Final Paper

I chose an ecommerce dataset from Kaggle with data about their customers where each observation corresponded to a customer purchases for their electronic products. It included attributes like cost of a product, discount, weight of the product, product importance, mode of shipment, where a particular product was shipped from, customer rating, whether it reached on time or not and associated customer care calls. Also, it included number of prior purchases the customer did.

The reason I decided to work on this dataset was because of its high usability rating given the limited time we had to make progress on our assignments. I wanted to select a fairly complete dataset that is large enough and had a good number and mix of quantitative and qualitative variables to be able to perform different data preparation techniques depending on the task. Some other datasets I explored required more data preparation steps and understanding of the context.

However, looking at the all variables and the sequence of the events in the process of buying, delivering and receiving feedback, I assumed that the outcome and the variable to be predicted would be the customer rating received at the end of the process and that would be what a company would try to optimize taking into consideration the different data points during the process. Also, I assumed that if I were to predict whether a product was delivered on time or not, I would consider other variables such as the shipping company, whether there was a shipping confirmation or not, type of shipping etc.

That is why for the first assignment I tried to predict the customer rating, while it was stated in the description of the dataset that it was designed to be used to predict whether a product reached on time or not.

In the first assignment, first I did an initial exploratory data analysis by looking at the different variables, statistical summaries of the variables and counts of the categorical variables. Then I split the data into training and test datasets and repeated the same exploratory analysis for the training and test datasets. One thing I missed as per the requirements of the Project 1, and I got to correct in Project 2 was to use `stratify=y` parameter in `train_test_split` command while splitting the data which ensures that the data values of y represented equally in proportion to size of the group.

In terms of data preparation, since the dataset was fairly clean, standardized and complete. Thus, I had to make some adjustments to my data to simulate having missing datapoints. I did that on the dataset outside by randomly selecting (assigned random numbers in Excel and removed the values that corresponded to a certain digit of number) deleting datapoints in weights_in_gms. The dataset initially had 10999 observations. I chose to remove the rows with missing data since I had only around 818 of them.

In Project 2, as part of data preparation I converted the ordinal variable Product_importance to numeric then along with other numeric variables, I used scaling. For the categorical variables I used OneHotEncoder.

I visualized the variables in my data using histogram and scatter matrix. The histograms showed the distributions of the numeric variables. The scatter matrix showed the bivariate relationships across the numeric variables. There were no significant relationships discovered in the numeric variables used in visualizing these. The scatter matrix could be helpful if the predicted variable was also numeric also if there was any visible relationship between any of the numeric variables. If I started with the right outcome variable, another visualization I could have tried would be to create a box and whisker plot of the numeric variables against reached_on_time varible, which could inform how different numeric variables could potentially inform some relationship between the outcome.

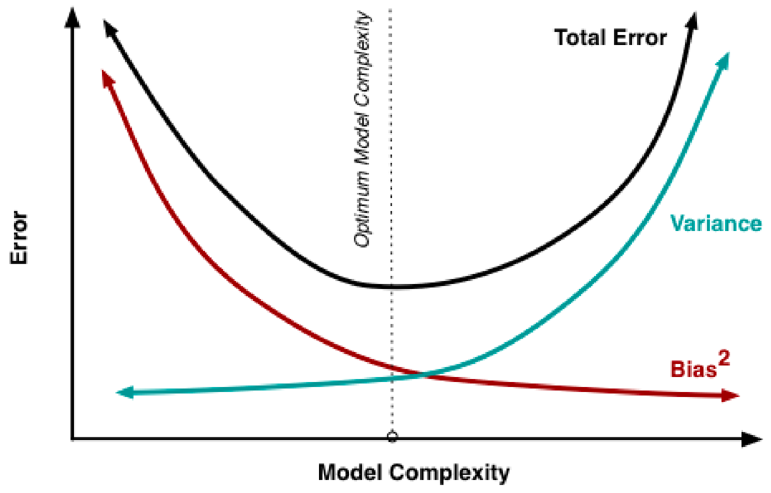
I used K-nearest Neighbor Classifier algorithm in predicting the categorical variable ReachedonTime_YN. First I started with 5 nearest neighbors, used cross validation and calculated accuracy, precision, recall and F1 scores. I got not a very strong score of 0.63 – 0.64 for these.

Then I tried Random Forest Classifier and I got an accuracy of 1.00 for the training dataset and 0.66 for the test dataset. With cross validation I got around 0.66 of accuracy. The precision, recall and F1 scores were only marginally improved with Random Forest Classifier from the previous analysis to be around 0.65 – 0.66.

Next I did a GridSearch to find the best parameters for both of the algorithms. For K-nearest Neighbor Classifier, I tried finding optimum leaf size from a range of 1 through 50 and number of neighbors from a range of 1 through 50. For the leaf size of 1, Euclidean distance and neighbors of 8, I got an accuracy score of 0.65 and precision, recall and F1 at around 0.64 – 0.65, with very minimal improvement than the first iteration. I also generated a validation curve to visualize this range for training and cross validation scores for different values of neighbors, it was apparent that the accuracy score would not improve beyond this range.

For the GridSearch for Random Forest Classifier I looked at n estimators and max depth for the ranges 100 increments up to 500 and 10s increments up to 50 respectively, and got the best n estimators as 100 max depth as 10, which gave a score of 0.67 and precision, recall, F1 scores at around 0.67 – 0.70. I selected this as my best performing algorithm.

Random Forest Classifiers can be expected to give better results in predicting the relatively sparse data compared to K-nearest neighbors. I tried using a range of parameters using Gridsearch and also visualized the validation curve for a range of values for n-neighbors for Knn model, but parameter-tuning would not further improve the model. But in both cases the models are underfitting and are not good predictor of the outcome variable for the test data, they are in high bias, low variance range as they are not complex enough as per the bias-variance trade-off curve below:



The relationship between bias-variance and model complexity.
Credit: *http://scott.fortmann-roe.com/docs/BiasVariance.html*

In Project 3, I used PCA to evaluate the relative strength of 18 features with 95% of the variance explained, which ended up considering 13 features. Since the relative explained ratio of the most significant variable (0.11) to those that were cut off with PCA (starting from 0.03) did not have that big of a difference, so once the features that were not considered it is expected that there could be a slight decrease in the model performance (0.67 vs 0.61 with PCA).

For unsupervised clustering algorithms, in general the scores were very low close to 0 and using PCA gave marginally better results. The silhouette score for k-means clustering increased from 0.18 to 0.19 with PCA. But since the datapoints are not really seperable, neither of the models are successful in finding meaningful clusters, thus the scores are low for all k-means clustering, agglomerative clustering and DBSCAN algorithms.

Overall, the models I used and evaluated for this dataset was not very strong in predicting the predictive variable in supervised learning and in determining meaningful clusters in unsupervised learning. If I were to do this assignment again, I would spend more time doing exploratory data analysis and feature engineering and try experimenting with different transformation techniques. I could also try the same process several other more complex models in both supervised and unsupervised learning methods.