Capstone Project – Machine Learning Engineer

An analysis on Starbucks simulated customer data

C. Deserno

November 09th 2020

Definition

I chose the starbucks dataset to build my first machine-learning project. My choice was this because i do not have much experience with own data projects and thought it would be an example easy enough to follow which turned out to not exactly be true.

The data we are looking at is artificial customer data that shows how customers react to offers given to them via different media channels and which they can convert into rewards in at least some of the offer types.

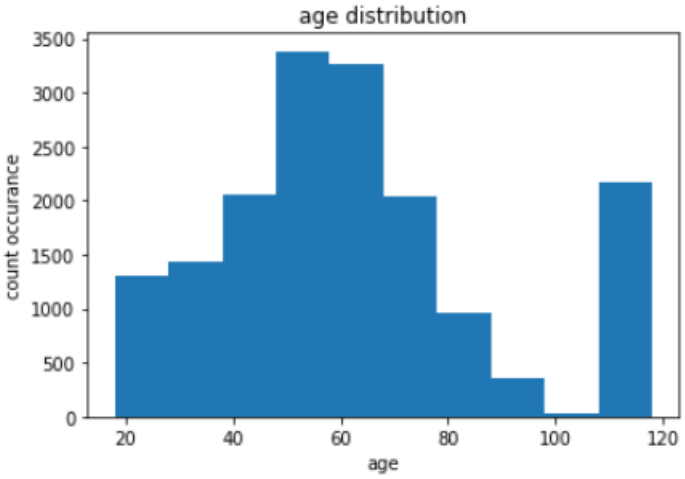
The datasets and inputs

The given dataset consists of three tables

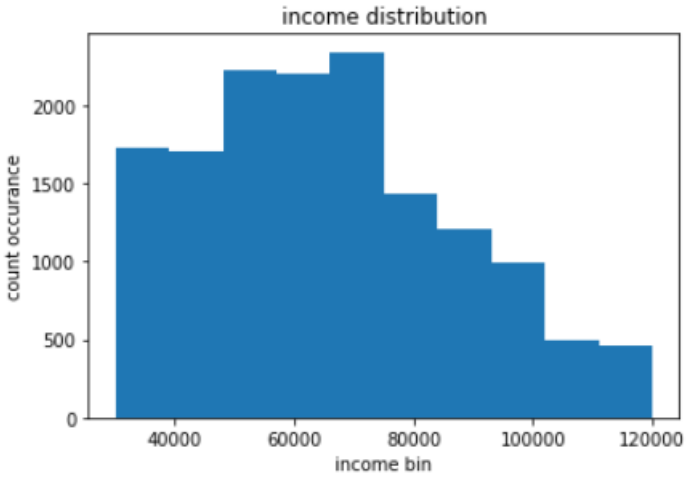
- Portfolio
  The portfolio holds information about what kind of offers are used for advertising. Three different offer types are present. A “buy one get one” (BOGO), a discount offer and just pure informational offers. The first two offers come with a reward measured in dollars while the informational offer is just not rewarded and pure advertisement. Each offer is described by a unique id, a duration for how long the offer is valid, a difficulty meaning the value a customer must spend to be able to receive the award of the offer, the offer type (BOGO, discount, informational), the channels how the offers are broadcasted (email, mobile, social, web) and a reward.

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	168	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	120	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	96	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	168	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	240	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7
5	3	[web, email, mobile, social]	7	168	discount	2298d6c36e964ae4a3e7e9706d1fb8c2
6	2	[web, email, mobile, social]	10	240	discount	fafdcd668e3743c1bb461111dcafc2a4
7	0	[email, mobile, social]	0	72	informational	5a8bc65990b245e5a138643cd4eb9837
8	5	[web, email, mobile, social]	5	120	bogo	f19421c1d4aa40978ebb69ca19b0e20d
9	2	[web, email, mobile]	10	168	discount	2906b810c7d4411798c6938adc9daaa5

- Profile
  The profile table holds information about the customers. Mainly demographic data can be found as gender, age and income. Also an information about when a person became customer via the app. The profile data frame contains implausible values, which need to be taken care of. A distribution of the age reveals that there is a larger ratio of customers having 118 years of age which is quite implausible.



The distribution of income seems to be plausible at first glance



The following graphic displays the data frame as it is

	gender	age	id	became_member_on	income
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
5	M	68	e2127556f4f64592b11af22de27a7932	20180426	70000.0
8	M	65	389bc3fa690240e798340f5a15918d5c	20180209	53000.0
12	M	58	2eeac8d8feae4a8cad5a6af0499a211d	20171111	51000.0
...	...	...	...	...	...
16995	F	45	6d5f3a774f3d4714ab0c092238f3a1d7	20180604	54000.0
16996	M	61	2cb4f97358b841b9a9773a7aa05a9d77	20180713	72000.0
16997	M	49	01d26f638c274aa0b965d24cefe3183f	20170126	73000.0
16998	F	83	9dc1421481194dcd9400aec7c9ae6366	20160307	50000.0
16999	F	62	e4052622e5ba45a8b96b59aba68cf068	20170722	82000.0

- Transcript

  The transcript table contains data about different types of events. These events can be transactions or offer related status information. It is listed, how much a customer spent at what time or they can hold information on offer status. This information can either be

	person	event	value	time
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	0
1	a03223e636434f42ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0
2	e2127556f4f64592b11af22de27a7932	offer received	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}	0
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer received	{'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}	0
4	68617ca6246f4fbc85e91a2a49552598	offer received	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	0
...	...	...	...	...
306529	b3a1272bc9904337b331bf348c3e8c17	transaction	{'amount': 1.5899999999999999}	714
306530	68213b08d99a4ae1b0dcb72aebd9aa35	transaction	{'amount': 9.53}	714
306531	a00058cf10334a308c68e7631c529907	transaction	{'amount': 3.61}	714
306532	76ddbd6576844afe811f1a3c0fbb5bec	transaction	{'amount': 3.5300000000000002}	714
306533	c02b10e8752c4d8e9b73f918558531f7	transaction	{'amount': 4.05}	714

whether an offer is received, whether it is viewed or completed and information about when this event took place. Rewards are listed as well.

I would like to build a model that can predict whether a customer will convert on an offer meaning whether he or she will be willing to spend money on the offer. This model can be used to check which offer type should be rolled out to which type of customer by just forward prediction of a customers' feature or behavior.

Different models or at least different parameterizations will be tested for a good fit. Metrics should be recall and precision regarding a conversion of a customer.

Metrics

A variety of metrics is available to judge whether a model performs well or not so well. The task is to predict whether a customer will react positively on an offer he or she receives. As a company I want to make sure to send out offers rather to a customer who is likely to convert. A high recall value regarding a predicted value of "customer converts being true" means that most of the customers that would actually convert are identified by the model as potential customers. The recall regarding "customer converts being false" is less important here assuming that a sent out offer that is neglected by the customer does not require a lot of money effort by the company.

As second important metric one can still look at the precision. A general good precision should be in place to not address too many customers with offers they would not react to. This can lead to a higher customer satisfaction and thus again to higher purchases even without offers being sent.

The recall however remains as the major metric

Analysis

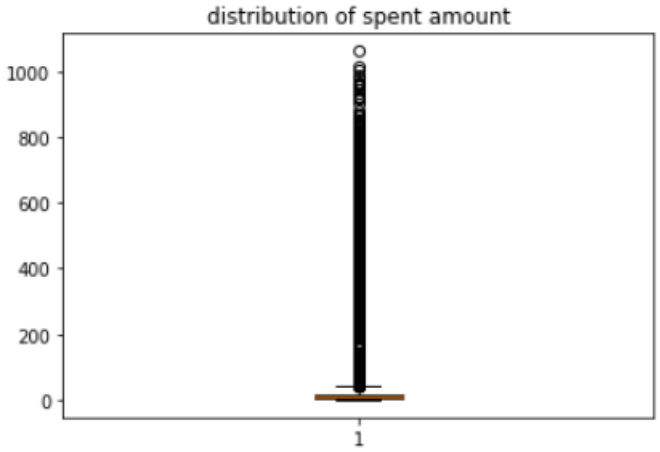
Preprocessing tasks

While looking at the data for the first time it became clear that the data preprocessing step will take a larger part in this project. Examples why this is the case are listed below.

- The portfolio table contains nested information regarding the type of broadcasting on channels, this needs to be extracted
- The profiles contain incomplete data sets having null values in income and gender and implausible high age
- The transcript data again holds nested information in the value column which needs to be extracted

Besides these basic cleansing techniques there are more preparation steps to take.

- spending seem to have a large variation, that might decrease the quality of the models ability to predict correctly



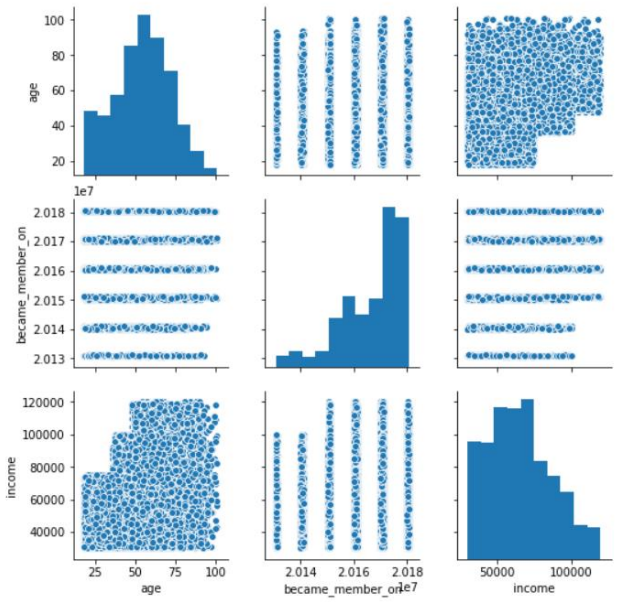
- There is no information about when an informational offer is completed, this needs to be defined. For this we define a validity period that lets us check whether an informational offer was followed by a transaction within a some hours after the offer has been viewed. This transaction can then be taken as offer completion so we can amend the transcript data frame by simulations of completed offers in the described case.

- In some cases there is overlapping of offers. These overlapping offers cannot be assigned to certain transactions or otherwise one transaction would be counted to multiple offers regarding their completion. To avoid this we should consider to eliminate offers that overlap. To do so might be followed in later steps if there is time

- The way how offers are listed in the transcript data is explained in the following typical walk through. An offer is usually rolled out and results in a received status with the customer at a ceratin point of time. This is represented as the first entry in the data set. As soon as the customer views the offer this is recognized and results in a second row. When enough money is spent within the offer, a new row will be created and marked as completed. This would be the standard procedure. But offers can also be completed even if they have not been viewed. Offers can also be viewed after they have been completed. For these special cases, special treatment has to be developed.

Label distribution

Several remarks are to be made about the label distribution and class imbalance. In the profiles data set male and female customers are balanced regarding count, other gender is existing in the data set yet it only makes up a small ration of about xxx percent. One cannot expect to create a highly reliable model regarding this branch of demographic data.



As is revealed by the pairplot above, also income distribution is tending to incomes lower than $80.000 for higher – not as dramatically as for gender of type other, this can influence the predictive capabilities of the model. Generally the more ‘fresh’ customers are present within the dataset, this is also an imbalance that might reflect in prediction accuracy

Looking at the transcript data set

It was already shown that an imbalance exists due to few but very high expenses in. The broadcast of offer types seems to be very evenly distributed and each offer makes out about 10 percent of the total.

Feature engineering

The main feature to be considered will be the conversion regarding an offer - whether a customer bought something as a response to an offer. The conversion rate is the quotient of conversion counts divided by total offers viewed. Only viewed offers can be called convertible because an offer a customer does not view can hardly lead to an influence. For creating the features we have transformed the transcript data set and enriched it. Offers that are listed in the transcript data set are transformed to one row per offer as it is explained being necessary above. This row holds information about when the offer was received, viewed and completed. Having this information accumulated in one data set, the following features can be created.

The feature engineering will cover the different steps of creating features like this:

- average time deltas between received and viewed
- average time deltas between viewed and completed
- columns for converted per offer type
- number of transactions
- average spending
- accumulation of viewed offers
- accumulation of completed offers
- accumulated reward

After having calculated the features of the whole transcript data frame we can have a look into the data before creating models.

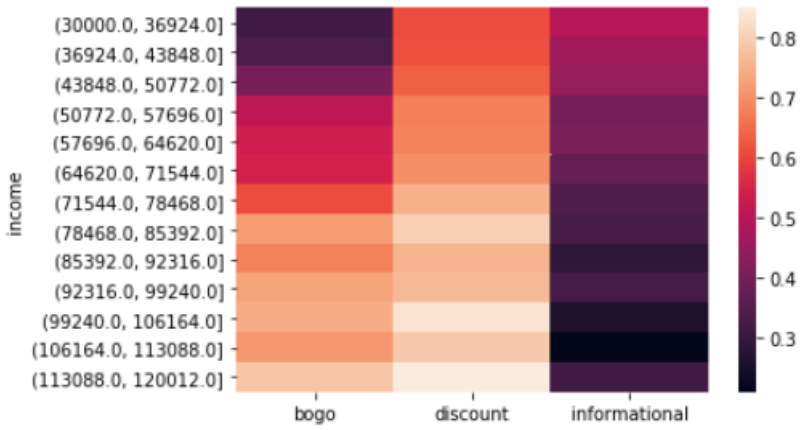
First lets have a look at the general conversion rate

```
bogo - conversion:  0.5387339978748166
discount - conversion:  0.7046728971962617
informational - conversion:  0.3854700854700855
```

We can also split the conversion rate by demographic data like gender

gender	bogo	discount	informational
F	0.650673	0.766456	0.375448
M	0.456973	0.660086	0.392891
O	0.676157	0.723247	0.385093

Where a table gives great overview on very limited data only, we can check the dependency on income better in a heat map



Higher income seems to correlate with higher conversion rates, discount generally seems to have a higher conversion rate than bogo offers. Informational offers range very low in general, lower incomes seem to correlate with higher conversion in this offer type

Algorithms and techniques

I will mainly use xgboost as model algorithm for predicting the conversion of a customer. It said to be used for many problems in kaggle competitions. Xgboost stands for extreme gradient boosted trees and works in a way that it concatenates or chains models together so that misclassification by trees upstream the chain can be corrected downsteram. By this chaining of models weaker parts of the chain and stronger parts of the chain level out and produce an after all stronger or more precise model It is easy to train

The algorithm can be tuned by different parameters. For a hyper parameter tuning the following are common to play around with

- max_depth: describes the depth of trees used – or simply how many decisions can be made after another. If set too high, it can lead to overfitting easily.
- eta: this refers to the learning rate meaning adjusting the weights when iterating another step. Strong changes might counteract already good learning where as too weak changes might not reach the goal of optimization
- min_child_weight: is a measure for how complex a model should possibly become thinking of it as how many paths to leafs should possibly being created. A higher number of min_child_weight means the model gets more complex and thus yields overfitting
- subsample: once per boosting iteration xgboost will take onle this percentage of data to train
- gamma: gamma can be seen as another parameter to set the possible complexity of the model so it must as well be watched for possible overfitting.

For a comparison a linear model will be used to create a base line. Sklearns linear learner will be used.

Implementation

First I want to implement a simple model for a benchmark. Generally it seems plausible that the features fit well into a regression. SKLearn offers easy to use models and metrics to make a first check on outcomes. Lets have a look into the logistic regression of the linear_model Class.
In this approach I simply trained a model with base parameters without any tuning or optimization. The data set is only split into training and test data, no validation is done during training.

A classification report shows already results that are not too bad for an untrained model

	precision	recall	f1-score	support
0.0	0.63	0.59	0.61	2717
1.0	0.67	0.71	0.69	3212
micro avg	0.65	0.65	0.65	5929
macro avg	0.65	0.65	0.65	5929
weighted avg	0.65	0.65	0.65	5929

Precision and Recall as the most interesting values for metrics are defined as

Recall = TP/(TP + FN)

Precision = TP/(TP + TN)

T begin True, P being positive, N being negative

For the task at hand to check whether an offer will result in a conversion of a customer we will chose the recall on conversion being True as the major metric.

Now that we have a benchmark with a quite simple model lets check out if the metric values can be raised by other models or hyper parameter tuning.

For this we switch to Amazon sagemaker and use stronger virtual machines than the one the notebook is running on.

The training data still consisted of a combination of all offer types. To make a model one needs to feed each model with only one offer type to check which type of customer will convert on which type of offer. I tested a few different scenarios

- one model training on each offer type, no hyper parameter tuning, demographic data only
- one model training on each offer type, no hyper parameter tuning, complete feature set
- training one offer type on multiple models using hyper parameter tuning, demographic data only
- training one offer type on multiple models using hyper parameter tuning, complete feature set

Further training results are mentioned in the jupyter notebook.

Results

One can clearly see that data derived from a complete feature set is more precise than only selecting demographic data. The original task was to chose demographic data only to infer which offer might be best for a customer, however more detailed knowledge about a customer and his recent behavior seem to yield better results on prediction.
Results on hyper parameter tuned xgboost

Demographic data only – with one model per offer type	Complete feature set – with one model per offer type
bogo [[1964 753] [982 2230]] precision recall f1-score support 0.0 0.67 0.72 0.69 2717 1.0 0.75 0.69 0.72 3212 micro avg 0.71 0.71 0.71 5929 macro avg 0.71 0.71 0.71 5929 weighted avg 0.71 0.71 0.71 5929 discount [[488 994] [362 3132]] precision recall f1-score support 0.0 0.57 0.33 0.42 1482 1.0 0.76 0.90 0.82 3494 micro avg 0.73 0.73 0.73 4976 macro avg 0.67 0.61 0.62 4976 weighted avg 0.70 0.73 0.70 4976 informational [[1455 292] [666 395]] precision recall f1-score support 0.0 0.69 0.83 0.75 1747 1.0 0.57 0.37 0.45 1061 micro avg 0.66 0.66 0.66 2808 macro avg 0.63 0.60 0.60 2808 weighted avg 0.64 0.66 0.64 2808	bogo [[1331 708] [491 2486]] precision recall f1-score support 0.0 0.73 0.65 0.69 2039 1.0 0.78 0.84 0.81 2977 micro avg 0.76 0.76 0.76 5016 macro avg 0.75 0.74 0.75 5016 weighted avg 0.76 0.76 0.76 5016 discount [[497 572] [343 2815]] precision recall f1-score support 0.0 0.59 0.46 0.52 1069 1.0 0.83 0.89 0.86 3158 micro avg 0.78 0.78 0.78 4227 macro avg 0.71 0.68 0.69 4227 weighted avg 0.77 0.78 0.77 4227 informational [[1309 187] [636 292]] precision recall f1-score support 0.0 0.67 0.88 0.76 1496 1.0 0.61 0.31 0.42 928 micro avg 0.66 0.66 0.66 2424 macro avg 0.64 0.59 0.59 2424 weighted avg 0.65 0.66 0.63 2424

Demographic data only – xboost after hyper parameter tuning:	Complete feature set – xboost after hyper parameter tuning:
[[1939 778] [962 2250]]	[[1352 687] [494 2483]]
precision recall f1-score support 0.0 0.67 0.71 0.69 2717 1.0 0.74 0.70 0.72 3212 micro avg 0.71 0.71 0.71 5929 macro avg 0.71 0.71 0.71 5929 weighted avg 0.71 0.71 0.71 5929	precision recall f1-score support 0.0 0.73 0.66 0.70 2039 1.0 0.78 0.83 0.81 2977 micro avg 0.76 0.76 0.76 5016 macro avg 0.76 0.75 0.75 5016 weighted avg 0.76 0.76 0.76 5016
[[1104 378] [1272 2222]]	[[687 382] [824 2334]]
precision recall f1-score support 0.0 0.46 0.74 0.57 1482 1.0 0.85 0.64 0.73 3494 micro avg 0.67 0.67 0.67 4976 macro avg 0.66 0.69 0.65 4976 weighted avg 0.74 0.67 0.68 4976	precision recall f1-score support 0.0 0.45 0.64 0.53 1069 1.0 0.86 0.74 0.79 3158 micro avg 0.71 0.71 0.71 4227 macro avg 0.66 0.69 0.66 4227 weighted avg 0.76 0.71 0.73 4227
[[875 872] [432 629]]	[[503 993] [252 676]]
precision recall f1-score support 0.0 0.67 0.50 0.57 1747 1.0 0.42 0.59 0.49 1061 micro avg 0.54 0.54 0.54 2808 macro avg 0.54 0.55 0.53 2808 weighted avg 0.57 0.54 0.54 2808	precision recall f1-score support 0.0 0.67 0.34 0.45 1496 1.0 0.41 0.73 0.52 928 micro avg 0.49 0.49 0.49 2424 macro avg 0.54 0.53 0.48 2424 weighted avg 0.57 0.49 0.48 2424

Conclusion

Training different models revealed higher accuracy when choosing models with higher complexity, the task at hand was partially to explain also the dependencies so a less complex model should be chosen. A well trained linear regression might still be able to explain most of the variance in the data.

Future projects might want to look into the traps of these data sets in more detail. Some issues were not explainable for example overlapping offers for customers. The data preprocessing took most of the time since a lot of complexity is sitting within cleaning the data or determining the criteria for when an informational offer is completed. The cleansing and plausibilisation steps might not yet be perfect Training and deploying models came a little short after a lot of time was spent on preparation and feature engineering.

The trained models can all be used to predict a customer behavior and to derive whether he or she should be given an offer of kind bogo, discount or informational.

Further investigation is necessary why generally recall and precision are less on predictions of non-conversions (zeros in the classification report)