”.

**D1. SPLITTING THE DATA**

**The submission provides reasonably proportioned training and test data sets.**

The “train\_test\_split” function from the sci kit learn library in python was used to split the data into training and test datasets. The test set was set to 30% of the original dataset, with the remaining 70% being used in the training dataset. All datasets were exported to csv files and are attached in the files “Xtrain\_T1.csv”, “Xtest\_T1.csv”, “ytrain\_T1.csv”, and “ytest\_T1.csv”.

**D2. OUTPUT AND INTERMEDIATE CALCULATIONS**

After training the classification model with the training datasets, the model accuracy was assessed using the testing datasets. Model predictions were run using the test feature dataset (Xtest\_T1.csv). The resulting model predictions were then compared to the classification test dataset (ytest\_T1.csv). To compare model predictions to the test data, the sci-kit learn “metrics” module was used (Pedregosa et al., 2011). This module includes functions to calculate the accuracy score, AUC, confusion matrix, and F-score of the model.

A screenshot of the confusion matrix comparing model predictions to the test dataset is shown below. Overall, there are more true positive and true negative values than false positive and false negative values, respectively.



The “classification\_report” function in the sci-kit learn metrics module provides more information on model accuracy by class. A screenshot of the output is provided below.

A screenshot of a computer screen

Description automatically generated with low confidence

In the classification report, the precision represents the percent of correct positive predictions relative to total positive predictions. In other words, , where “TP” is true positive and “FP” is false positive. The precision shows that of the customers the model predicted would churn (class = 1), only 66% actually did.

Recall represents the percent of correct positive predictions. , where “TP” is true positive and “FN” is false negative. This model shows that the model correctly identified 79% of the customers that did churn (class = 1). The F1-score is a calculated using both the precision and recall values using the equation:

For the F1-score, the closer the value is to 1 the better the model. The F1 score is slightly higher for the churn predictions (class = 1) than the no churn predictions (class = 0). The overall F1 score for the model is 0.69, which is okay but there is room for model improvement.

**D3. CODE EXECUTION**

The python script for all data preparation and analysis was attached (D209\_T1.py). The python script was written and executed in Visual Studio Code.

**E1. ACCURACY AND AUC**

Both the accuracy and AUC were calculated using the sci-kit learn metrics module (Pedregosa et al., 2011). A screenshot of the output of both scores is provided below.

A black screen with white text

Description automatically generated with low confidence

Accuracy is calculated as the fraction of correct predictions. The equation for accuracy is: , where “TP” is true positive, “TN” is true negative, “FP” is false positive, and “FN” is false negative. The overall accuracy of the model is 0.69, meaning that 69% of model predictions were correct. This is not bad, but the model could be improved.

AUC is calculated as the area under the receiver operating characteristic (ROC) curve. The closer the AUC is to 1, the better them model is at separating and identifying different classes. The AUC of this model is 0.79, which is a good AUC. This means the model is good at making predictions, but there is still room for some improvement.

**E2. RESULTS AND IMPLICATIONS**

The question addressed with this work was, “Can a Naïve Bayes classifier be used to predict telecom customers at risk of churn using categorical feature data?”. According to the results, we can create a Naïve Bayes classifier using only categorical feature data to predict customer churn with 69% accuracy. This model could be used to identify customers at risk of customer churn, but the telecom company should know that the model predictions are not 100% accurate and may identify customers who are not at risk of churning, and it may miss identify customers who are at risk of churn as those that will not churn. Therefore, caution and understanding of the model error should be used if the model is implemented in its current state. The lower accuracy of this model implies that there may be some other features that are not included in the model that may be necessary for better predictions of customer churn.

**E3. LIMITATION**

One limitation of this data analysis is the use of only independent categorical variables as features for prediction the classifier. Using the Naïve Bayes classifier limits us to the use of unrelated categorical variables due to the assumption of independence. In addition, the research question was limited to categorical variables. Limiting features incorporated into the model for both reasons may have resulted in exclusion of important features. These important features, if included, may have improved model predictions.

**E4. COURSE OF ACTION**

Based on the model outcome, improvements need to be made in the classifier before it should be used for business predictions. As the classifier has below 70% accuracy, there is room for improvement prior to use. The company may use the classifier in its current state while improvements are being made, but it should be noted that the accuracy is 69%% and that incorrect classifications, both false positive and false negative, are possible and likely to occur.

The initial accuracy of 69% shows promise and suggests that there are some valuable features informing current model classifications. Feature optimization should be used to improve the model classifications, with a target accuracy of at least 80%. Improving the classification accuracy of the model will provide the business with more confidence when trying to predict the likelihood of customer churn for new and current customers. This will be important so that the company can work efficiently, targeting retention of customers that are likely to churn. Using a model with low accuracy could result in the company focusing their efforts on false positive customers (customers that the model identified as likely to churn that would not have churned). This would be a waste of company resources. Hence, it is crucial to improve model classifications as a first step before implementing the model predictions on a large scale.

**F. PANOPTO RECORDING**

The Panopto link was attached. The link is:

**G. SOURCES FOR THIRD-PARTY CODE**

No additional web sources of data or third-party code were used.

**H. SOURCES**

Awan, A. (n.d.) *Naive Bayes Classification Tutorial using Scikit-learn*. DataCamp. <https://www.datacamp.com/tutorial/naive-bayes-scikit-learn>

Garavaglia, S., & Sharma, A. (1998). A smart guide to dummy variables: Four applications and a macro. In Proceedings of the northeast SAS users group conference (Vol. 43).

Harris, C.R., Millman, J.K., van der Walt, S.J, Gommers, R., Virtanen, P., Cournapeau, V., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., Fernandez del Rio, F., Wiebe, M., Peterson, P., Gerard-Marchant, P., Sheppard, K., … Oliphant, T.E. (2020). Array programming with NumPy. Nature, 585(7825), 357-362.

Hunter, J.D. (2007). "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95.

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, *18*(1), 559-563.

McKinney, W. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).

Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., and Dubourg, V. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., … van Mulbregt,P. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261-272.

Waskom, M. L. (2021). Seaborn: statistical data visualization. Journal of Open Source Software, 6(60), 3021.