Section 1: Project Definition

Overview:
This project performs data analysis on Starbucks promotional dataset in order to discover which customers will be receptive to a promotion. The purpose is to optimize Starbucks promotional strategy to encourage customers to spend more. Customer receptiveness was gauged based on a combination of user and promotional data.

Dataset:
There are three files:

1. Portfolio.json - table of promotions
2. Profile.json - User data for each customer
3. Transcript.json - Transaction records

Schemas:
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

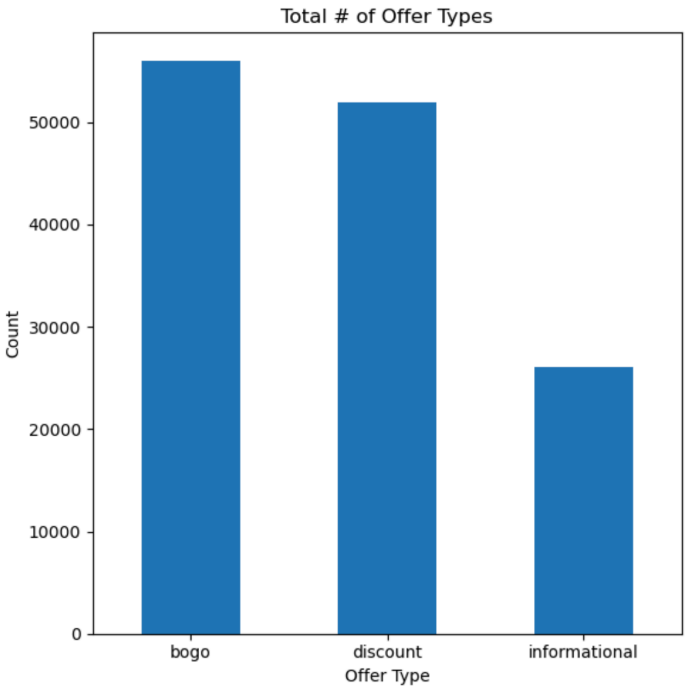
- age (int) - age of the customer
- became_member_on (int) - date when customer created an app account
- gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
- id (str) - customer id
- income (float) - customer's income

transcript.json

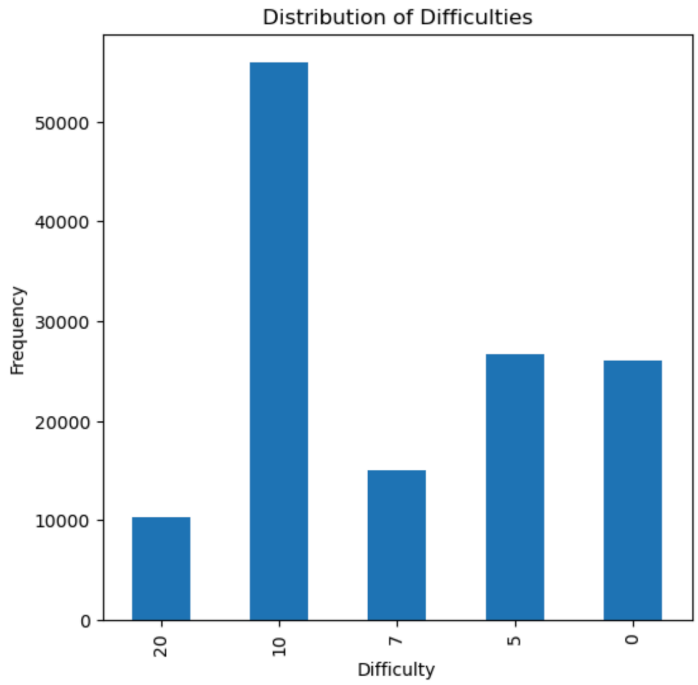
- event (str) - record description (ie transaction, offer received, offer viewed, etc.)
- person (str) - customer id
- time (int) - time in hours since start of test. The data begins at time t=0
- value - (dict of strings) - either an offer id or transaction amount depending on the record

All three datasets were eventually merged and passed into a random forest pipeline.

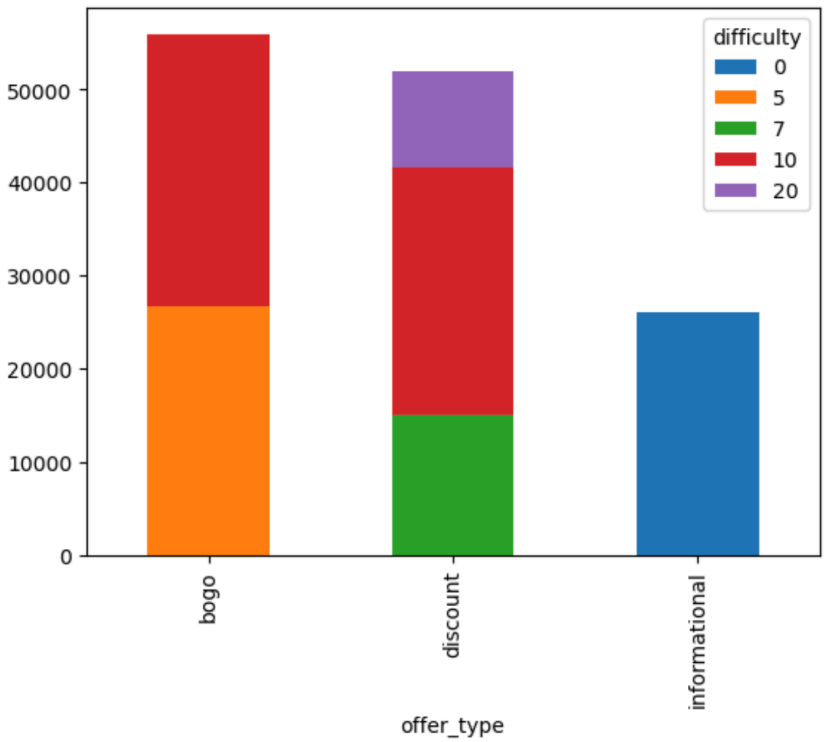
Data Visualization



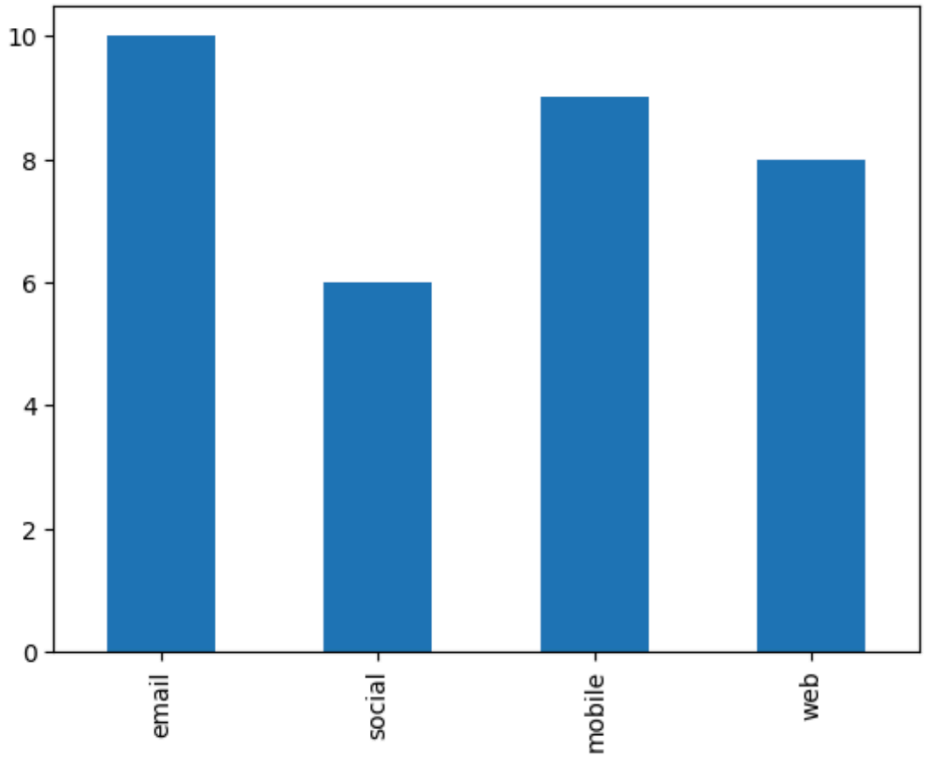
It seems the least offered type of discount is informational, followed by discount and bogo. There appears to be a preference towards discount strategies.



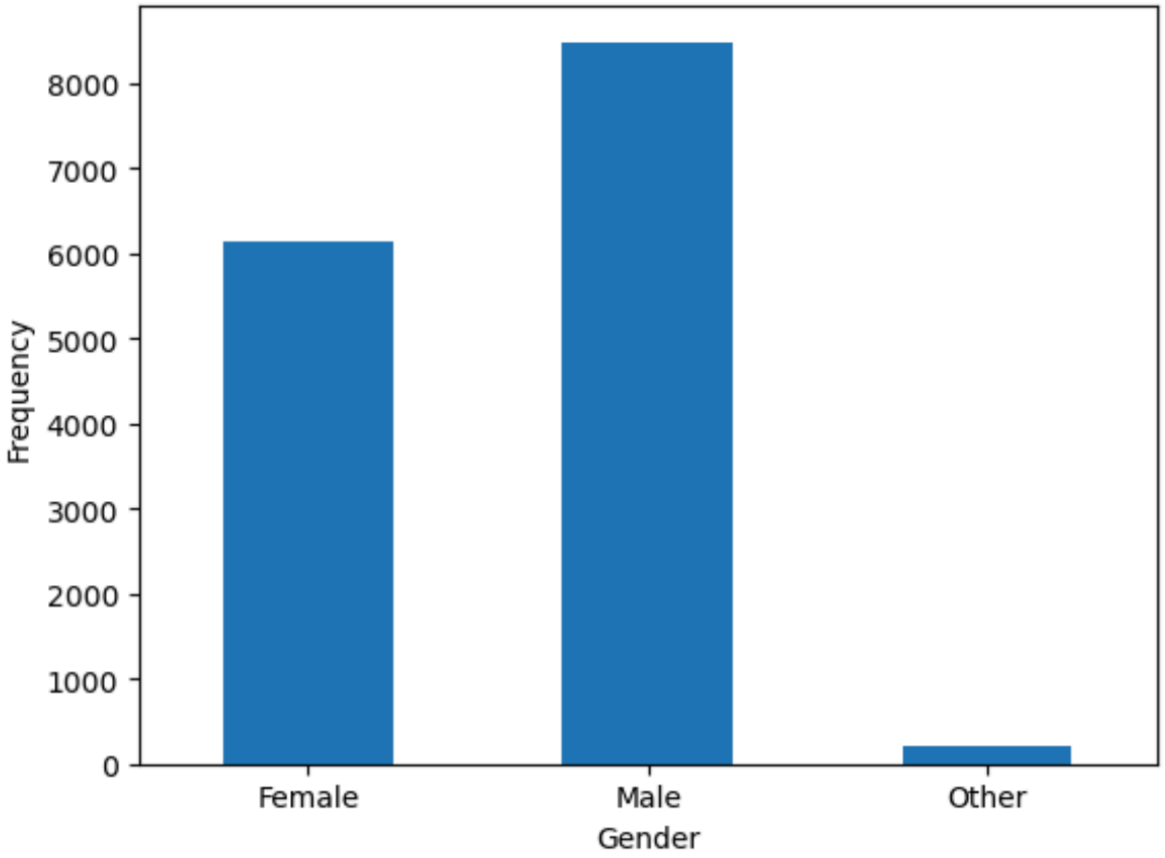
According to the chart, promotions with higher minimum purchase thresholds are more frequent.



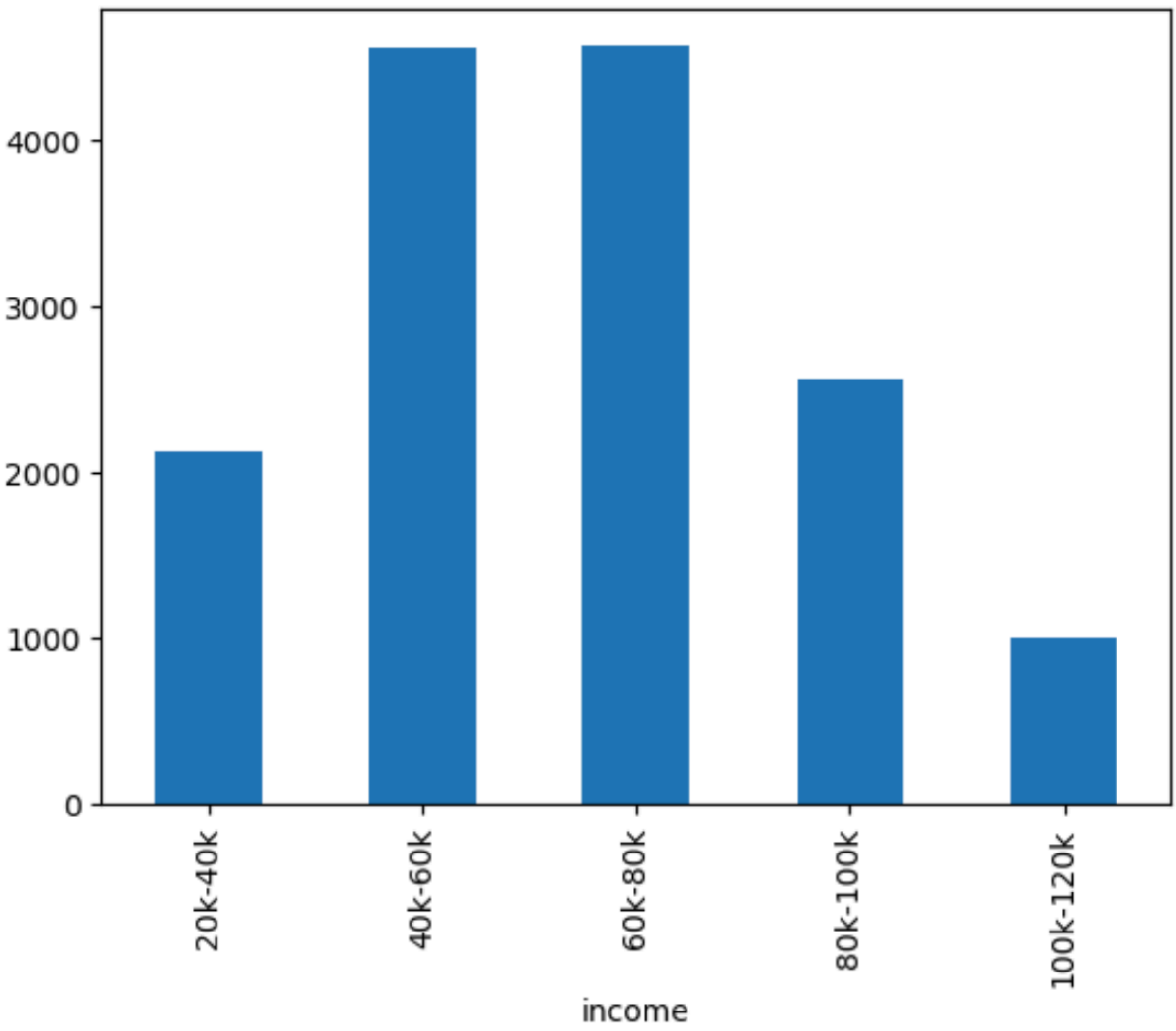
Combining these two categories reveals that between the ‘bogo’ and ‘discount’ type promotions, ‘bogo’ is on average less difficult to redeem than traditional discounts.



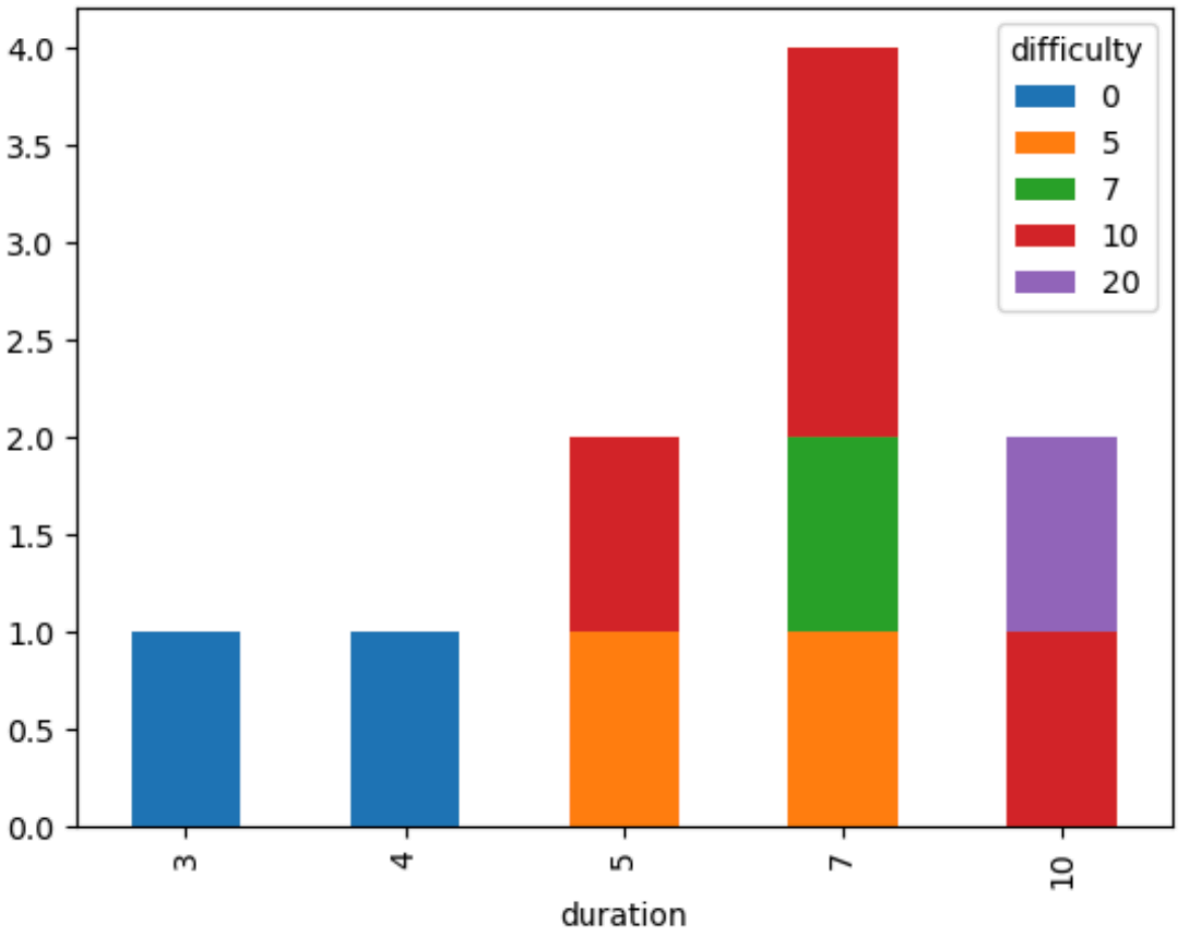
Despite the small sample size, it seems that all four channels are evenly represented.



There appear to be significantly more male customers represented than female.



Most customers appear to be within the 40-80k range. This may be attributed to the population mean income at the time the data was collected.



Shorter-dated promotions are strictly informational, while Starbucks gives customers between 5-10 days to redeem all other types.

Section 3: Methodology

I first started by defining the problem statement: Predict which customers are going to positively react to the ads. A quick and dirty way to attract consumer spending would be randomly distribute ads across the entire population (ie. all customers irrespective of group). Aside from being uneconomical due to ad costs, there is the risk of customers responding negatively by treating them as spam.

Alternatively, analyzing data collected through user profiles and transaction history can reveal spending patterns that can be used to tailor targeted ad strategies. Determining which customers would respond positively to ads based on income, gender, or other data would save costs and drive up sales. So the question is: how do we go about determining which customers to target, and which ones we should not?

First, the data was cleaned of problematic nulls and unrealistic outliers. For example, there were some user accounts who listed users as 118 years old. The data was then split into training and test sets and trained on a ML pipeline using a random forest classifier. Random Forest was selected due to its ease of use and good performance when it comes to capturing non-linear relationships between discrete features.

Precision and recall were used as the main performance metrics. Precision will indicate what percentage of customers that will receive promotions will actually respond positively. A high precision would demonstrate efficient ad targeting, thereby reducing wasted resources on uninterested customers. A high recall would indicate how, out of all the customers who would positively respond to ads, what percentage of that group the model correctly identifies. This ensures the company is not missing out on potential responders.

Cleaning:

I noticed that there were three main values for gender: Female, Male, and Other. There were also some marked as Null. Further inspection revealed the Null rows also had invalid values for income and age, so they were removed.

The value column was a dict, which forced me to separate out the individual keys into columns on their own. I extracted the offer_id, and noticed that some keys were ‘offer id’ without the underscore, so I made sure to include those in the final column.

I then defined what would be considered a success or failure. If a promotion is successful, the customer will have received, read, and redeemed the promotion. Otherwise, it would not be possible to determine that the customers actions were directly influenced by the ad.

Performance:

	precision	recall	f1-score	support
0	0.76	0.77	0.76	7668
1	0.64	0.62	0.63	4990
accuracy			0.71	12658
macro avg	0.70	0.70	0.70	12658
weighted avg	0.71	0.71	0.71	12658

The Random Forest classifier shows moderate performance with a weighted precision and recall of 0.7, so on average the model accurately identifies which customers are receptive to ads ~70% of the time. The rather low precision indicates that the model is identifying a lot of false positives.

In order to tune our model, I used GridSearchCV to optimize the hyperparameters for the task.

Param grid I used:

```
param_grid = {
    'classifier__n_estimators': [200, 300, 400, 500],
    'classifier__max_depth': [15, 20, 25, 30],
    'classifier__min_samples_split': [5, 10, 15],
    'classifier__min_samples_leaf': [2, 4, 6],
    'classifier__max_features': ['sqrt', 'log2'],
    'classifier__class_weight': ['balanced', 'balanced_subsample']
}
```

Justifications:

- The class weight was used because the classes in the target variable were imbalance (7668 negative vs 4990 positive)
- In order to prevent overfitting, I set various max depths and # of estimators.

Optimal params:

```
Fitting 5 folds for each of 576 candidates, totalling 2880 fits
Best parameters: {'classifier__class_weight': 'balanced', 'classifier__max_depth': 15, 'classifier__max_features':
'sqrt', 'classifier__min_samples_leaf': 4, 'classifier__min_samples_split': 15, 'classifier__n_estimators': 300}
Best cross-validation score: 0.7166037973668528
```

- N_estimators : 300
- Max_depth : 15
- Max_features: sqrt
- Min_samples_leaf : 4
- Min_samples_split : 15
- Class_weight : balanced

After fitting the grid search with the above parameters, I was able to achieve better overall performance:

	precision	recall	f1-score	support
0	0.85	0.70	0.77	7668
1	0.64	0.82	0.72	4990
accuracy			0.74	12658
macro avg	0.75	0.76	0.74	12658
weighted avg	0.77	0.74	0.75	12658

There is higher precision for identifying non-receptive individuals as well as higher recall for identifying candidates for ads. The higher precision suggests that the new model is better for identifying what customers that respond negatively to ads, while also identifying what customers would respond to an ad.

Let's try comparing it to a SVC pipeline to judge which model is better:

```python
# SVM param grid
svm_param_grid = {
    'classifier__C': [1],
    'classifier__kernel': ['rbf'],
    'classifier__gamma': ['auto'],
    'classifier__class_weight': [None, 'balanced']
}
```

```
Best parameters: {'classifier__class_weight': 'balanced', 'classifier__max_depth': 15, 'classifier__max_features':
'sqrt', 'classifier__min_samples_leaf': 4, 'classifier__min_samples_split': 15, 'classifier__n_estimators': 300}
Best cross-validation score: 0.7166037973668528
              precision    recall  f1-score   support

           0       0.85      0.70      0.77      7668
           1       0.64      0.82      0.72      4990

    accuracy                           0.74     12658
   macro avg       0.75      0.76      0.74     12658
weighted avg       0.77      0.74      0.75     12658
```

It looks like using a SVM model with minimal tuning is already yielding comparable if not better results than the random forest classifier. This tells me that both models may be being bottlenecked by the quality of data, and that both models have reached the limits of their utility. It is also possible that SVC is more robust to overfitting than random forest considering it didn't require much tuning in order to achieve similar performance to the tuned random forest example.

Further tuning:

Considering that informational promotions cannot be accurately measured since they are not redeemable, it may be appropriate to remove this data entirely. It may also make sense to experiment with other models that perform well with discrete data such as decision trees.

Reflection:

Overall, using traditional ML methods is not perfect - given the current dataset, we can only expect about 75% accuracy when it comes to identifying candidates for ads. It is possible that both models are limited by the information available in both datasets. The fact that both models exhibit almost identical performance in precision, recall, and f1 scores across all classes shows the current features in the dataset may not be predictive enough to achieve higher performance regardless of what model is used. Further analysis may be warranted by analyzing feature importances in the Random Forest model in order to understand what features are most/least predictive, as well as feature engineering to capture more complex relationships between features. It is also possible that there needs to be more thorough checks for data quality issues such as null values and other inconsistencies.