Customer Segmentation using Multi-class Classification

Exploring automobile domain's customer segmentation data and building models to foster business trend.

Lavita Pereira
Department of Computer Science
Dalhousie University
Halifax, Canada
lv795165@dal.ca

Sagar Devesh
Department of Computer Science
Dalhousie University
Halifax, Canada
sg226953@dal.ca

Saurabh Jayeshbhai Das
Department of Computer Science
Dalhousie University
Halifax, Canada
sr850847@dal.ca

Sushumna Srinivasa Pradeep
Department of Computer Science
Dalhousie University
Halifax, Canada
ss508518@dal.ca

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Customer segmentation is the process of dividing customers into groups based on common characteristics so that the company can target promotions, sales, marketing, and thus increase profit [1]. In B2C marketing approach, the firm might segment the customers according to industry, number of employees in the company, previously purchased products, and even location. In case of B2C marketing, companies switch over to age, gender, marital status, location, lifestyle, and even income scale **[2a]**.

Companies put effort into segmenting customers so that the marketing team can better tailor their marketing efforts according to the audience's interests. These efforts then turn in to help in product development as well. Thus, customer helps a company to achieve the following:

- Reach specific group of customers with marketing messages that are relevant and thus resonate with them.
- Select the best communication medium for a particular segment, i.e., either email, social media, text messages, push notifications, radio advertising, or any other approach.
- Recognize ways to improve products, develop new products, and extend service opportunities.
- Estimate better ranges of the pricing tiers.
- Improve customer relationships and customer service.
- Filter most profitable tiers in the customer and focus on increasing revenue with this tier.
- Upsell and cross-sell related products and services to the customers.

To segment customers, the company needs to gather specific information about the customer and then need to analyze and identify patterns out if it. Most of this information can be gathered from the purchasing information of the customers. In the internet era, it is even easier by collecting data of each customer using each of their transactions keeping data privacy rules and regulations into consideration. However, for business not operating on e-commerce platforms, the firm needs to setup face-to-face interviews, telephonic interviews, surveys, and even setting up focus groups. With the customer segmentation done, it is really very important that the marketing and sales team then knows how to use this information derive quantifiable results and also achieve specific goals that can help the firm grow further.

The dataset our team selected comes in from an automobile firm. The firm already has five different products (namely P1, P2, P3, P4, and P5) in sales with good profits. They intent to enter new market. To shrink the time that slips in making profits, the firm already figured out that this new market is analogous in behavior with the existing market that the company has been operating in. With this information handy, the firm wants to reuse the customer segmentation that their sales team distilled out. There are 4 segments, and the firm intends to apply the same promotion and customer tactics using this previous knowledge on the 2627 new potential customers. As big data and machine learning engineers, our team's task is to use the raw data provided by the sales team and then predict the segment of new customers so as to help the manager increase business profit. **[Abishek Sudarshan, "Customer Segmentation," Kaggle.com, 2021. [Online]. Available:**

https://www.kaggle.com/datasets/abisheksudarshan/customer-segmentation. [Accessed: Dec. 07, 2022]].

Initially, when we narrowed downed to finding datasets on the problem "Customer Segmentation", we got a lot of datasets on Kaggle **["Kaggle: Your Home for Data Science," Kaggle.com, 2020. [Online]. Available: https://www.kaggle.com/. [Accessed: Dec. 07, 2022]]**. We filtered out the ones that had good problem domains attached to it so that we can build a project that can derive imperative insights rather than just obvious conclusions. We found interest in two datasets viz. "Customer Segmentation" **[Cited above]** by Abishek Sudarshan **["Abishek Sudarshan — Expert," Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/abisheksudarshan. [Accessed: Dec. 07, 2022]] and "E-commerce Data" [Carrie, "E-Commerce Data," Kaggle.com, 2017. [Online]. Available: https://www.kaggle.com/datasets/carrie1/ecommerce-data. [Accessed: Dec. 07, 2022]]** by Carrie **["Carrie — Novice," Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/carrie1. [Accessed: Dec. 07, 2022]]**. The later one had around 550,000 rows and thus seemed to be a perfect fit for a big data domain's data but we shunned it because it had too less features and the problem domain was unsupervised clustering which meant that we would have to label the data by our own before performing the classification. The former one had around 8000 tuples in training and 2000 tuples in the testing dataset but showed more promise in the number of features and was labelled as well. Thus we decided to go with the "Customer Segmentation" dataset. The dataset has volume, variety, and value.

- There's volume since the dataset has around 8000 tuples in the training data.
- There's variety in the dataset as we have integer, float, and categorical values across the feature set.
- The dataset definetly has value since we are deriving classes of potential customers with out machine learning models which will help the automobile firm gain profits in their business.

II. Related Work

For the customer segmentation dataset that we chose, we found multiple approaches and solutions on Kaggle. In one of the approaches, George Vosorov **[x]** used models like Logistic regression, K nearest neighbors, Support Vector Machine and Xgboost. **[y] [x -"George Vosorov — Contributor," Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/georgevosorov. [Accessed: Dec. 07, 2022]] [y -georgevosorov, "customer segmetation (Xgboost - 0.5278)," Kaggle.com, Aug. 31, 2022. [Online]. Available: https://www.kaggle.com/code/georgevosorov/customer-segmetation-xgboost-0-5278. [Accessed: Dec. 07, 2022]]** Out of the four models, George obtained the best accuracy for the Xgboost model, which was 52.78%. Preprocessing methods like one hot encoding for categorical features and normalization for standard features were used. George made a few observations on the train data before preprocessing, using pie charts and count plots. According to his insights, unmarried customers are usually in segment D, while married customers are in segments A, B or C. Graduated customers are mostly in segments A, B or C, while undergraduate customers are in segment D. Furthermore, even though there is not a big difference in the scale of numerical features as per George, standard scaling is still important to avoid any problems with the model's learning process.

The approach used by George is good, as far as preprocessing of the dataset is concerned. However, when it comes to implementing the models, there was no validation set considered before proceeding with the hyperparameter tuning of the models. Implementing a model on the validation set is an important part of the model evaluation procedure as it is tested for the first time on unseen data, and then hyperparameter tuning is conducted based on the results obtained after testing the model on the validation set. This helps in improving the model performance or accuracy when the model is tested for the first time on test set.

In another approach, Abhinav Jhanwar **[a]** performed exploratory data analysis of the dataset, post which he used preprocessing methods like label encoding for categorical features **[b]**. Later, he implemented three models, namely Logistic Regression, Xgboost and Deep Learning, and then compared the performances of the three. **[a]"AbhinavJhanwar — Expert," Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/abhinavjhanwar. [Accessed: Dec. 07, 2022] [b] abhinavjhanwar, "Customer-Segmentation Accuracy:53," Kaggle.com, May 26, 2021. [Online]. Available: https://www.kaggle.com/code/abhinavjhanwar/customer-segmentation-accuracy-53. [Accessed: Dec. 07, 2022]**. Logistic regression provided an accuracy of 52%, the deep learning model got an accuracy of 52.79%, whereas the XGboost model again topped with an accuracy of 53%.

Abhinav made a few observations or assumptions based on the exploratory data analysis that he conducted. The assumptions were fairly similar to the ones provided by George **[y]** in his notebook. Some additional observations made by Abhinav were that the customers in the healthcare and marketing fields are mostly in segment D, whereas artists and engineers are usually in segments A, B or C. Another observation made was that 'Low' spenders are usually in segments A or D, whereas 'high' and 'average' spenders are in segments B or C. These observations/assumptions were fairly correct as it was portrayed by the pie charts plotted in the notebook.

One of the highlights of Abhinav's notebook was the visualizations. A pie chart was plotted for each of the categorical features which portrayed the effect of each label in that feature on the target variable. Fig. 1. is an example pie chart from the notebook depicting the effect of the gender feature on customer segmentation.

A weakness that could be brought forth in Abhinav's solution is again no use of the validation set. There is no validation set created for evaluating the model. Furthermore, hyperparameter tuning is not performed for any of the models,

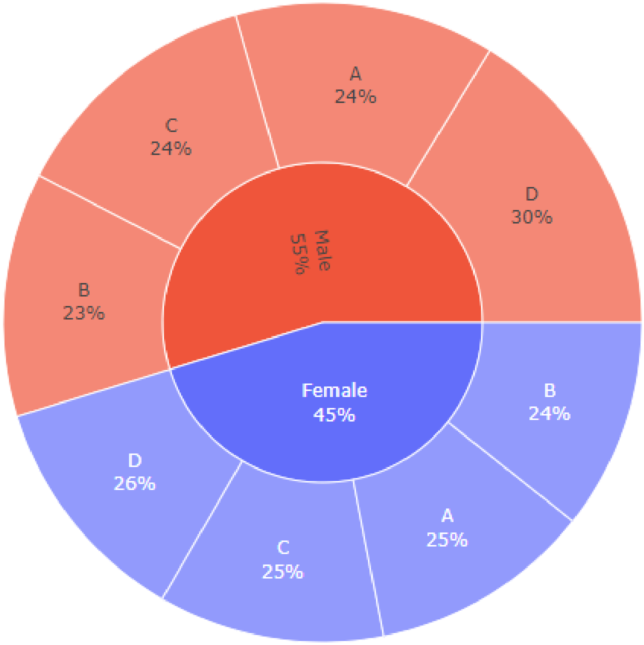


Fig. 1. Effect of Gender on Customer Segmentation.

which could have perhaps given us better accuracy.

Luiz Bueno [c] has used an alternative approach in one of the preprocessing steps in his notebook [d]. **[c]"Luiz Bueno — Master," Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/juniorbueno. [Accessed: Dec. 07, 2022] [d] juniorbueno, "Customer/K-Means/Hierarchical Grouping/DBSCAN," Kaggle.com, Nov. 22, 2021. [Online]. Available: https://www.kaggle.com/code/juniorbueno/customer-k-means-hierarchical-grouping-dbscan/notebook. [Accessed: Dec. 07, 2022]**. He treated the missing values for numerical features differently by replacing them with the mean of all the values for that feature. Later, he used the Standard Scaler library to normalize the numerical features. Luiz has used three clustering algorithms to group the customers into the respective five segmentations. The clustering algorithms used were K-Means, Hierarchical clustering and DBSCAN. As per the notebook, K-means was not able to do an efficient separation of the data. Agglomerative hierarchical clustering performed better than the K-means algorithm, however, the results were not satisfactory enough. DBSCAN, on the other hand, did significantly better in the grouping. It presented better and faster results as compared to K-means and agglomerative clustering algorithms. However, the notebook ended with an input stating that the DBSCAN algorithm would perform even better for more complex problems and a much larger database. Even though the models performed fairly well, they could have performed even better had the missing values not been replaced. During the pre-processing step, Luiz had replaced the all the missing values in the numerical feature with the mean of their values. Imputation of the missing values may not only result in decreased model accuracy, but may also lead to severely biased estimates.

III. Methodology

We chose to solve our problem using various machine-learning classification algorithms. We initially went through the dataset and the machine learning algorithms applied to the final dataset we used. And researched various models which can be applied.

We first used Bernoulli Naive Bayes Classification algorithm. We use Bernoulli naive Bayes as it is the best classifier for smaller datasets and gives us more precise values as compared to other models it is fast add helps us in making real-time-predictions **[1]Namita Mutha, "Bernoulli Naive Bayes," OpenGenus IQ: Computing Expertise & Legacy, May 30, 2020. [Online]. Available: https://iq.opengenus.org/bernoulli-naive-bayes/. [Accessed: Dec. 07, 2022]**

But we wanted to try our dataset on different models and predict the best classifier model for our dataset. Thus, we used another variant of Naïve Bayes, which is Gaussian Naive Bayes. This model is better than Bernoulli's naive Bayes thus it gave us a higher accuracy rate. It is better than Bernoulli naive Bayes because it is gaussian distribution or normal distribution which computes the probabilities of likelihood of data **[2] A. Saini, "Naive Bayes Algorithm: A Complete guide for Data Science Enthusiasts," Analytics Vidhya, Sep. 16, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/09/naive-bayes-algorithm-a-complete-guide-for-data-science-enthusiasts/. [Accessed: Dec. 07, 2022]**

Decision tree is one the easiest models to understand and interpret, and it can handle both numerical and categorical data. The goal in the decision tree is to plot a tree that predicts the final value by learning rules and by moving downwards from the root node, we can predict the final classification result **[3]. "1.10. Decision Trees," scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html. [Accessed: Dec. 07, 2022]**

We wanted to construct many decision trees to get a better classification model. So, we used Random Forest, which is the combination of many decision trees. Random Forest produces better results for huge datasets. Random Forest algorithm produces higher accuracy than the outcomes of decision tree. It also handles the overfitting of decision tree **[4]. "Introduction to Random Forest in Machine Learning," Engineering Education (EngEd) Program — Section, 2020. [Online]. Available: https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/. [Accessed: Dec. 07, 2022]**

XGBoost is one of the most efficient algorithms which has higher accuracy, efficiency, and has higher feasibility. It has the capacity to do parallel computation on a single machine. It is the regularized version of Gradient Boosting Machine. It's easy to use **[5]. N. Kumar, "Advantages of XGBoost Algorithm in Machine**

Learning," Blogspot.com, 2019. [Online]. Available: http://theprofessionalspoint.blogspot.com/2019/03/advantages-of-xgboost-algorithm-in.html. [Accessed: Dec. 07, 2022]

The next and last approach we have used is Multilayer Perceptron. It is a supervised learning algorithm. It is a feed-forward artificial neural network. It uses backpropagation for training. Since it is a deep-learning algorithm, it has better learning methods thus producing higher accuracy. Multilayer Perceptron has 3 different layers, input layer, hidden layer, and output layer **[7 put hyphen after Multilayer]. Wikipedia Contributors, "Multilayer perceptron," Wikipedia, Sep. 08, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Multilayerperceptron. [Accessed: Dec. 07, 2022]**

A. Data Flow Diagrams

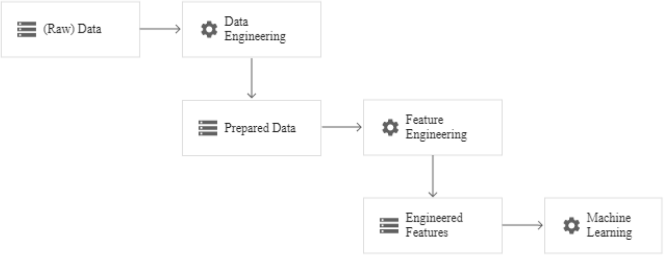


Fig. 2. Data-flow diagram for preprocessing [X] "Data preprocessing for ML: options and recommendations — Cloud Architecture Center — Google Cloud," Google Cloud, 2022. [Online]. Available: https://cloud.google.com/architecture/data-preprocessing-for-ml-with-tf-transform-pt1. [Accessed: Dec. 07, 2022]

Data preparation is the first step in order to apply any machine learning algorithm. In this, we do data engineering like the preparation of data, like dropping null values, duplicate values, etc to prepare the data.

After the data is prepared, the next step is the feature selection, in the feature selection, we select the most important features and drop the least important ones. The features which influence the final result are selected.

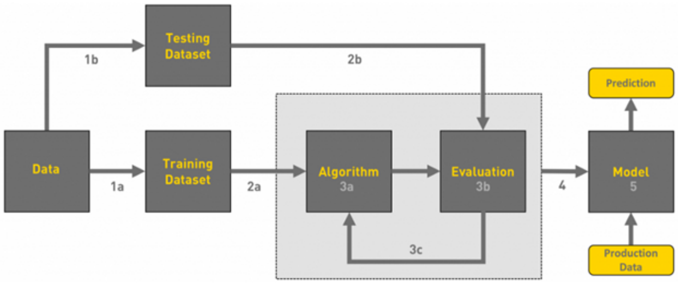


Fig. 3. Data-flow diagram for the ML model [] A. Pant, "Workflow of a Machine Learning project - Towards Data Science," Medium, Jan. 11, 2019. [Online]. Available: https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94. [Accessed: Dec. 07, 2022].

After data preparation and feature engineering, we need to split the data into training and testing datasets. We can also divide our training dataset into training and validation datasets. We use training datasets to train our model and apply machine-learning algorithms. We fit our training data into the algorithm. Next, we evaluate our model with the split dataset. Thus, our model is ready. And we send our sample data and predict the result based on the trained model.

IV. Experiments

A. Data Collection

The dataset comes from an automobile company that wants to expand their business into new markets with their existing products. The company conducted a thorough market analysis research and concluded that the new market would behave the same as the existing market. Hence, the company decides to follow the strategy that worked for them in their existing business. Sales team analyzed existing market data and classified consumers into 4 different categories (A, B, C, D). The team then tailored outreach and communication for each segment of the market, which resulted in huge profits for the company. As a result, the company wants to use the same strategy in its new market as well. The new market consists of 2627 potential customers and the manager needs to predict the right category for new consumers [] **[Abishek Sudarshan, "Customer Segmentation," Kaggle.com, 2021. [Online]. Available: https://www.kaggle.com/datasets/abisheksudarshan/customer-segmentation. [Accessed: Dec. 07, 2022]**].

The dataset we used outlines the spending habits of around 10000 customers. From Kaggle [] **[Abishek Sudarshan, "Customer Segmentation," Kaggle.com, 2021. [Online]. Available: https://www.kaggle.com/datasets/abisheksudarshan/customer-segmentation. [Accessed: Dec. 07, 2022]**]we got both train and test dataset. Train dataset consists of 8068 rows and test dataset consists of 2627 rows. Each row in the datasets identifies a customer with 10 significant features pertaining to the customer. The train dataset also contains an additional column called 'Segmentation', which is the target variable column. The target variable takes values A, B, C, or D, indicating the segment a customer belongs to. Following is the meaning of each attribute in the dataset.

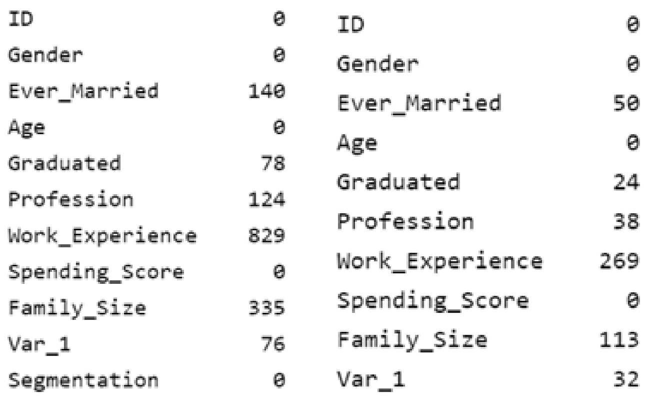
1) **ID** - Unique ID
2) **Gender** - Gender of the customer
3) **Ever_Married** - Marital status of the customer
4) **Age** - Age of the customer
5) **Graduated** - Identifies if a customer is a graduate
6) **Profession** - Profession of the customer
7) **Work_Experience** - Work experience in years
8) **Spending_Score** - Spending score of the customer
9) **Family_Size** - Number of family members for the customer (including the customer)
10) **Var_1**- Anonymized category for the customer
11) **Segmentation** - Customer Segment of the customer (Target variable)

Apart from this dataset we also explored another customer segmentation dataset **[x] [Carrie, “E-Commerce Data,” Kaggle.com, 2017. [Online]. Available: https://www.kaggle.com/datasets/carrie1/ecommerce-data. [Accessed: Dec. 07, 2022]]**.

B. Data Preprocessing

Data pre-processing is an important step in machine learning. Real world data is often missing values, inconsistent, or maybe devoid of certain behaviors or trends, and is also likely to be error prone. Data preprocessing involves cleaning, formatting and organizing raw data to extract useful information. This makes the data suitable for machine learning model and increases the model’s accuracy and efficiency **[] [“Data Preprocessing in Machine Learning: 7 Easy Steps To Follow — upGrad blog,” upGrad blog, Jul. 15, 2021. [Online]. Available: https://www.upgrad.com/blog/data-preprocessing-in-machine-learning/. [Accessed: Dec. 07, 2022]]**. As part of this step, we have checked for null values, duplicates, outliers, and performed normalization, and encoding in the train and test dataset.

1) Data Cleaning: Data cleaning is the process of preparing data for analysis by identifying and correcting errors. This step includes removing duplicate values, filtering unwanted outliers, fixing errors in the data, and imputing or dropping missing data. Real-world data often has a lot of missing values. It is important to handle missing values during data preprocessing for the model to work accurately and without any bias. We performed an analysis of the data and found some missing values. Among the missing values, most of them came from categorical features - Ever_Married, Graduated, and Profession. We decided to drop the missing values because the missing tuples accounted for around 110th of the training dataset. Imputing these values with synthetic values would have further decreased the accuracy of the model. After performing this step, we did not find any duplicate tuples in the datasets. Later, we also checked for outliers and there were no outliers in any of the features.



```
ID                   0
Gender               0
Ever_Married       140
Age                  0
Graduated           78
Profession         124
Work_Experience    829
Spending_Score       0
Family_Size        335
Var_1               76
Segmentation         0
```

```
ID                   0
Gender               0
Ever_Married        50
Age                  0
Graduated           24
Profession          38
Work_Experience    269
Spending_Score       0
Family_Size        113
Var_1               32
```

Fig. 4. Count of NULL values in training and test dataset.

2) Encoding: Features such as Gender, Ever_Married, Graduated, Profession, Spending_Score, Var_1, and Segmentation are categorical variables, and they have a limited number of possible values. Some models such as decision trees work well with categorical data. However, most machine learning models require variables to be numeric as they cannot operate on label data directly **[] [“ML — One Hot Encoding to treat Categorical data parameters - GeeksforGeeks,” GeeksforGeeks, Jun. 12, 2019. [Online]. Available: https://www.geeksforgeeks.org/ml-one-hot-encoding-of-datasets-in-python/. [Accessed: Dec. 07, 2022]**]. Here we are performing label-encoding by using SciKit learn library to convert categorical data into numerical data.

3) Normalization: Data preparation in machine learning often includes normalization where values are scaled between 0 and 1. Normalization is useful when we do not know the feature distribution. This technique helps to improve the performance and accuracy of the model. We are using normalization as the columns Age, Work_Experience, and Family_Size features in the dataset have different ranges. Here we are using Min-Max scaling. The formula used for normalization is **[] [A. Bhandari, “Feature Scaling — Standardization Vs Normalization,” Analytics Vidhya, Apr. 03, 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/. [Accessed: Dec. 07, 2022]**]:

$$X' = \frac{X-Xmin}{Xmax-Xmin}$$

Xmax - maximum value of the feature
Xmin - minimum values of the feature

One drawback of min-max scaling is that does not work well if there are outliers in the data. Since our dataset does not have any outliers, we have used min-max scaling.

C. Feature Selection

Feature selection is the process of reducing the number of features to be used for training a machine learning model. Having a lot of features increases the chances of overfitting since the model becomes increasingly dependent on the data it was trained on and overfits, resulting in poor performance on real data **[] [A. Kolte, B. Mahitha and N. V. G. Raju, ”Stratification of Parkinson Disease using python scikit-learn ML library,” 2019 International Conference on Emerging Trends in Science and Engineering (ICESE), 2019, pp. 1-4, doi: 10.1109/ICESE46178.2019.9194627.]**. We have used Pearson Correlation and Select K-Best algorithms for feature selection.

1) Pearson Correlation: Pearson’s correlation coefficient measures the statistical relationship between two continuous variables. It is a method that takes only a subset of relevant features from the dataset. Pearson correlation is a number between -1 and 1 that indicates the relationship between two variables. If the two variables are strongly correlated the value is closer to +1. If the two variables are negatively correlated the value is closer to -1. A value closer to 0 implies weak correlation and exact 0 implies there is no relationship between the variables **[]. [Sangita Yemulwar, “Feature Selection Techniques - Analytics Vidhya - Medium,” Medium, Sep. 27, 2019. [Online]. Available: https://medium.com/analytics-vidhya/feature-selection-techniques-2614b3b7efcd. [Accessed: Dec. 07, 2022]]**.

Furthermore, if two or more independent features are highly correlated, they can be considered as duplicates and may be removed.

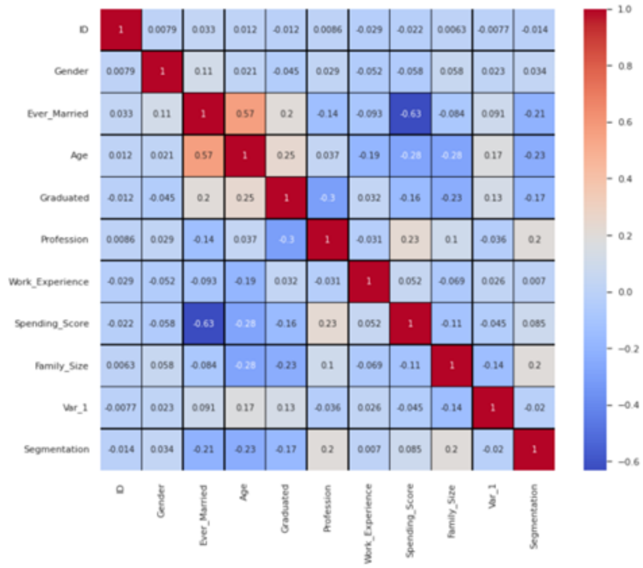


Fig. 5. Heatmap of the segmentation training dataset.

The above figure illustrates the heatmap of the segmentation training dataset. Here our target variable is Segmentation (dependent variable) and from the above figure we find out strong and weak correlation with independent variables. We can see that Profession and Family_Size are highly correlated with the target variable. Apart from this, Gender, Work_Experience, and Spending_Score are also positively correlated to the target variable. Var_1, Graduated, Ever_Married, and ID are negatively correlated with the target variable.

2) Select K-best: Univariate Selection is a feature selection technique that uses statistical test to select those features that have the strongest relationship with the target variable. The Scikit-Learn library provides the SelectKBest method that can be used with a different statistical test metrics to select a specific number of features in a dataset. Sklearn library provides predefined score functions that can be used to define a metric. Since ours is a classification model, the metric that we used is f_classif which computes the ANOVA F-value between label and features for classification tasks **[] [“Feature Selection - Hands-on Machine Learning with Scikit-Learn,” Educative: Interactive Courses for Software Developers, 2022. [Online]. Available: https://www.educative.io/courses/hands-on-machine-learning-with-scikit-learn/myxq1v4mXxA. [Accessed: Dec. 07, 2022]]**. We chose the value of K as 10, calculated the metrics between the target and each feature, sorted them, and then selected the best features. Using SelectKBest algorithm we got the lowest score for the feature ‘ID’. Hence, we concluded that it is the least relevant feature for model prediction. The original dataset consists of 10 features among which ID is insignificant in the training process. Hence, it is dropped from both the training and test datasets.

D. Splitting Data

The dataset that we have is already divided into train and test datasets. Training data is almost balanced with 1616 records for A, 1572 records for B, 1720 records for C, and 1757 records for D segments. The training dataset is further split into train and validation sets. 90% of the data is given as a training set and the remaining 10% is given as a testing set to the model. The test, train and validation data is stored in the form of two variables X and Y. We used 90:10 as it gave us better results during the model training phase.

E. Models

We are performing supervised classification of customer data into different segments. We have considered the supervised classification for the segmentation of customers using Bernoulli Naive Bayes, Gaussian Naive Bayes, Decision Tree, Random Forest, XGBoost Classifier, and Multilayer Perceptron.

1) Bernoulli Naive Bayes: We first used Bernoulli Naive Bayes classification algorithm. It is based on Bayes theorem of probability which gives the likelihood of the occurrence of the event. We used this algorithm as it is the best classifier for smaller datasets and gives us more precise values as compared to other models. **[][From Sushumna — Namita Mutha, “Bernoulli Naive Bayes,” OpenGenus IQ: Computing Expertise & Legacy, May 30, 2020. [Online]. Available: https://iq.opengenus.org/bernoulli-naive-bayes/. [Accessed: Dec. 07, 2022]**. We used BernoulliNB class from Sklearn to implement Bernoulli Naive Bayes algorithm. The accuracy achieved using the Bernoulli Naive Bayes classifier was 43.78% after applying the normalization and encoding techniques to the training data.

Hyperparameter Tuning Once the model is built, the next step is hyperparameter tuning. This step is important as it significantly affects the accuracy of the model. Different components of the machine learning model can be treated as hyperparameters and optimized. We have performed hyperparameter tuning using the Repeated Stratified K-Fold cross-validator method of the Sklearn library **[][“sklearn,” scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/l. [Accessed: Dec. 07, 2022]**]. Stratified K-Fold is run n times, producing different splits in each repetition. Here we have chosen the number of folds (n_splits) to be 10 and the number of times cross-validator needs to be repeated (n_repeats) to be 30. Apart from this, we also used the Grid

Search CV function from sklearn to get the best hyperparameters **[][“sklearn,” scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/l. [Accessed: Dec. 07, 2022]**].

After applying hyperparameter tuning to the dataset we were able to find the following best parameters:

- alpha: 0.001
- fit_prior: False

We applied Bernoulli Naive Bayes with the above hyperparameters on the training dataset. Hyperparameter tuning using Grid Search CV we were able to get an accuracy of 49.13%.

Learning Curve We obtained the below learning curve for Bernoulli Naive Bayes algorithm for training and validation data. From the figure, we can see that the generalization error between the training curve and the validation curve decreases with the increase in training dataset size. From this, we can infer that the model’s performance might not increase by adding more examples to our model. This problem might be solved by adding more features or making the model more flexible to reduce assumptions **[] [D. Team, “Learning Curves Tutorial: What Are Learning Curves?,” Datacamp.com, Mar. 09, 2022. [Online]. Available: https://www.datacamp.com/tutorial/tutorial-learning-curves. [Accessed: Dec. 07, 2022]**].

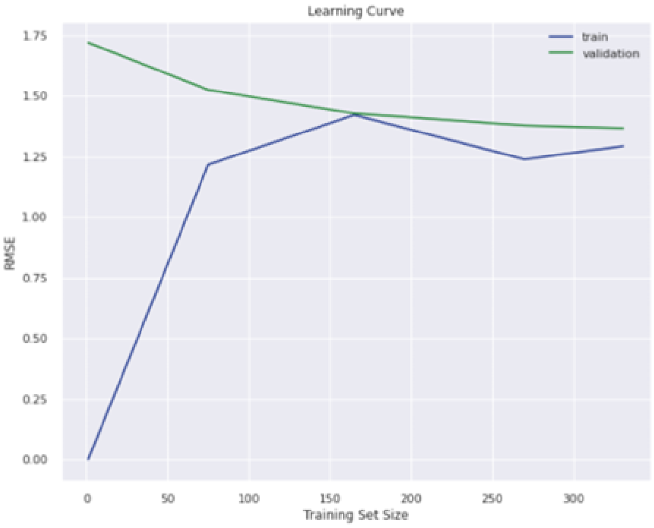


Fig. 6. Learning curve for Bernoulli Naive Bayes.

2) Gaussian Naive Bayes: We wanted to try our dataset on different models and predict the best classifier model for our dataset. Thus, we used another variant of Naïve Bayes, which is Gaussian Naive Bayes. This model is better than Bernoulli’s naive Bayes thus it gave us a higher accuracy rate. It is better than Bernoulli naive Bayes because it is gaussian distribution or normal distribution which computes the probabilities of likelihood of data **[] A. Saini, “Naive Bayes Algorithm: A Complete guide for Data Science Enthusiasts,” Analytics Vidhya, Sep. 16, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/09/naive-bayes-algorithm-a-complete-guide-for-data-science-enthusiasts/. [Accessed: Dec. 07, 2022]** We trained the Gaussian Naïve Bayes model using the train data and used the validation data to make predictions. The accuracy that we received for the validation set was 49.03%, which is comparatively more than the accuracy of the Bernoulli Naïve Bayes model before hyperparameter tuning.

Hyperparameter Tuning

For hyperparameter tuning of the Gaussian Naïve Bayes model, we used the GridSearchCV module. We have performed hyperparameter tuning using Repeated Stratified K-Fold cross validator method of Sklearn library. Grid search is a technique for finding the optimal parameter values from a given set of parameters in a grid, trying out all the possible combinations of parameters. When we implemented GridSearchCV on the Gaussian Naïve Bayes model, it returned a ’var_smoothing’ of: 0.03511191734215131, as the best parameter. Gaussian Naive Bayes does not really have a lot of hyperparamters to tune unlike other machine learning classifiers, except for var_smoothing. Therefore, we performed repeated K-fold cross validation to get the best version of the Gaussian Naive Bayes model.

Learning Curve We obtained the below learning curve for Gaussian Naive Bayes algorithm for training and validation data. The generalization error between the training curve and the validation curve is less, implying that the model has done well on the validation set. Both the curves are stable after 75 instances, thereby implying that increasing the number of instances would not further improve the model.

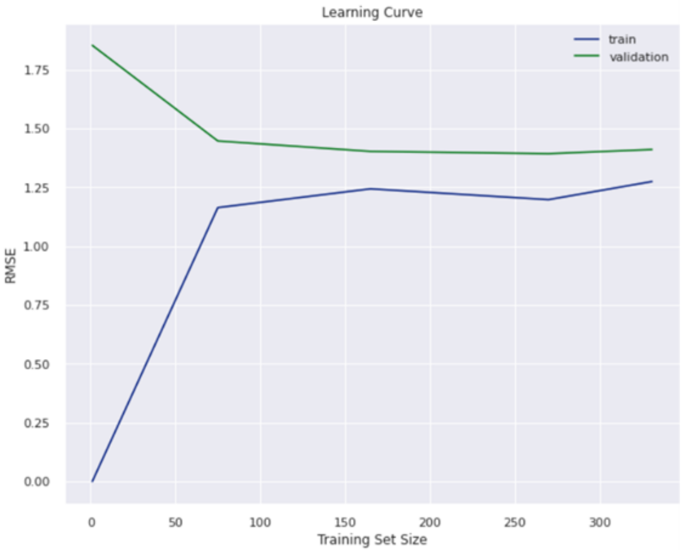


Fig. 7. LC.

3) Decision Tree: The third model that we used for classification is the decision tree model. It is a model which accounts for computational complexity, and views an algorithm as essentially being a decision tree. Decision trees are very useful in data analytics and machine learning because they break down complex problems or data into smaller manageable parts.

We fit the decision tree using the training data with 'entropy' criterion. The criterion parameter determines how the impurity of the split will be measured. It could either be 'gini' or 'entropy' however we have used 'entropy'. When we trained the model on the train data and made predictions with the model on the validation data, we got an accuracy of 43.73%.

Hyperparameter Tuning

For hyperparameter tuning of the decision tree model, we used the Randomized Search CV module. Randomized Search CV randomly passes a set of hyperparameters and returns the best combination of parameters that would eventually give us the best model performance. When we implemented Randomized Search CV on the decision tree model, it returned the following best set of parameters:

- max_depth: 3
- min_samples_leaf: 1
- min_samples_split: 7

When the model was rerun with these parameters, it gave an improved performance with an accuracy of **48.63%**.

Learning Curve

For the decision tree model, we obtained the below learning curve for training and validation data. As we can see, the generalization error between the training curve and the validation curve is less, implying that the model has performed well on the validation set. However, it has come at the cost of slightly declining performance on the training data, however, overall, it is a good model. Also, both curves are almost stable after 250 training instances, suggesting that adding more instances might not further improve the model. **[g] ' https://www.datacamp.com/tutorial/tutorial-learning-curves**

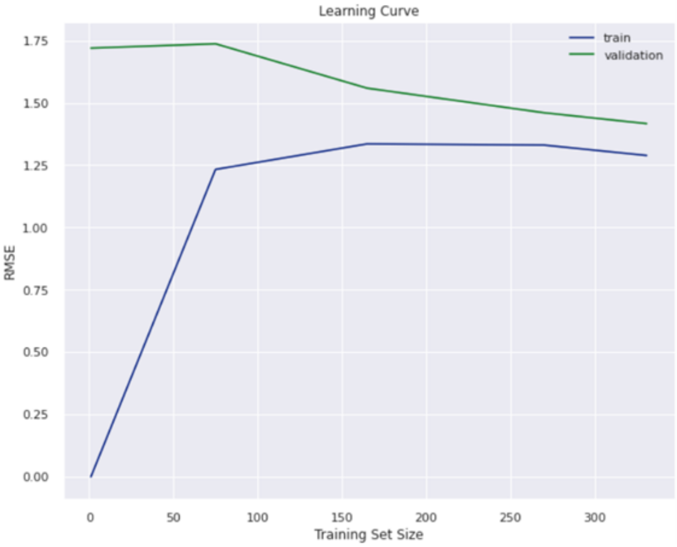


Fig. 8. LC.

4) Random Forest: A Random forest classifier is made up of a collection of decision trees whose results are aggregated into one result. It usually can perform both regression and classification tasks and can handle large datasets efficiently. We fit the random forest model using the train data and made predictions on the validation data. For this task, we got an accuracy of 50.82%.

Hyperparameter Tuning

For hyperparameter tuning of the random forest model, we used the Randomized Search CV module. Implementing Randomized Search CV on the random forest model returned the following best parameters:

- max_depth: None
- min_samples_leaf: 5
- min_samples_split: 5

When we implemented another RF model with the above parameters, we got an improved model with an accuracy of **52.98%**.

Learning Curve

For the Random Forest classifier model, we obtained the learning curve for training and validation data. As we can see, the generalization error between the training and validation curve is not that big. So, overall, the model has performed well. Both the curves become stable after around 75 training instances, thereby implying that increasing the number of instances further is not going to impact the model performance.

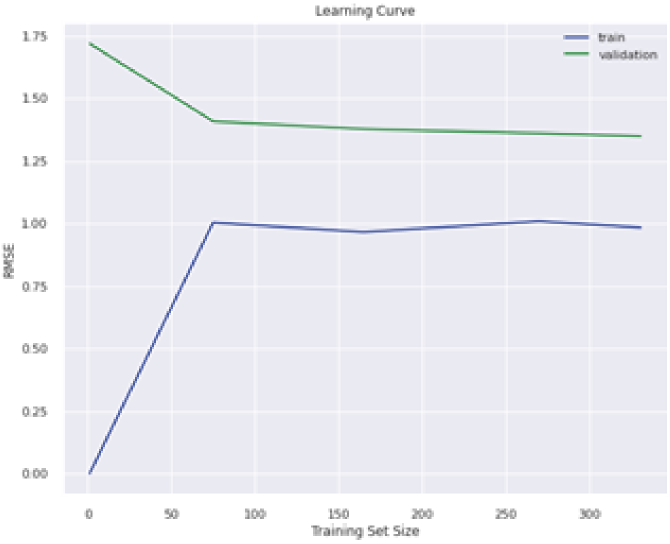


Fig. 9. Learning Curve for Random Forest.

5) XGBoost Classifier: XGBoost stands for Extreme Gradient Boosting. It is a scalable distributed gradient-boosted decision tree machine learning library. It is a leading machine learning library for classification tasks as it offers parallel tree boosting. In our case, we trained the model and then received an accuracy score of 56.37% on validation data, which is more than what others have received in the previous work. One change that we made as compared to previous work is that we split the train data into train and validation sets in 90:10 ratio. Since we already have a separate test dataset, we do not need to create it again. With 90:10 train-validation ratio, the XGBoost model returned an accuracy of 56.37%.

Hyperparameter Tuning

For hyperparameter tuning of the XGBoost model, we used the RamdomizedSearchCV module. Implementing RandomizedSearchCV on the XGBoost model returned the following parameters which turned out to be the best combination of parameters.

- subsample : 1
- min_child_weight : 1
- max_depth : 5
- learning_rate : 0.01
- gamma : 1
- eta : 10
- colsample_bytree : 0.6

Learning Curve

For the XGBoost model, we obtained the learning curve for training and validation data. We can see that the generalization error keeps decreasing as the number of instances increases. The train and validation curves appear to be converging even at the end of the graph, implying that increasing the number of instances may further improve the model.

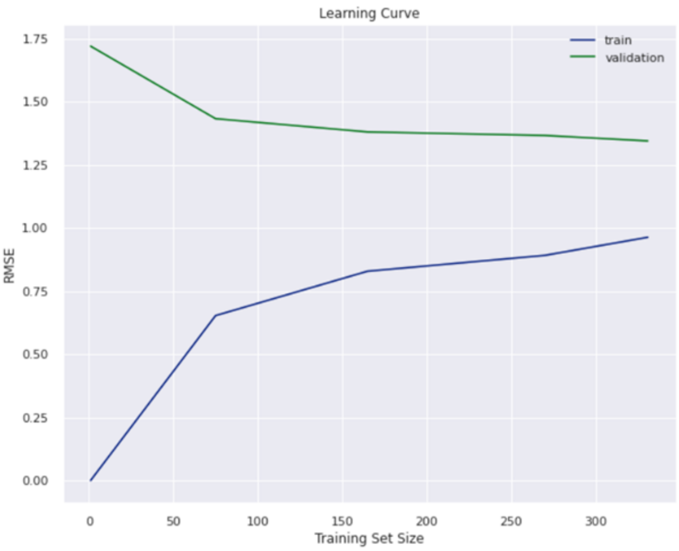


Fig. 10. Learning curve for XGBoost classifier.

6) Multilayer Perceptron: The final model that we used was Multilayer Perceptron (MLP). MLP is a supplement of feed forward artificial neural network **[] [Wikipedia Contributors, "Multilayer perceptron," Wikipedia, Sep. 08, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Multilayer_perceptron. [Accessed: Dec. 07, 2022]]**. It utilizes supervised learning technique called backpropagation for training **[] [Ş. Ozan and L. O. Iheme, "Artificial Neural Networks in Customer Segmentation," 2019 27th Signal Processing and Communications Applications Conference (SIU), 2019, pp. 1-4, doi: 10.1109/SIU.2019.8806558.]**. It has three layers – input, output, and hidden layer. The input layer We used MLPClassifier class from Sklearn to implement MLP algorithm. In the parameters we set the hidden_layer_sizes to be 100 to set 1 hidden layer with 100 hidden units. When we trained the model on the train data and made predictions with the validation data we got an accuracy of **54.72%**.

Hyperparameter Tuning

For hyperparameter tuning of the MLP model, we used the Randomized Search CV module of Sklearn to find the best hyperparameters. The parameters that we considered are hidden_layer_sizes, activation, solver, alpha, and learning_rate. On performing hyperparameter tuning we were able to find the following best parameters:

- solver: adam
- learning_rate: constant
- hidden_layer_sizes: (50, 50, 50)
- alpha: 0.05
- activation: tanh

When we implemented MLP with the above hyperparameters on the training dataset we were able to get an accuracy of **55.12%**.

Learning Curve

We obtained the below learning curve for MLP for the training and validation dataset. We can see that there is a reduction in error for the validation dataset. The generalization error is initially higher, but it gradually decreases with an increase in the training data size. Both the curves are stable for training sizes of 250 and above. From this we can conclude that adding more examples might not improve model performance **[] [D. Team, "Learning Curves Tutorial: What Are Learning Curves?," Datacamp.com, Mar. 09, 2022. [Online]. Available: https://www.datacamp.com/tutorial/tutorial-learning-curves. [Accessed: Dec. 07, 2022]]**.

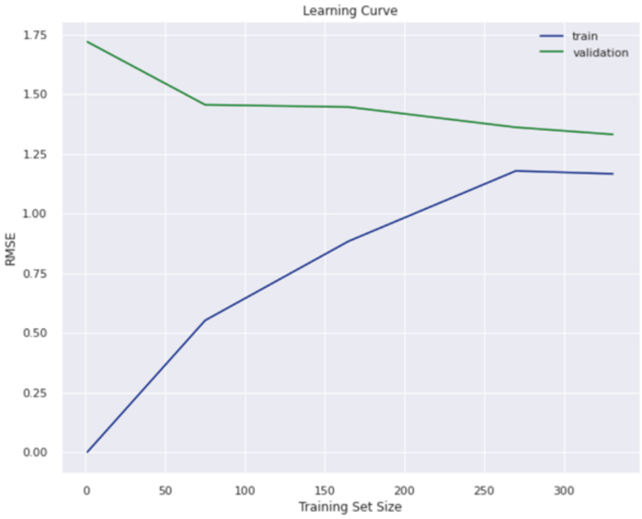


Fig. 11. Learning curve for Multilayer Perceptron.

V. Conclusion

In our attempt to do customer segmentation using different machine learning models, we achieved different results as discussed earlier. We got varied accuracies for the models we implemented. The XGBoost model gave us the best accuracy of 56.37%. For comparing all the model performances, we summarized the model results both numerically and using a box plot.

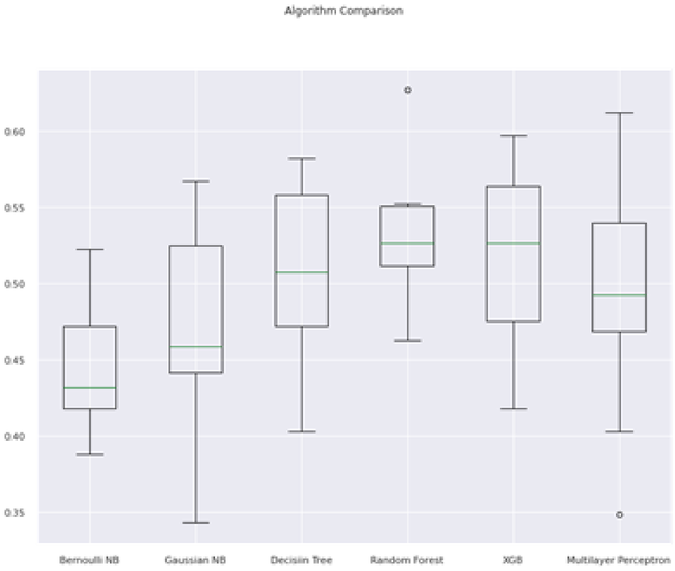


Fig. 12. Box plot for model comparison.

If we compare the mean values of all the box plots, we see that the XGBoost algorithm performs the best. Even numerically, XGBoost achieved the highest accuracy of 56.37%.

Statistical Significance Test

We performed a Statistical significance test to compare the models and to evaluate whether the difference in the mean performance of the models is a real number or not. We compared two models at a time to check if they are statistically significant with respect to each other. The null hypothesis is that the given two models are significantly similar. We chose the value of alpha as **0.05**, and then calculated the p-value. P-value is a number that describes how likely we will get a particular set of observations if the null hypothesis were true. In our case, if the p-value is greater than the value of alpha (0.05), it would mean we cannot reject the null hypothesis. This condition would imply that both models are quite similar. Typically, our objective is to get the two models to be significantly different. If the p-value is less than the value of alpha, we can reject the null hypothesis and conclude that the give two models are significantly different. While comparing the Bernoulli Naïve Bayes and Gaussian Naïve Bayes models, we found that both the models are not statistically significant, whereas when we compared Bernoulli Naïve Bayes and XGBoost algorithms, we found that both the models are statistically significant. We made more such comparisons and found out the results for other models.

VI. Future Work

The main impediment we faced in training the model was the scarcity of data. Even though we had around 8000 tuples for training, we really feel that business-intensive tasks like these should be backed by at least a million rows to make models that are robust in nature. We could have applied much more complex deep learning models.

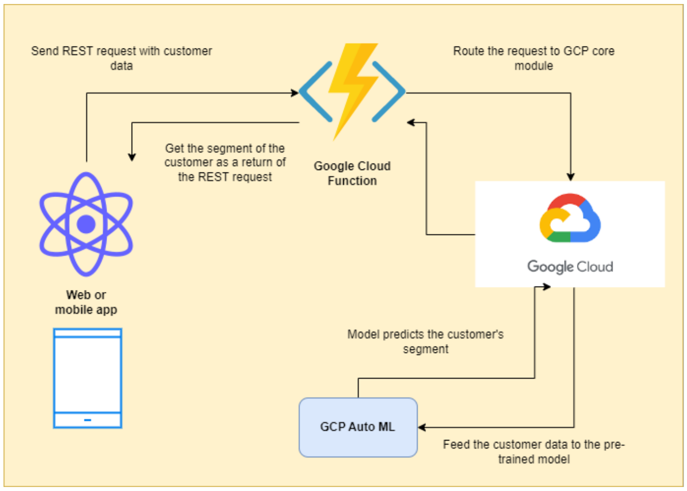

Send REST request with customer data
Route the request to GCP core module
Get the segment of the customer as a return of the REST request
Google Cloud Function
Google Cloud
Web or mobile app
Model predicts the customer's segment
GCP Auto ML
Feed the customer data to the pre-trained model

Fig. 13. Handling real-time classification using GCP.

The main application of a machine learning model is providing prediction or classification for the task. Our problem domain of Customer Segmentation would be useful to the automobile firm only when the model is easily accessible to the manager as explained in the introduction part of this report. To make this possible, we thought of a solution as shown in Fig. 4. where in we will deploy either a web or a mobile app on the business side that can take input as details of a new potential customer. To get the customer segment, we will deploy a pre-trained XGBoost (given that it performed the best in our experiments) to GCP Auto ML. So, the business can input the details of the new customer in the UI, and just press a "Submit" button which will send a REST API request to GCP using GCP Cloud Functions as a mediator, this will thus feed the model with these collected values and then the model will then classify this new customer into a segment and send this result over to the UI via GCP Cloud Function back again. If this performs well, we can think of selling this product of ours as a SaaS (Software-As-A-Service) to any businesses that want on-the-go customer segmentation.

References

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.
[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
[3] I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
[4] K. Elissa, “Title of paper if known,” unpublished.
[5] R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.