# Starbucks Challenge! Find Hidden Customer Segments with Unsupervised Learning.

Raphael Bürki
Feb 4, 2019


[Source]

**Create distinct segments based on purchasing behavior using PCA and k-means. Learn about the importance of fine-tuned preprocessing. Data provided by Starbucks and Udacity.**

---

This blogpost focuses on finding a strategy to discover patterns for offer type preference under messy conditions: difficult data, missing experimental conditions. Details and code can be found on GitHub.

**Introduction**

The data contains 3 sets of 'events', demographic customer data and offer data. It is simulated and mimics the behavior for 17,000 customers on the Starbucks rewards mobile app over the course of 29 days. Once every few days, Starbucks sends out an offer to users of the mobile app.

The following simple plot, showing the distribution of roughly 317,000 'events' over the trial period, shows that the general strategy seems to work: A release of offers is followed by offer viewings on the app, a raise in transactions and eventually, if a certain spending threshold is met, with an offer completion that leads to a reward (Fig 1).

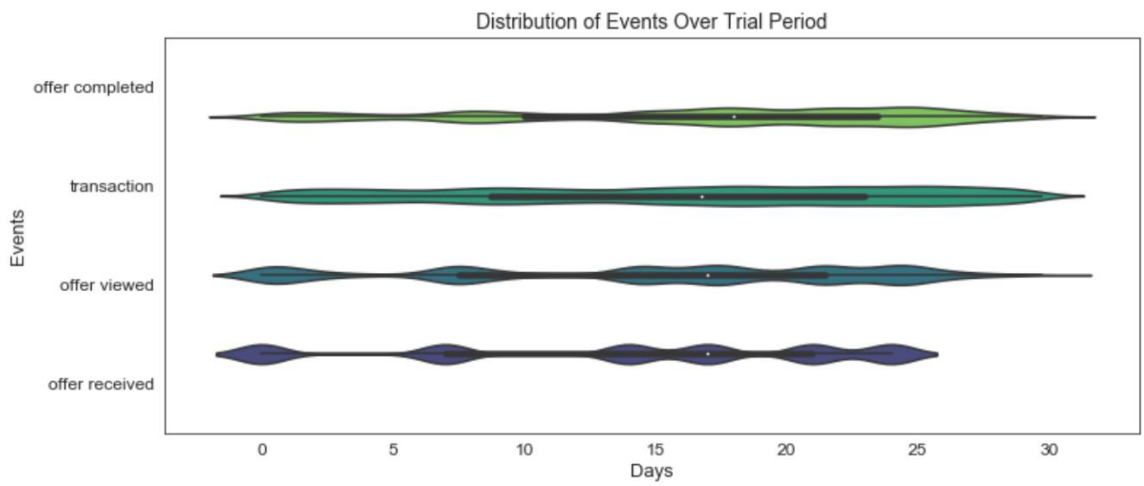



Fig 1: The different types of events in a timeline

On average a customer received 4–5 offers during the test period and viewed 3–4 of them. Totally there's 10 different offers that come in *three types*:

1. a discount offer (discount)
2. a buy-one-get-one-free offer (BOGO)
3. merely an advertisement for a drink (info)

**The goal of the project is to determine which customer groups respond best to which offer type.**

I will present my findings in 6 short sections:

1. Cleaning / Feature Engineering
2. High-level Comparison of Offer Types
3. Modelling: PCA and Clustering
4. Segment Analysis Based on Customer Behavior
5. Conclusions on Demographic Groups
6. Learnings

But before we start, let me say that there's two things making this project a real challenge:

1) *There is no apparent experimental setup and we have no control over the variables*: Not all users receive the same offers / offer types or the same amount of offers. During certain weeks some customers may receive multiple offers at the same time, while others don't receive any at all. Also, the different offers have different characteristics making a successful completion more or less likely.

2) *The transaction data needs extensive and pretty complex preparation.* The transactions a customer makes are not linked to any offers she has received. And even when an offer is sent to a customer, this doesn't mean she has viewed it and hence is influenced by it. Also, we have to make sure that we only count the valid offer completions which occur within the defined period of validity and result in a reward.

---

**1. Cleaning / Feature Engineering**

Not too much cleaning is necessary for the two sets containing the offer characteristics and the demographic customer data. The most important thing to note is that demographic data is missing for 2,175 of the 17,000 customers — and all of them have their age set to a default value of 118 years. I replaced that with a NaN value.

I did not try to impute the missing demographic data, as I was focusing on the purchasing behavior contained in the events set and so would not need it for my modelling (I will come back to this point later).

But the events data definitely needs a lot of cleaning (Fig 2):

| | event | person | time | value |
|---|---|---|---|---|
| **139604** | transaction | 55c69bafc66d4bf6a7df7f1f752c1b38 | 372 | {'amount': 5.51} |
| **150404** | transaction | 664c0533a818495781cb3a3a1c5cc5e6 | 402 | {'amount': 24.63} |
| **79400** | transaction | 2f3964e445744ce29ecd95c7656fbf22 | 192 | {'amount': 1.29} |
| **16671** | offer viewed | 2f937953414c4fc5aae319f8ba4d441c | 6 | {'offer id': '5a8bc65990b245e5a138643cd4eb9837'} |
| **305641** | transaction | 7b74e68b25754abb968231adc71c7e3a | 714 | {'amount': 14.84} |
| **265959** | transaction | 6bdd08b358ac44cbb9b379ff6ad6e9cc | 588 | {'amount': 29.99} |
| **154639** | offer received | d5059ee547a24f9a92c5c6c9892e469d | 408 | {'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'} |
| **2916** | offer received | a3627a341176408c9512c6c6a8378458 | 0 | {'offer id': '3f207df678b143eea3cee63160fa8bed'} |
| **296335** | transaction | c099206f76b1414db7552f163520053c | 672 | {'amount': 0.29} |
| **171317** | offer viewed | b527eab600344530991f159dffc3ac53 | 420 | {'offer id': '5a8bc65990b245e5a138643cd4eb9837'} |

Fig 2: A sample of the raw input data of the 317,000 events. The events are all assigned to a person, but the transactions are not assigned to any offers.

After some tidying (and recoding of the person IDs and offer IDs to make the whole data more readable) the event data looks much better (Fig 3):

| | event | person_id | time | amount | offer_id |
|---|---|---|---|---|---|
| **48537** | transaction | p_88 | 138 | 14.13 | NaN |
| **101463** | transaction | p_88 | 282 | 15.20 | NaN |
| **110892** | offer received | p_88 | 336 | NaN | o_5 |
| **147228** | transaction | p_88 | 396 | 24.29 | NaN |
| **147229** | offer completed | p_88 | 396 | NaN | o_5 |
| **150660** | offer received | p_88 | 408 | NaN | o_6 |
| **177275** | offer viewed | p_88 | 432 | NaN | o_6 |
| **191230** | transaction | p_88 | 468 | 17.53 | NaN |
| **191231** | offer completed | p_88 | 468 | NaN | o_6 |
| **196846** | transaction | p_88 | 486 | 13.72 | NaN |
| **201631** | offer received | p_88 | 504 | NaN | o_9 |
| **243239** | offer viewed | p_88 | 570 | NaN | o_9 |
| **268861** | transaction | p_88 | 594 | 18.81 | NaN |
| **296332** | transaction | p_88 | 672 | 33.26 | NaN |

Fig 3: Events for random customer ‘p_88’ after basic cleaning (and ID recoding)

But there is still no assignment of transactions to offers — this is the second of the two problems mentioned above. To make sure that I only assigned transactions and completions that were made under valid conditions (within period of validity, after viewing, before completion), I applied a 2-stepped process:

1. Flag all events that could be linked to an offer starting from the moment it was received until the end of it’s period of validity. (I did this with help of the offer’s ‘duration’ feature).
2. Make sure that of those events flagged in step one only those count that occur after the actual viewing of the offer and only up to the moment the customer has completed the offer.

*A word of warning: I did this the best I could, with a couple of encapsulated for-loops. On my laptop the procedure iterated for about 20 (!) hours on 317,000 rows over 10 offer columns … (no details here, but you can have [a look at the code] if you want).*

| | event | person_id | time | amount | offer_id | o_1 | o_2 | o_3 | o_4 | o_5 | o_6 | o_7 | o_8 | o_9 | o_10 | reward |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **273587** | transaction | p_88 | 138 | 14.13 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **273588** | transaction | p_88 | 282 | 15.20 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **273589** | offer received | p_88 | 336 | NaN | o_5 | NaN | NaN | NaN | NaN | 0.0 | NaN | NaN | NaN | NaN | NaN | NaN |
| **273590** | transaction | p_88 | 396 | 24.29 | NaN | NaN | NaN | NaN | NaN | 0.0 | NaN | NaN | NaN | NaN | NaN | NaN |
| **273591** | offer completed | p_88 | 396 | NaN | o_5 | NaN | NaN | NaN | NaN | 0.0 | NaN | NaN | NaN | NaN | NaN | 5.0 |
| **273592** | offer received | p_88 | 408 | NaN | o_6 | NaN | NaN | NaN | NaN | NaN | 0.0 | NaN | NaN | NaN | NaN | NaN |
| **273593** | offer viewed | p_88 | 432 | NaN | o_6 | NaN | NaN | NaN | NaN | NaN | 1.0 | NaN | NaN | NaN | NaN | NaN |
| **273594** | transaction | p_88 | 468 | 17.53 | NaN | NaN | NaN | NaN | NaN | 0.0 | 1.0 | NaN | NaN | NaN | NaN | NaN |
| **273595** | offer completed | p_88 | 468 | NaN | o_6 | NaN | NaN | NaN | NaN | NaN | 1.0 | NaN | NaN | NaN | NaN | 3.0 |
| **273596** | transaction | p_88 | 486 | 13.72 | NaN | NaN | NaN | NaN | NaN | 0.0 | 0.0 | NaN | NaN | NaN | NaN | NaN |
| **273597** | offer received | p_88 | 504 | NaN | o_9 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 0.0 | NaN | NaN |
| **273598** | offer viewed | p_88 | 570 | NaN | o_9 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 1.0 | NaN | NaN |
| **273599** | transaction | p_88 | 594 | 18.81 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 1.0 | NaN | NaN |
| **273600** | transaction | p_88 | 672 | 33.26 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

Fig 4: Events for the same customer 'p_88' after flagging with first with 0 (Step 1) and then with 1 (Step 2)

Explanation of the final, 'flagged' events data (Fig 4): '0' denotes events that can be assigned to an offer ('o_1' to 'o_10'), be it because there is an offer ID attached to them, or, in the case of transactions, because they occured within the valid offer duration. If a customer has received several offers at the same time, a transaction can be assigned to more than one offer (for example the transaction at hour 486). Now '1' denotes events that occurred after the customer has viewed an offer and before it expires or is completed (whatever comes first). Let's again look at transaction 486: I have *not* counted it for offer 5 because the customer already completed that offer and even if he had not, he actually never saw that offer in his app (so was never influenced by it). And I did *not* count it for offer 6 because the customer already completed that one before, too. (But in this case he has seen the offer, so I counted the transaction at hour 468, that led to the completion and the reward).

This tricky data preparation provided a solid base for the subsequent steps.

---

**2) High-level Comparison of Offer Types**

| offer_id | difficulty | duration | offer_type | reward |
|---|---|---|---|---|
| o_1 | 10 | 7 | bogo | 10 |
| o_2 | 10 | 5 | bogo | 10 |
| o_3 | 5 | 7 | bogo | 5 |
| o_4 | 5 | 5 | bogo | 5 |
| o_5 | 20 | 10 | discount | 5 |
| o_6 | 7 | 7 | discount | 3 |
| o_7 | 10 | 10 | discount | 2 |
| o_8 | 10 | 7 | discount | 2 |
| o_9 | 0 | 4 | informational | 0 |
| o_10 | 0 | 3 | informational | 0 |

Fig 5: The 10 offers and their main characteristics — 'difficulty' is the amount to be spent within the 'duration' (in days) to get the 'reward'. (As you can see informational campaigns don't have spending thresholds and rewards.)

But let's first have a look at the different offers and their characteristics (Fig 5). I believe there are too many uncontrolled variables for a direct comparison. Although some general trends can be spotted (have a look at my [EDA notebook]() if you want) this will not help to solve the problem.

A more robust trend becomes visible, when we aggregate the offers by type and compare some key metrics (Fig 6):

| offer_type | difficulty | duration | reward | prop_rewards | rel_difficulty | view_to_complete |
|---|---|---|---|---|---|---|
| bogo | 7.50 | 6.0 | 7.5 | 1.000000 | 1.285714 | 0.437114 |
| discount | 11.75 | 8.5 | 3.0 | 0.269643 | 1.357143 | 0.561783 |
| informational | 0.00 | 3.5 | 0.0 | NaN | 0.000000 | 0.000000 |

Fig 6: Offers aggregated by type, with mean values and additional key metrics

First let me explain **the view-to-complete rate (vtc rate), the most important metric for me in this project**. I have calculated it as the amount of offers that a customer has actually viewed and validly completed, divided by the amount of the offers he has viewed. It can range from one (all viewed offers have also been completed in time by the respective customer, meaning he reacts well to offers) to zero (none of the viewed offers have been completed in time, meaning the customer does not react well to offers). — Drawback: it doesn't work for info offers which can not be completed.

Now, let's forget about the informational offers for a moment and compare discounts and BOGOs only. **It is clear to see, that the average vtc rate is significantly higher for discounts than for BOGOs. And that's even though the monetary rewards for BOGOs are much higher.** (Note: The higher difficulty (= spending threshold) of discounts is mostly offset by their longer duration: I calculated this as the 'relative difficulty', the mean amount to be spent per day to reach the completion threshold before the offer ends.)

So, this is interesting — **why would a company want to send out expensive BOGO offers when it can have a better activation of it's customer with the cheaper discount offers?**

At this point ***two questions*** have to be answered:

- Have the two offer types been viewed by approximately the same groups of customers?
- Are there certain customer segments that are responsible for the difference or is this a general trend?

And that's where unsupervised Machine Learning comes into play!

---

**3. Modelling: PCA and Clustering**

**Feature Selection**

The dataframe I worked with now had a row for every customer with his events aggregated in total and by offer type specifically. I also added the non-promo condition as a kind of fourth offer type.

As my end goal was to find out if the purchasing patterns differ for different demographic groups, I removed all demographic features (age, gender, income and also the duration of membership) from the set used for modelling. I think this is important to note and something that often get's done wrong.

I carefully selected which of the remaining features I would pass for PCA and clustering to avoid high correlation and bias. That proved to be quite tricky. The result of a first run with many features thrown into the mix were 6 clusters with good silhouette scores. But when I tried to analyze them, I discovered that I more or less only had separated customers which had viewed a high share of one offer type from those who had viewed higher shares of other types. Those segments wouldn't qualify for a meaningful comparison of their respective purchase preferences because they had been treated with different offer strategies.

So I tried to be more careful with my selection. The final feature configuration looked like this:

```profile.columns = ['viewed_received', 'total_trans', 'total_amount', 'prop_bogo_viewed', 'bogo_vtc', 'prop_bogo_amount', 'prop_discount_viewed', 'discount_vtc', 'prop_discount_amount', 'prop_info_viewed', 'prop_info_amount', 'prop_np_trans', 'prop_by_accident']```

- viewed_received: ratio of all offers viewed to all offers received
- total_trans / total_amount: total transactions and amount spent
- prop_np_trans: proportion of transactions made when no promo was active
- prop_by_accident: proportion of the completed offers that had not been viewed (and so were completed 'by accident' / regular spending)
```

- for every offer type:
  - *ratio of viewed offers of that type to all the viewed offers*
  - the view-to-complete rate
  - the proportion of the total amount spent under offer conditions

What did the trick for me was to introduce the highlighted feature (italic). It helped to separate the customers more clearly by the 'offer type mix' they were exposed to (you'll see what I mean in section 5).

**Data Preparation and Dimensionality Reduction**

*Outlier detection*: Both [k-means](k-means) and PCA are sensitive to outliers. And there are some serious outlying customers present in respect to the total amount they have spent. I removed all total amounts exceeding 1.5 times the IQR from the third quartile which means approx 5% of the customers. This may be controversial, but I prefer to have a more general model than to make compromises for customers that behave so special that your proposed treatments probably don't apply to them anyway.

*Normalization and Scaling*: As most of the data is skewed, I tried different kinds of scaling with and without normalization and it was especially interesting to see how different feature transformations (natural log, yeo-johnson, box-cox) led to slightly different patterns after the ensuing application of dimensionality reduction with PCA.

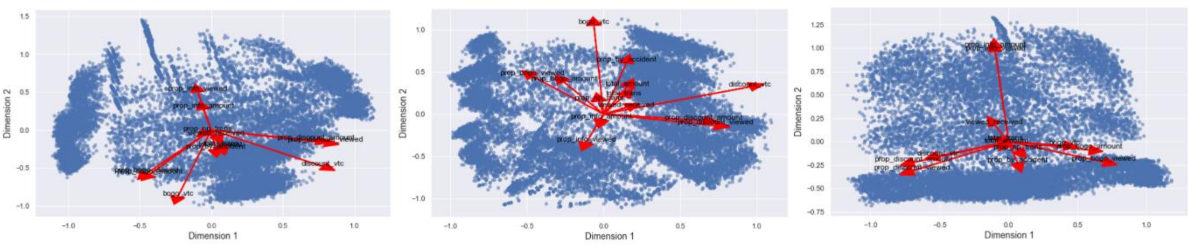



Fig 7: Different Patterns on the same data after reducing to 2 dimensions with PCA. The only input difference was the type of transformation prior to the PCA application. (From left to right: natural log, box-cox transformation, yeo-johnson transformation, all scaled to a range of 0 to 1)

I evaluated these different patterns using Biplots (Scatterplots with data reduced to 2 dimensions and overlying original feature projections, see

Figs 7 and 9) and — more important — Barplots describing the new PC-dimensions (see Fig 8 for an example). As can be seen, log and box-cox transformed data would probably yield better clustering results than yeo-johnson with its quite blurred distribution of datapoints (plot on the right in Fig 7).

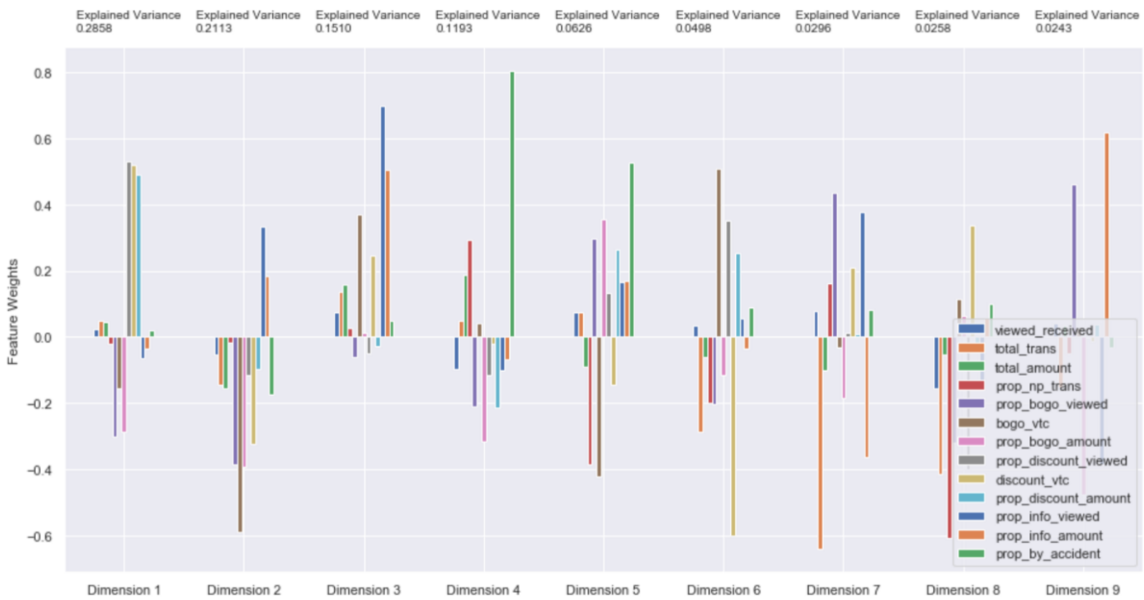



Fig 8: Nine Dimensions of PCA (95% of original variance retained) on log transformed data. Explanation: The first dimension separates customers that spent heavily on BOGOs or discounts. The fourth dimension separates customers who spent above average but did not respond well to offers.

**Clustering**

To make a long (and interesting) story short: After some testing and tuning I went for the log-transformed data (scaled to a range of 0,1), applied PCA to reduce the feature space to 2 dimensions and then clustered with k-means to have 12 segments (Fig 9).

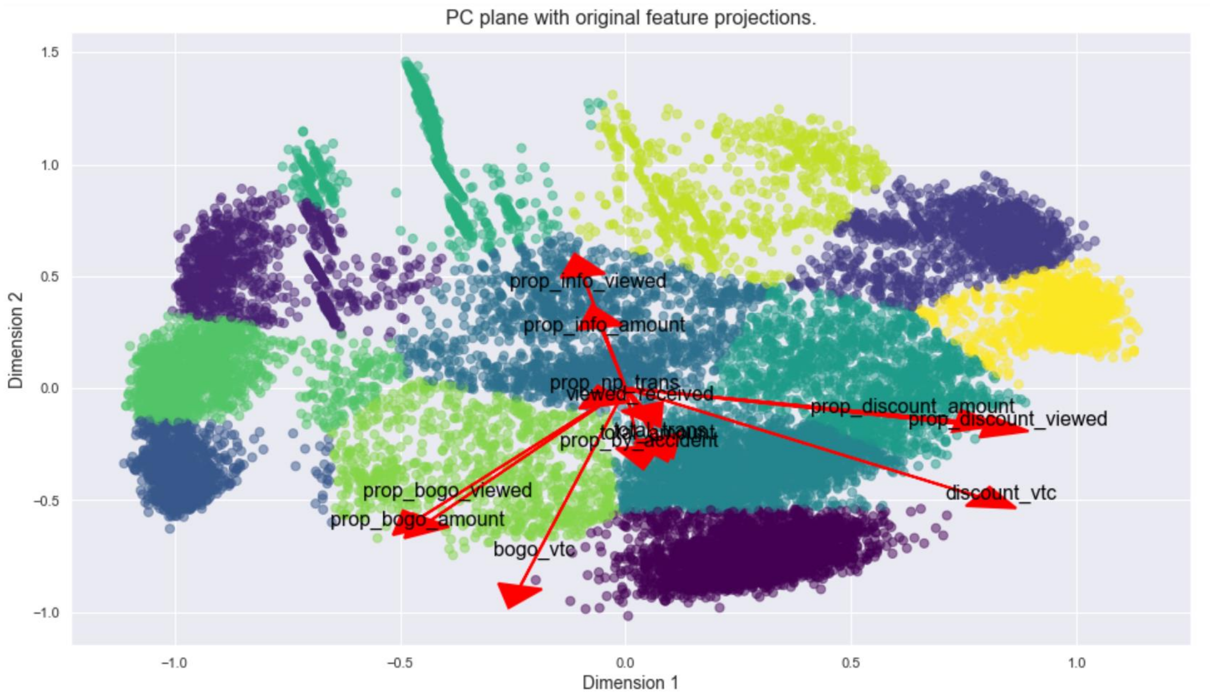



Fig 9: Scatterplot showing the 12 segments and the original feature projections on the PCA-reduced data (to 2 dimensions).

Let me explain my choices:

The main quantitative metric for evaluation was the [Silhouette score](#).

*Number of PCA dimensions*: Because a reduction to 2 dimension means quite a loss of information / original variance, I tried to cluster with less reduction (to 4 or 6 dimensions) but then I could not get good clusters (by measure of their silhouette score — see Fig 10).

*Clustering algorithm*: I tested k-means vs. hierachical clustering. k-means scored slightly better than hierarchical clustering (with ‘average’ linkage).

*Choice of number of segments*: I knew from the first run that to few segments would result in a blurred picture due to the different offer treatments customers were exposed to. So I chose the beginning of the second small peak in Silhouette scores with 12 clusters (and not the single highest score with 4 clusters — see Fig 10).

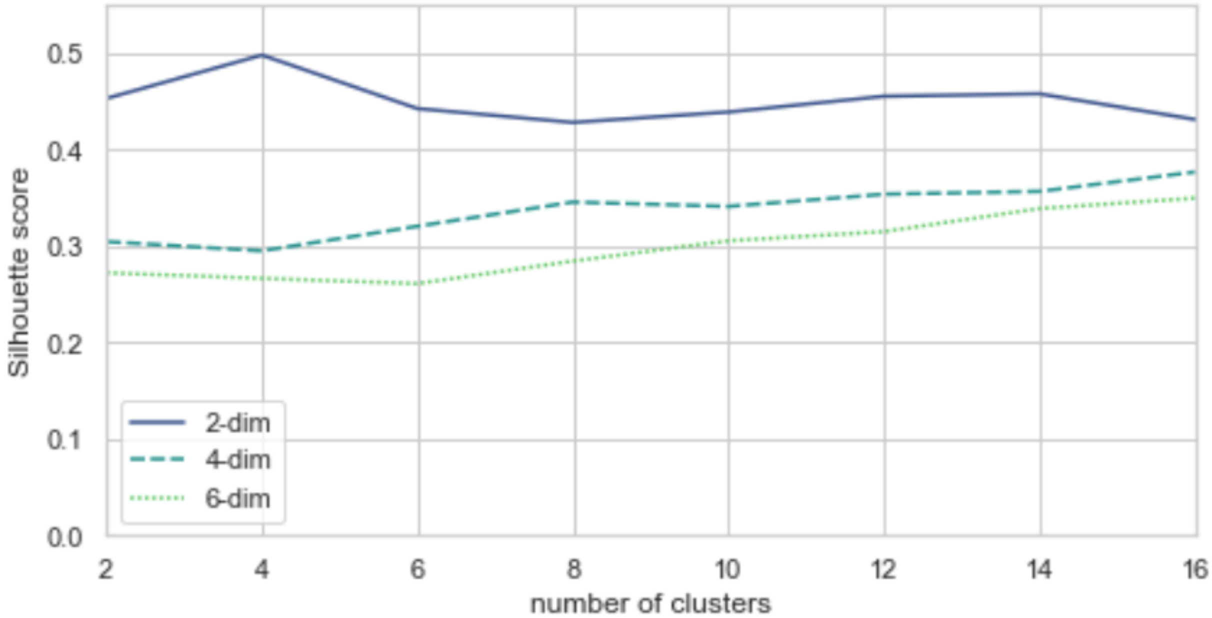



Fig 10: Silhouette scores for different number of k-means clusters on data reduced to 2, 4, 6 PC-dimensions. I chose 12 clusters on 2D-data, resulting in a silhouette score of 0.45

I want to emphasize once more that I believe it is not enough to rely on quantitative metrics like the silhouette score alone for this type of analysis. It is important to know what you are looking for and to assess the resulting segments carefully to see if they show meaningful differences in that respect.

---

**4. Segment Analysis Based on Customer Behavior**

First, please remember that the 12 resulting segments have been based on purchasing behavior only. **The cool thing is that this segmentation let's us control for the exposure of the customers to different offer types**. Now we can make meaningful comparisons!

The findings are listed below in somewhat simplified form (e.g. I won't mention info offers in the segment descriptions, but will add a remark and plots regarding them at the end).

**First group of segments** — *customers that have viewed both BOGO offers and discounts in a more or less similar share*:

- **Seg 1 (15.4% of total customers, biggest segment)**: Customers with highest spending / net revenue, react very well to discounts (mean view-to-complete rate of 0.9) and nearly as good to BOGOs (0.8). Top!
- **Seg 2 (14%, 2nd biggest segment)**: Customers with 2nd highest spending / net revenue, same vtc rate on discounts as Seg 1 but the vtc for BOGOs drops to 0.6. Probably increase discount share.
- **Seg 5 (5.1%)**: Now that's interesting. These customers react well on BOGO offers (vtc rate 0.7), but seem totally unimpressed by discounts (vtc rate 0.05). Go for BOGO then.
- **Seg 12 (8.4%)**: And here it's exactly the other way round: Discounts are ok (vtc rate 0.7), but viewed BOGOs are completed at less than 1%.

**Second group of segments** — *medium spending customers that have either been exposed to BOGO offers or to discounts only*:

- **Seg 3 (6.7%) and Seg 6 (9.9%)**: Have seen BOGO only. Seg 3 reacts ok with vtc rate of 0.7, Seg 6 considerably less with a vtc rate of 0.4.
- **Seg 4 (6%) and Seg 8 (7.2%)**: Have seen discounts only. Both Segments react ok with a vtc of 0.8 and 0.7 respectively. *(One bigger difference here is that Segment 8 was exposed to a high share of info offers.)*

**3rd group of segments** — *low spenders that could not be activated through offers of any kind*:

- **Seg 10 (8.7%)**: Quite regular customers (6 transactions a month vs. overall mean of 8), but spending small amounts only (total spending is only 25% of overall mean). They view the offers on the app regularly, but complete neither BOGOs nor discounts (vtc rate close to 0 for both).
- **Seg 11 (5.7%)**: Similar to Seg 10 but have viewed no discounts. The average transaction amount is a little higher, so maybe there would

be a chance to tickle them with some discounts. But probably they won't move to much.

- **Seg 2 (3.8%)**: Similar to 11 but have viewed no BOGO. Maybe worth a try to send them some BOGOs?

#### A Remark on Informational Offers

What about the **informational offers**? To be honest, I didn't analyze them in great detail, because as far as I can see, their effect seems limited compared to BOGOs and discounts. Here are two plots for why I think so (but, really, I didn't check this in detail and the following plots show correlations and not necessarily causations):

---

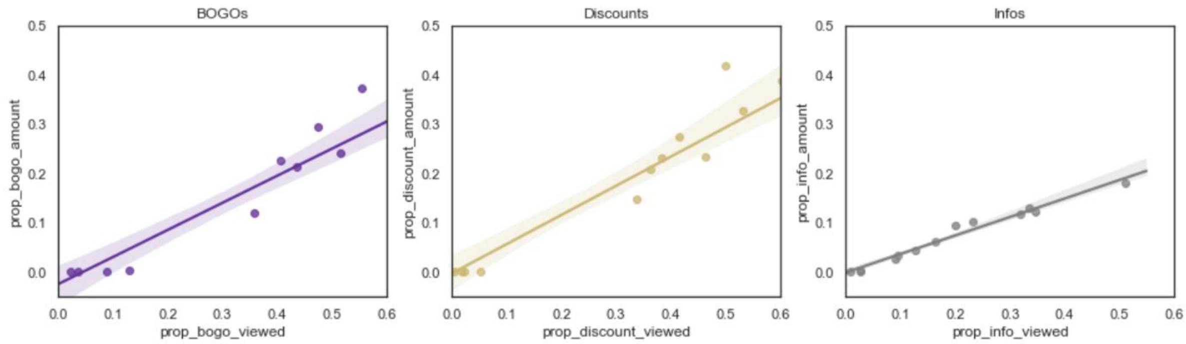



Fig 11a: The share of offers per type viewed by the 12 segments, compared to the proportion of amount spent under conditions of that offer type. Infos don't seem to trigger much purchases compared to the other types.

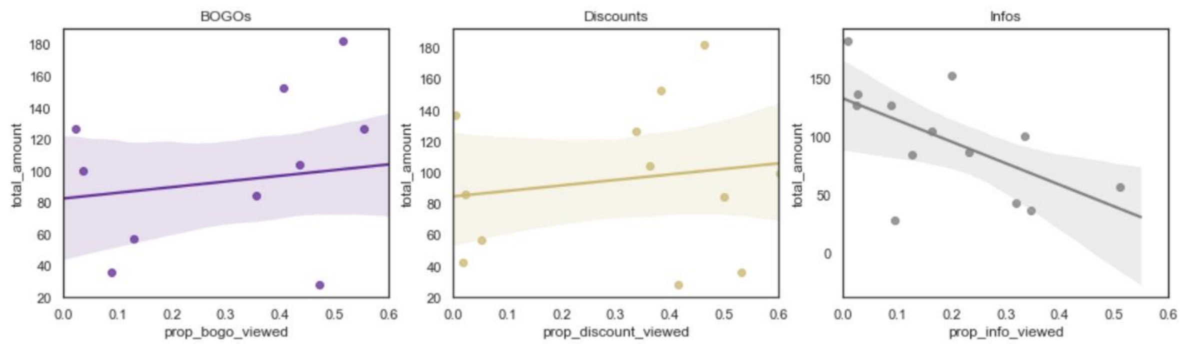


Fig 11b: What's even worse, the total mean amount spent for a segment decreases with more Infos in the mix. It's not that they don't work at all, but they perform worse than the other types.

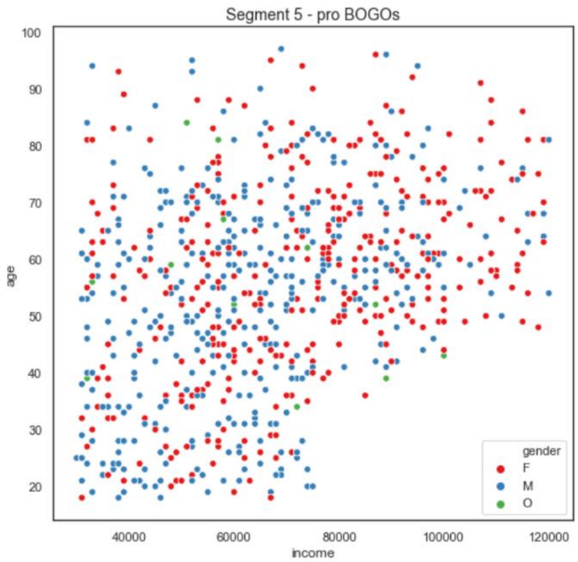
---

**5) Conclusions on Demographic Groups**

The next step includes appending the segment predictions to the demographic customer data. (Note: Remember that here are 2'175 customers for which we have no demographic info. I dropped them all. Because I also removed some outliers in the last step, only 14,167 of the original 17,000 customers are left for this part of the analysis. But that should be enough to get valid results.)

Again, the focus is on preferences of specific user groups for BOGOs vs. discounts. *In that respect the comparison of segment 5 (completed more than two thirds of viewed BOGOs, but no discounts) to segment 12 (just the opposite) seems the most promising to show a clear difference in user characteristics.*

I hoped to reveal a clear pattern doing some scatterplots with different combinations of demographic features for members of those segments — but no clear result would emerge:



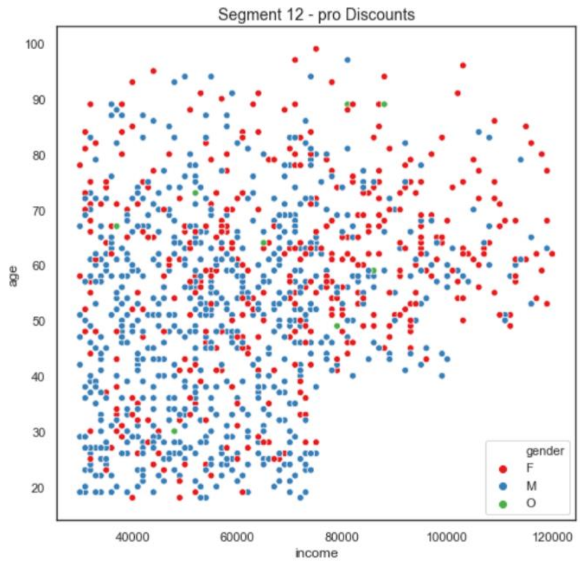



Fig 11: Hard to spot any patterns for combined demographic data between BOGO lovers and BOGO haters

But when looking at the features in isolation, a tendency becomes visible (and it is very consistent for different pairs of segment comparisons, as can be seen here if you want more details). The younger, of lower income and male customers are, the more they prefer discounts to BOGOs:

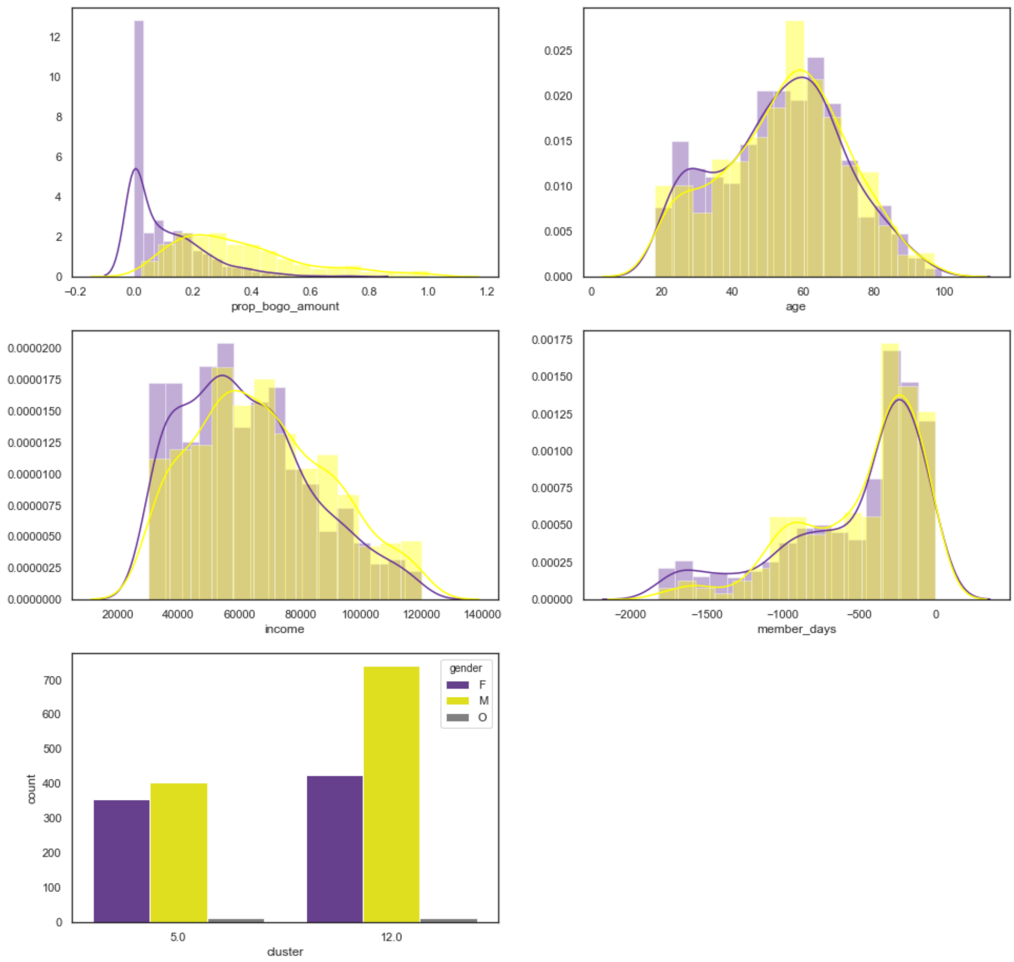



Fig 12: Segment 12 in purple vs. Segment 5 in yellow, the first Subplot shows the amount spent under BOGO conditions (within the duration of viewed BOGO offers) to make the distinction clear

Although there is a demographic difference for certain customers who absolutely prefer BOGOs to discounts and vice versa, there is also a lot of overlap between these groups. — And that leads me to the conclusions.

---

**6. Conclusions**

This dataset was a hard nut to crack. First it needed complex cleaning to extract the right signals (i.e. assigning the transactions to the corresponding offers and making sure only the viewed ones were counted ). That was the base for the construction of new features like the view-to-complete rate or the proportion of money spent on different offer types. This was followed by fine-tuned preprocessing, dimensionality reduction with PCA and finally the k-means clustering. Only then would I detect some customer groups that clearly behaved differently.

The following attempt to find distinct demographic groups was less successful. We probably could still dig a bit deeper into the demographic data and try to better single out some distinct demographic pockets of customers that react especially good on certain offer types. But the general demographic overlap is so large, that I really think it makes more sense to build segments based on purchasing behavior for the vast majority of customers.

If I had to give Starbucks some advice on improving their offer strategy, then I would definitively try to do some A/B testing on substituting a good amount of BOGOs with discount offers (there seems to be a lot of potential for that in the large Segments 1, 2 , 6 and so on (actually more or less in all segments but 5). They might also experiment with a totally new type of offer for the low spending segments 10, 11 and 2, as the existing set doesn’t work at all.

**Possible improvements / next steps**:

- *Refine the clustering.* Try other algorithms that need more initial tuning but could lead to even more sophisticated segments (DBSCAN or GMM). And check if we could go with fewer clusters (somewhere between 6 and 12) to reduce complexity when implementing the new offer strategy.
- *Analyze the effects of informational offers in more detail.* Check if there are some specific segments or pockets of demographic groups that react well enough to informational offers so they could replace the other offer types.

---

This is my capstone project for Udacity's Data Science Nanodegree. If you have any comments or questions, please let me know.