# D209 Data Mining I

Professor Keiona Middleton

Mackenzie Simon

# Part 1 Research Question:

A1: Our goal when looking at the Telecommunications Churn Data set is to predict customer churn. We will focus on our goal by creating a k nearest neighbors’ classification model when targeting the dependent variable Churn. After creating a valid k nearest neighbors’ model, we can then use our input parameters to predict whether a customer will churn. After running our k nearest neighbors’ model, we can answer the question “what to invest in to minimize customer churn?”.

A2: The goal of our analysis is to build a k nearest neighbor classification model for predicting customer churn. Once we build our k nearest neighbor model, we can test to see if our model is valid. If our model is valid, we know what variables will help us in predicting churn.

# Part 2 Method Justification

B1: Our k nearest neighbor classification model works by classifying each data point. Our model sets up classification parameters and looks to see what data points fall into each classification. Any new data is then classified based on similarity to our KNN model. We build our model based on training data and review our model using testing data. We expect our KNN model to accurately predict Churn based on input parameters.

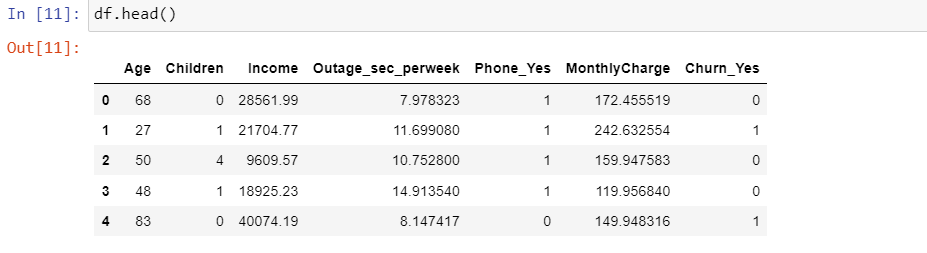
B2: KNN is a non-parametric based algorithm and does not require any assumptions on data distribution. Harrison (2019) explains how one of the assumptions for our K nearest neighbor’s classification method is to have a discrete output. Our KNN model has Churn (Yes or No) as a discrete output matching Harrison’s description.

B3: Some of the benefits of using Python are the libraries created for data analytics. In this project we used NumPy for indexing and arrays, Pandas for data formatting, Matplotlib for creating graphs, Seaborn for data visualizations, and SciKit for data modeling. With SciKit we scale our data, construct a training/test model, build a KNN classification model, create a classification report, and generate a confusion matrix. Zhidkov, R (2021) described Python as open source and having a large community for problem solving. Python is useful when analyzing large datasets due to its speed and processing. For this project, Python is useful because our data has 10,000 rows and 50 columns.

# Part 3 Data Preparation:

C1: One important pre-classification step for our KNN model is scaling our data. Data scaling can be achieved by normalizing or standardizing input and output variables. We use the Scikit Learn preprocessing library for data scaling.

C2: Our variables for our KNN model are listed below. We will use Age (continuous), Income (continuous), Outage Seconds Per Week (continuous), Phone Plan Yes (categorical), Monthly Charge (Continuous) as our X or predictor variables. We will use Churn (categorical) as our y or target variable.



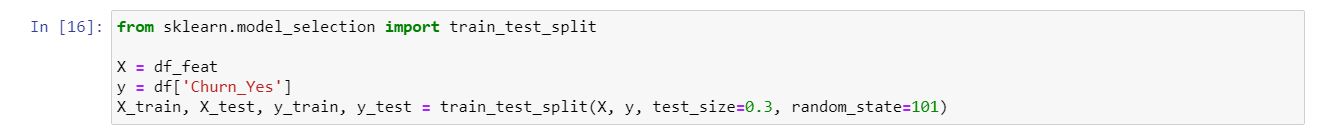
C3:

For my data preparation, I used the Pandas library to import the csv data. I started by converting all the binary columns into dummy variables. I then looked through the data and chose what columns I thought would be useful for predicting churn. I made sure these variables were continuous or binary and would be useful in predicting churn. I then dropped the columns I deemed not useful to our model. I proceeded to standardize our data using the preprocessing package from sklearn. We had to create a scaler model and fit the model to our cleaned data. Once the data was fit to the scaler, we transformed the data. I created two variables, one that scaled all the data under cleaned\_csv and another variable df\_feat which excluded the target variable Churn. I set our transformed data to the variable cleaned\_data in order to print the dataset.



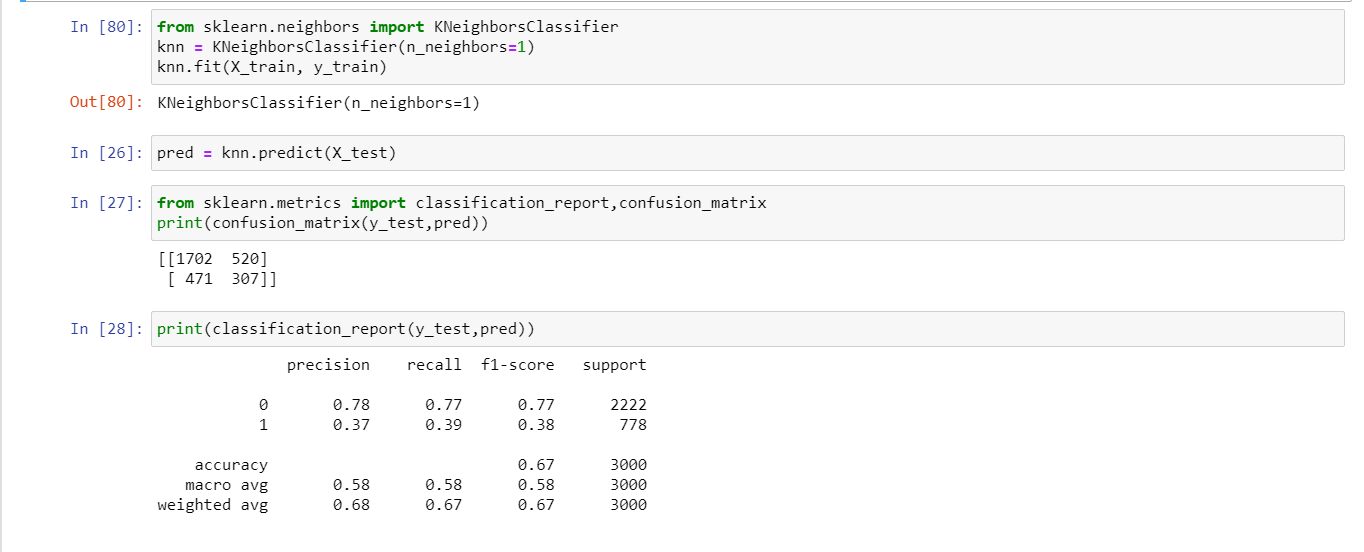
# Part 4 Analysis:

D1: We can use the train\_test\_split function from Sci Kit to setup the testing and training data. We set our scaled data as the independent variables and Churn as the dependent variable.



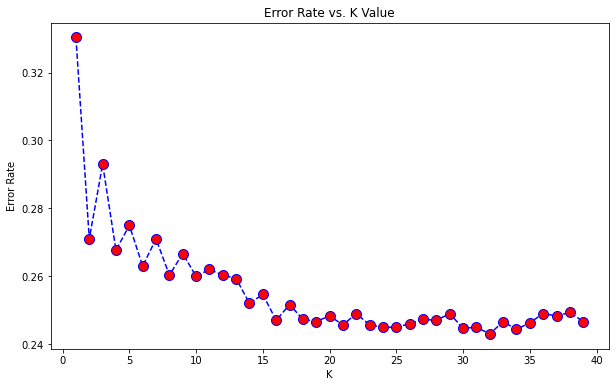
D2/D3:

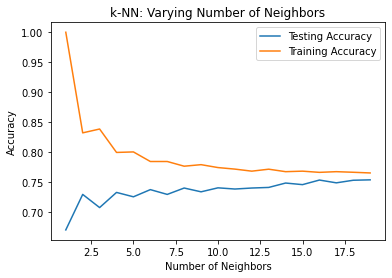
In order to setup our KNN model, we import the KneighborsClassifier from sklearn. We then create a KNN classifier model and set the model to the variable knn. We frame our model so it’s looking for the closest neighbor of 1. We then fit our KNN model with data X\_train and y\_train. Once we have our model fit, we can run our model with the test data. Finally, I then created a confusion matrix and classification report to analyze our model.



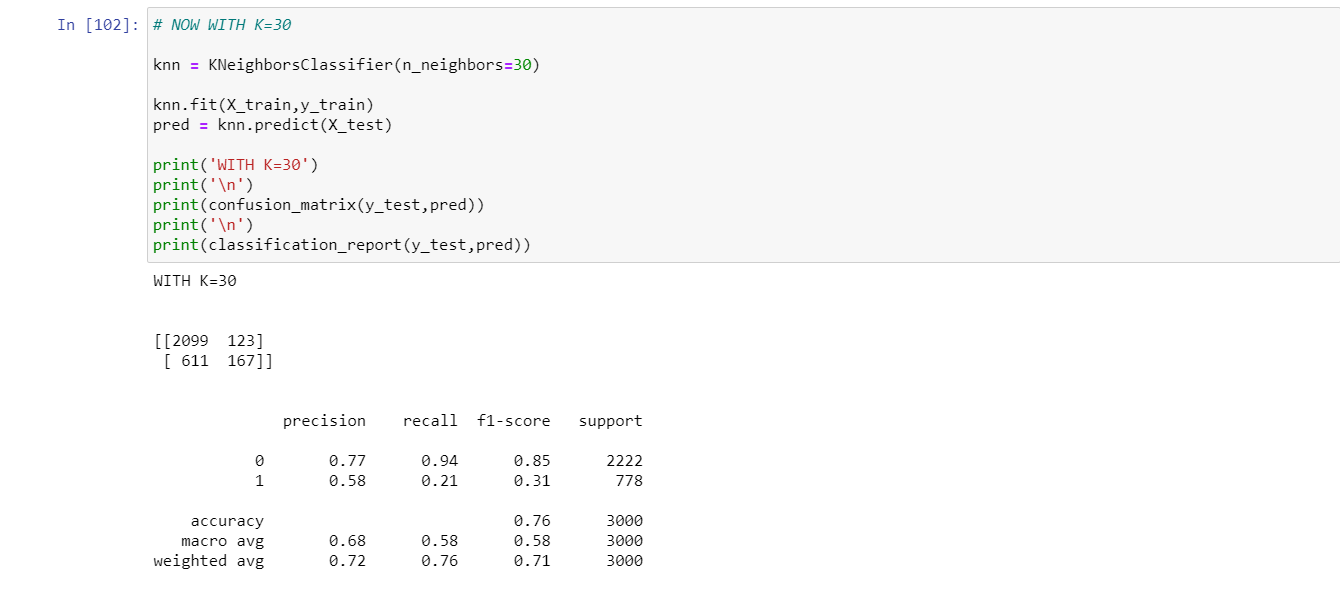
# Part 5 Data Summary and Implications:

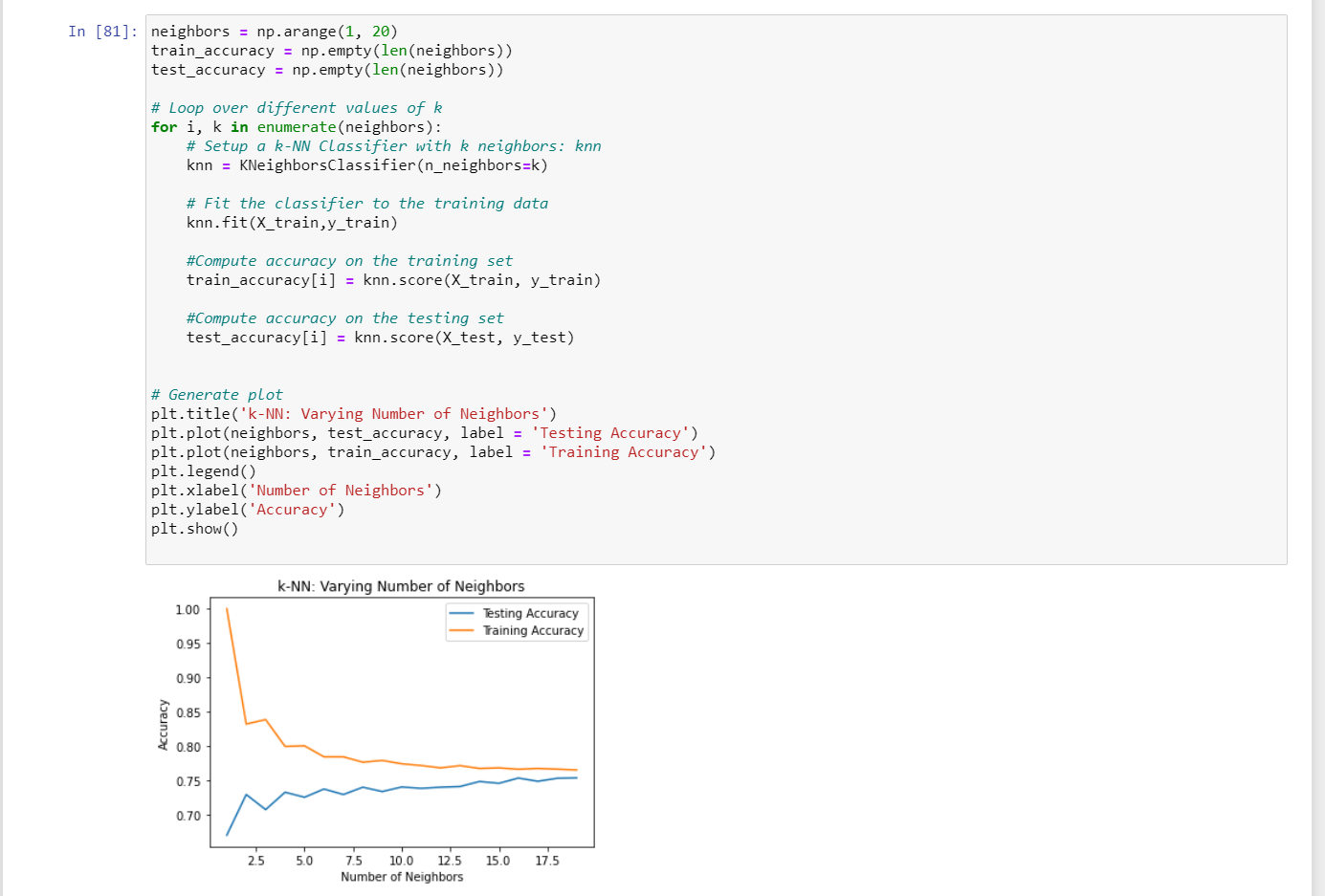
E1: Our AUC allows us to visually see the accuracy for both our training and testing data. As we see from our AUC curve, as the number of neighbors goes up the error rate goes down. The outcome shows that our model isn’t very strong when the number of neighbors is set to 1. The accuracy of our model goes up when we add additional neighbors. Our classification report and confusion matrix confirm our AUC curve. For our classification report we get a macro average of .58 for precision, recall, and f score. Overall, our f1-score accuracy is only .67.





E2: Based on our results, I would suggest increasing the number of nearest neighbors. After running the classification with a higher number of neighbors (30), we see an increase in precision from .58 to.68. Overall, our model isn’t very strong. We want to avoid overfitting the data, so setting the k nearest neighbors to n =15 would be the ideal situation.





E3: One of the limitations of our data is the high dimensionality. We have many variables that can be used for predicting Churn. I don’t think KNN would be the best due to the sheer amount of data and versatility. I think a random forest tree, or a more advanced classifier would be better for this dataset.

E4: Overall, I would suggest as a course of action to continue developing the KNN model. We should focus to increase the classification strength for customer churn. I would do this by adding additional variables to our classification model and work to improve our model accuracy. I would continue to invest in our independent variables as they contribute to a certain degree in Churn with precision sitting around .76 when using 30 neighbors. In addition, I would also try testing our model with datasets from different time periods.

# References

Sehra, C. (2020, November 30). *Decision Trees Explained Easily*. Medium. https://chiragsehra42.medium.com/decision-trees-explained-easily-28f23241248.

Zhidkov, R. (2021, January 10). *Why Python is Essential for Data Analysis*. RTInsights. https://www.rtinsights.com/why-python-is-essential-for-data-analysis/#:~:text=The%20object%2Doriented%20programming%20language,streamline%20large%20complex%20data%20sets.&text=Being%20fast%2C%20Python%20jibes%20well,not%20limited%20to%20scientific%20computing.

**Resources for Python Libraries:**

https://matplotlib.org/

https://numpy.org/

<https://pandas.pydata.org/>

https://scikit-learn.org/stable/

https://seaborn.pydata.org/