Project Report: Starbucks Capstone Challenge

Jared Gan

Udacity – AWS Machine Learning Nanodegree 2023

**Introduction:**

Starbucks is one of the most popular coffeehouse chains in the world with a presence in over 80 countries. The company has a mobile app that enables customers to pay for their orders, earn rewards, and avail of special offers. The challenge for Starbucks is to analyze customer behavior on the app and identify which demographic groups respond best to which offer types. This project proposes a machine learning solution to predict customer behavior upon receiving offers from Starbucks.

**Problem Statement:**

The goal of this project is to build a machine learning model that can predict which customers are more likely to make a purchase upon receiving offers. Starbucks sends out different types of offers to its customers, including Buy One Get One (BOGO), discounts, and informational offers. Not all customers receive the same offer, and some customers may not receive any offer during certain weeks. Therefore, the challenge is to analyze the customer behavior on the Starbucks rewards mobile app and identify which demographic groups respond best to which offer types.

**Proposed Solution:**

To address this challenge, the project proposes the following steps:

1. Exploratory data analysis (EDA): Perform EDA on the provided dataset to understand the relationship between customer demographics, offer types, and customer behavior on the mobile app.
2. Data preprocessing: Preprocess and clean the data, ensuring data integrity and addressing any missing values.
3. Machine learning techniques: Use machine learning techniques to build a predictive model that can determine which customers respond best to offers.
4. Hyperparameter optimization: Optimize the model's hyperparameters using techniques such as grid search or Bayesian optimization.
5. Evaluation: Evaluate the model's performance on the test set and compare it with the performance of other models.

**Deliverables:**

The deliverables of this project will be a well-documented codebase that includes data preprocessing, EDA, and a predictive model. The model should be able to predict which customers are more likely to make a purchase upon receiving offers. The project will be presented in the form of a technical report that documents the entire project's scope, methodology, results, and conclusions. The report will also include visualizations to help explain the findings and a detailed discussion of the limitations and future scope of the project.

**Evaluation Metric:**

The F1 score will be used as the primary evaluation metric for this project. The F1 score is a measure of a model's accuracy that considers both precision and recall. By using the F1 score as our primary metric, we can ensure that our model performs well in terms of both precision and recall, which are both important for our project's goals.

**Project Design:**

1. The project will be divided into the following steps:
2. Set up the environment: Create an Amazon SageMaker notebook instance and set up the required environment.
3. Import necessary libraries: Import the necessary libraries such as pandas, numpy, seaborn, matplotlib, and sagemaker.
4. Data loading and exploration: Load the Starbucks dataset into the notebook instance and explore it to gain a better understanding of the data.
5. Data preprocessing: Preprocess the data by performing tasks such as data cleaning, feature engineering, and feature scaling.
6. Data splitting: Split the data into training, validation, and test sets.
7. Machine learning model training: Train a machine learning model using one of the available algorithms such as XGBoost, Random Forest, or Deep Learning.
8. Hyperparameter optimization: Optimize the model's hyperparameters using techniques such as grid search or Bayesian optimization.
9. Model evaluation: Evaluate the model's performance on the test set and compare it with the performance of other models.

**Data Sets**

The data is contained in three files:

\* portfolio.json - containing offer ids and meta data about each offer (duration, type, etc.)

\* profile.json - demographic data for each customer

\* transcript.json - records for transactions, offers received, offers viewed, and offers completed

Here is the schema and explanation of each variable in the files:

portfolio.json

* id (string) - offer id
* offer\_type (string) - type of offer ie BOGO, discount, informational
* difficulty (int) - minimum required spend to complete an offer
* reward (int) - reward given for completing an offer
* duration (int) - time for offer to be open, in days
* channels (list of strings)

profile.json

* age (int) - age of the customer
* became\_member\_on (int) - date when customer created an app account
* gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
* id (str) - customer id
* income (float) - customer's income

transcript.json

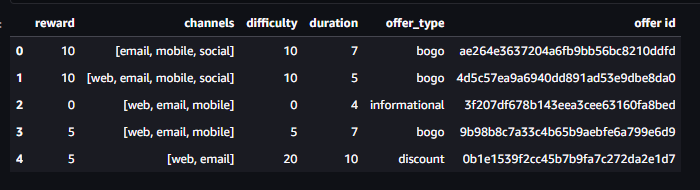
* event (str) - record description (ie transaction, offer received, offer viewed, etc.)
* person (str) - customer id
* time (int) - time in hours since start of test. The data begins at time t=0
* value - (dict of strings) - either an offer id or transaction amount depending on the record

**Perform exploratory data analysis and pre-process the dataset by performing feature engineering, handling missing values, and encoding categorical variables**

portfolio.json

clean version:

No data manipulation required, updated column to “offer id” for dataframe merging.



profile.json

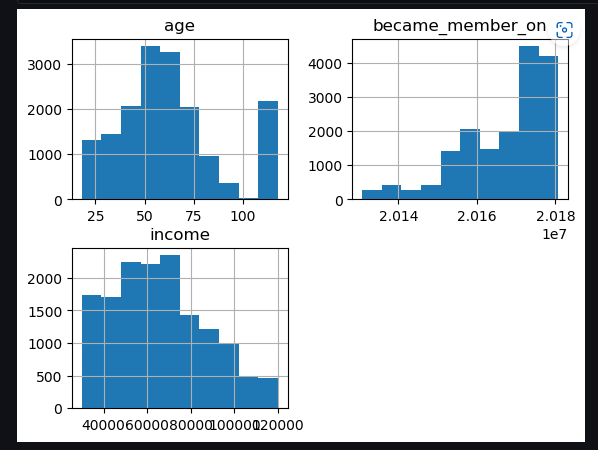
Observed irregular data due to age feature as seen in the histogram.

This was most likely caused by customer not providing their age/date of birth and 118 is the default value the app registers

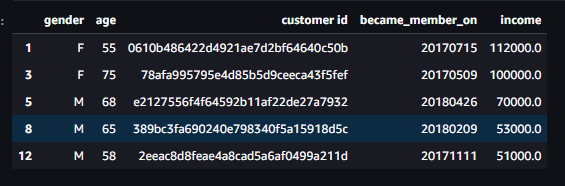
In the process of dropping invalid age data, null gender data was also removed

This implies that customers that did not provide age/date of birth, also did not provide their gender

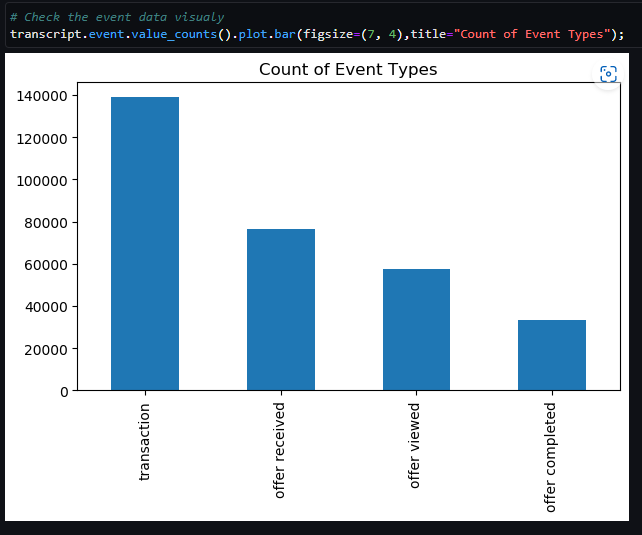
This then mplies such customers prefer not to disclose their personal data (age/date of birth and gender)

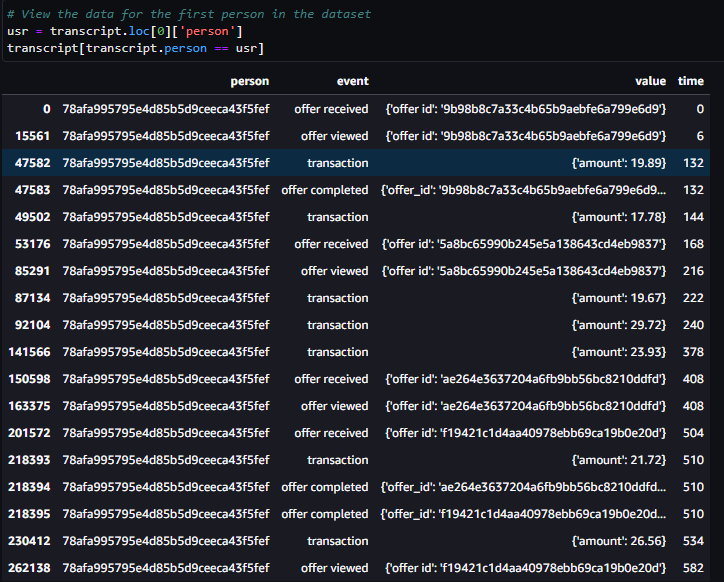


clean version:



transcript.json





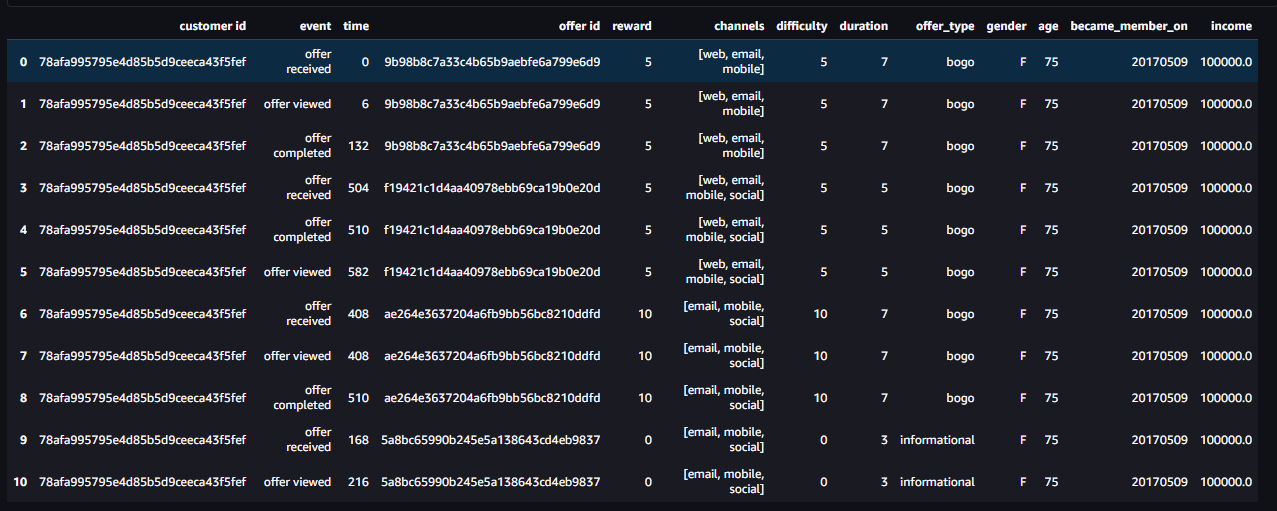
Data indicates that this dataset is a list of historical transactions (multiple transactions per individual)

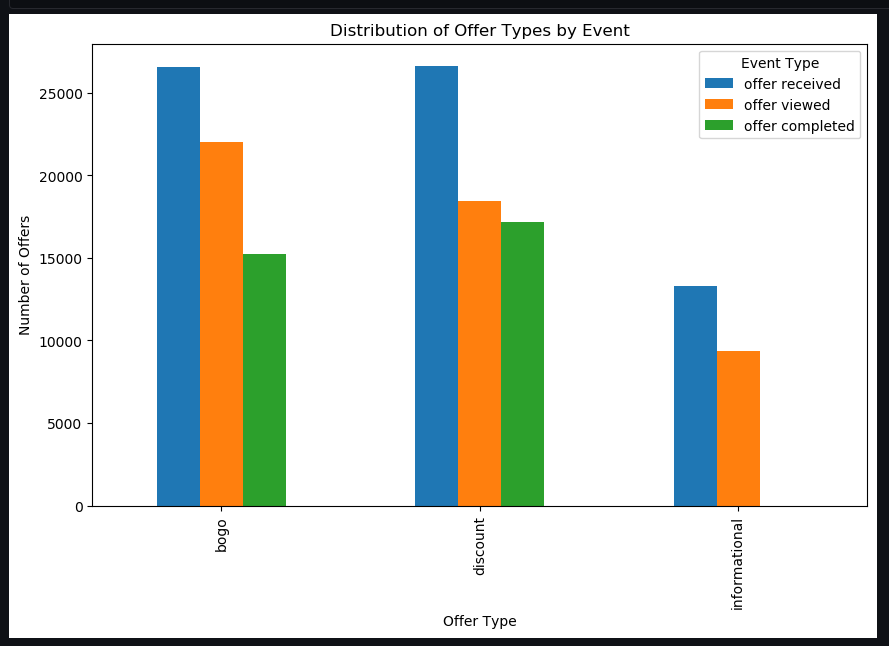
Dropping and renaming some columns:

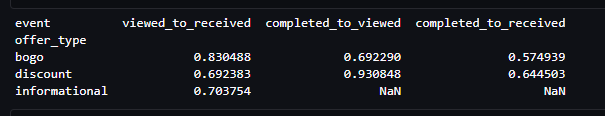


**Final Merged Dataframe**

Combine the 3 dataframes based on ‘customer id’ and ‘offer id’.

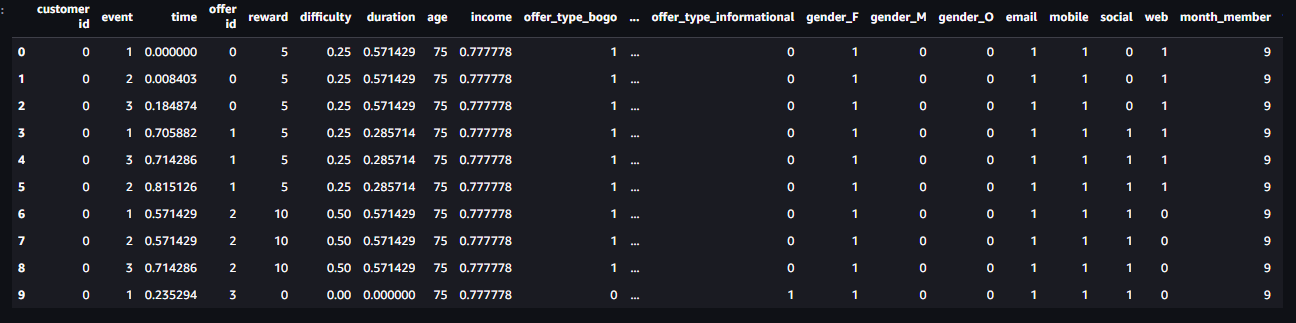
****

****

****

The data tells us that customers are extremely receptive to “bogo” offers if they see the offer having a conversion rate of 93.08%.

Perform feature engineering, final dataframe for building machine learning model:



Then we proceed to split the data into training dataframe and test dataframe to train and test the machine learning models.

**Benchmark model selection**

K-Nearest Neighbor (KNN) algorithm is a popular choice for benchmarking classification models, especially in cases where the dataset is relatively small, and the number of features is not too high. In this project, KNN was used as a benchmark model to predict the event response of customers to offers on the Starbucks rewards mobile app because it is a simple, non-parametric algorithm that is easy to implement and interpret.

KNN works by calculating the distance between each data point in the dataset and its K nearest neighbors based on some similarity metric. The predicted class for a given data point is then the majority class of its K nearest neighbors. The choice of K can significantly affect the performance of the model, and it is usually determined using cross-validation.

KNN has been widely used for customer segmentation in the retail industry, including the analysis of customer behavior, purchase history, and demographic information. Some examples of KNN applications in customer segmentation can be found in the following references:

* Hua, Y., Cheng, K. T., & Kannan, P. K. (2014). Customer clustering based on purchase sequences: Evidence from a large-scale dataset. Journal of Retailing, 90(4), 518-535. doi: 10.1016/j.jretai.2014.06.001
* Mikhaylov, S. J., & Wittink, D. R. (2012). Market segmentation using K-means clustering. Journal of Marketing Analytics, 1(4), 217-237. doi: 10.1057/jma.2013.3
* Ortega, F. J. G., Rodríguez, M. A. V., Pérez, J. A. R., & Yáñez, J. A. R. (2013). Customer segmentation based on shopping basket analysis using unsupervised techniques. Expert Systems with Applications, 40(8), 3160-3166. doi: 10.1016/j.eswa.2012.12.004

Overall, KNN is a popular choice for benchmarking classification models and has been used effectively in customer segmentation for the retail industry.

**Project Refinement and Improvement:**

To improve upon the initial solution, we can explore various approaches. Here are a few steps that we can take to improve the solution:

1. Fine-tune the hyperparameters: The performance of machine learning models is highly dependent on the hyperparameters chosen. We can perform a grid search or a randomized search over the hyperparameter space to find the optimal values for our model.
2. Feature Engineering: We can create additional features or transform existing ones to provide more information to the model. For example, we can add a feature that represents the ratio of the customer's income to the offer difficulty.
3. Model selection: We can experiment with different models, such as Decision Trees, Random Forests, Gradient Boosting, or Neural Networks, to see if they outperform the K-Nearest Neighbor model.
4. Ensemble models: We can combine the predictions of multiple models, such as K-Nearest Neighbor, Random Forest, and Gradient Boosting, using a weighted average to improve the overall performance.
5. Data Augmentation: We can generate synthetic data using techniques such as SMOTE (Synthetic Minority Over-sampling Technique) or ADASYN (Adaptive Synthetic Sampling), to increase the representation of the minority class and improve the model's ability to detect true positives.
6. Transfer Learning: We can use pre-trained models, such as BERT or GPT, to extract features from the text data and use them in our model. This can potentially improve the accuracy of the model.
7. Increase the size of the dataset: We can collect more data or use data from other sources to improve the model's performance. This can be particularly useful if we observe that the model is overfitting or underfitting the data.

Overall, we need to try different approaches and see what works best for our problem. By experimenting with different techniques and algorithms, we can refine and improve the model's performance.

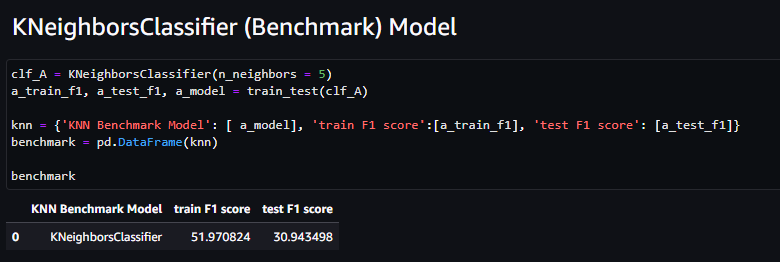
Once the improvements are incorporated, we can report both the initial and final solutions, along with intermediate solutions, if necessary. We can also compare the performance of the different models and techniques used to determine which approach worked best.

In this project,

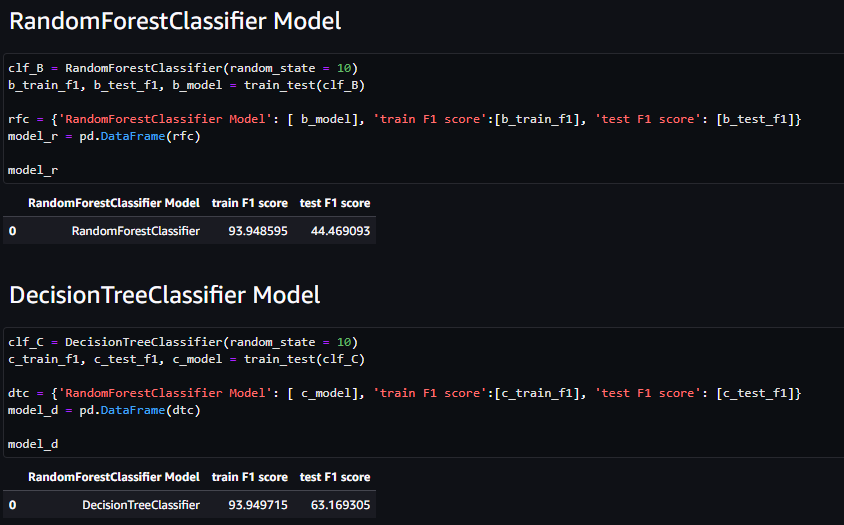
* ‘Feature Engineering’ was performed earlier to process categorical variables, normalize numerical features and obtain more features.
* ‘Model selection’, we will 3 models, K-Nearest Neighbor as the benchmark, two additional models: Random Forest and Decision Tree.
* ‘Fine-tune the hyperparameters’, of the best performing models from ‘Model Selection’, to fine tune hyperparameters to yield a model that results in the best F1-scores.

**Train and test the machine learning models**

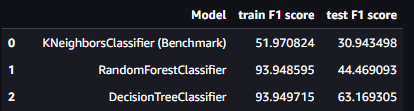
K-Nearest Neighbor algorithm is trained and tested as the benchmark model.



Two other models were also trained as part of model refinement and improvement, namely the Random Forest algorithm and Decision Tree algorithm.



Looking at the F1 scores - A score of 1 indicates perfect precision and recall, while a score of 0 indicates that the model has no predictive power, Decision Tree algorithm performs the best overall as seen below:

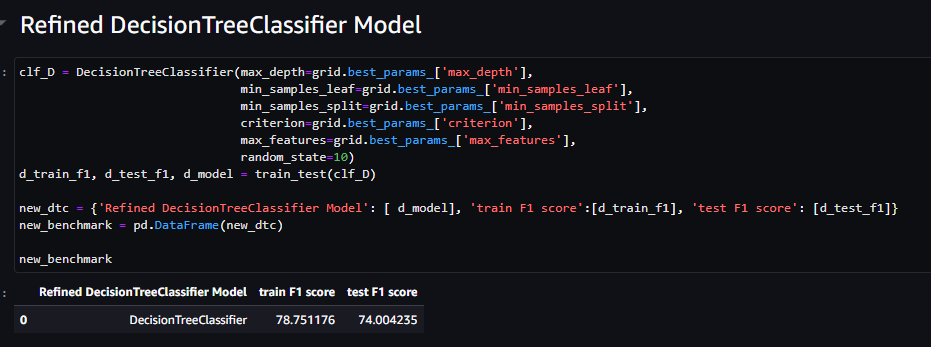


Thus, to get a further improved model proceed to tune the hyperparameters of the best performing model, the Decision Tree algorithm using GridSearchCV.

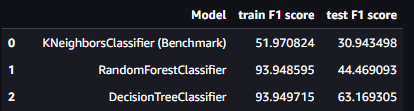


The best parameters from this tuning yields, {'criterion': 'entropy', 'max\_depth': 15, 'max\_features': None, 'min\_samples\_leaf': 2, 'min\_samples\_split': 10}.

We then plug these parameters into the Decision Tree algorithm to perform training and testing again.



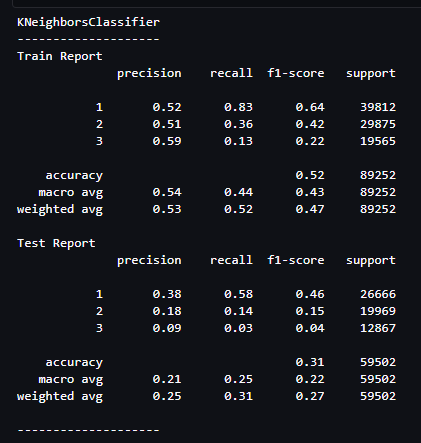
The F1-score (of the tuned Decision Tree model) looks better than the earlier 3 models.



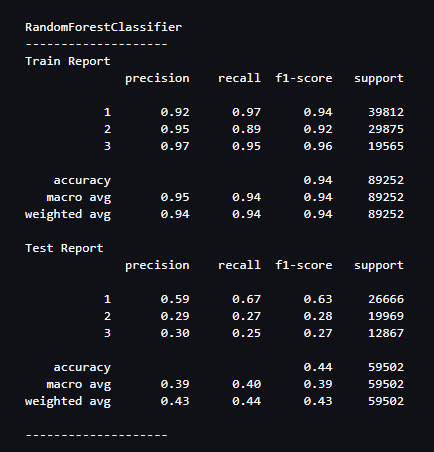


**Evaluating the models in more detail**

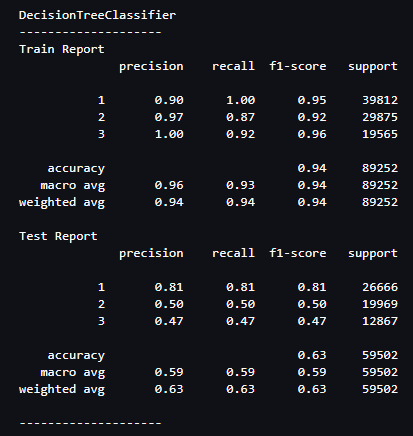
Benchmark model, K-Nearest Neighbor algorithm



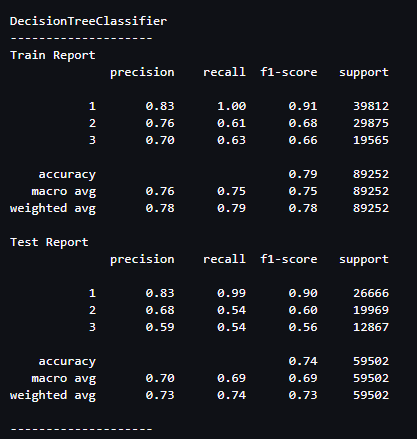
Random Forest algorithm model



Decision Tree algorithm model



Tuned model, Decision Tree algorithm



**Recap:**



Based on the F1 scores, the KNeighborsClassifier (Benchmark) model as the benchmark model performs the poorest of the trained models.

The RandomForestClassifier model performs better by being able to predict the training data well, but predicts relatively poorly when it comes to the testing data set, this could be due to overfitting.

The DecisionTreeClassifier model performs even better by being able to predict the training data well, but predicts moderately when it comes to the testing data set, this could also be due to overfitting.

The Refined DecisionTreeClassifier model which had undergone hyperparameter tuning would be the best model, as it performs "well" for both the training and test data set suggesting it generalizes well.

**Conclusion:**

The proposed solution aims to analyze the customer behavior on the Starbucks rewards mobile app and identify which customers respond best to offers. The final model (Refined DecisionTreeClassifier Model) serves moderately well as a base model to be further developed by collecting more data for further training, testing other algorithm which might be developed in future, etc. With further development, the machine learning model will eventually be able to accurately predict customer behavior.

In conclusion, this project has the potential to provide Starbucks with valuable insights into customer behavior on their mobile app, enabling them to tailor their marketing campaigns to better reach and engage their customers. By using machine learning techniques, we can predict customer behavior accurately and improve the overall customer experience.