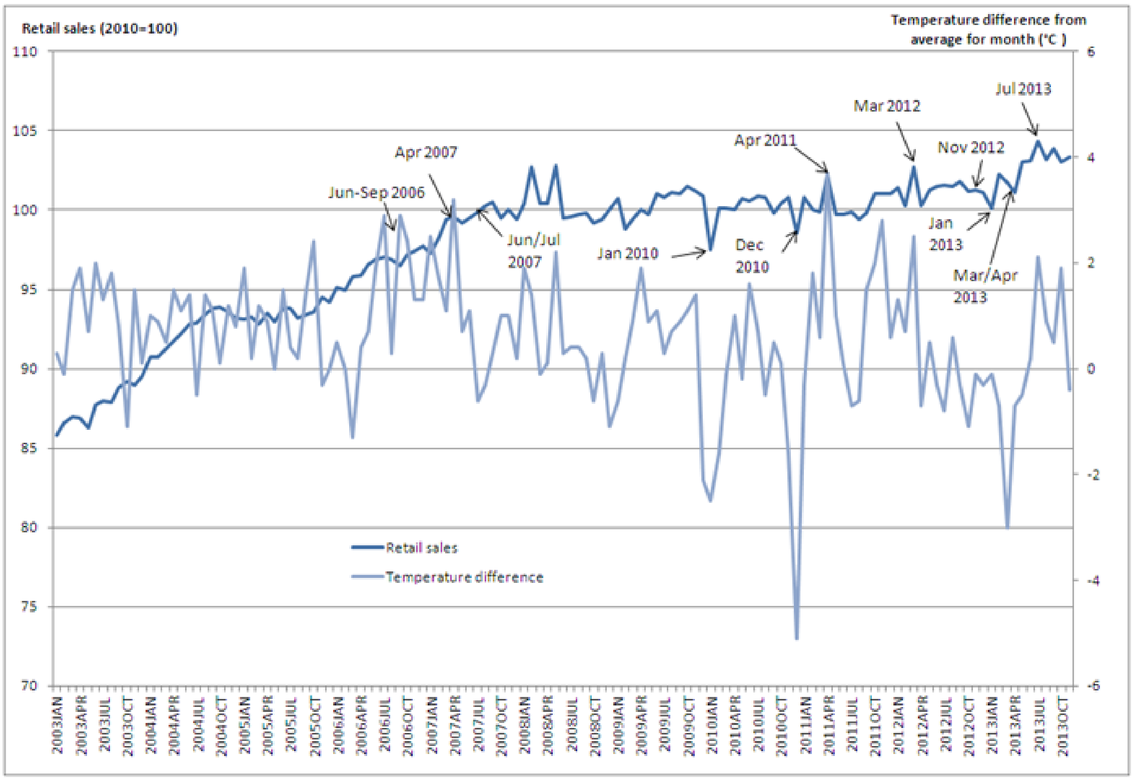
Report:

Nir Cafri

Introduction:

- product engagement increases sales - consumers pass through stages of increasing commitment (identification, enthusiasm, attention, absorption and interaction) before achieving full engagement with an offering.[1]
- War decreases buying power [2]
- Food recalls affect purchases (depends on country and recall strategy)[3]
- Weather affects retail sales: "extra sales growth (%) when the temperature is I degree Celsius higher , c.q. there is 10 hours more precipitation in that month, compared to the same month of the previous year."[4,5]



[1] Woodall, T., Rosborough, J., & Harvey, J. (2018). Proposal, project, practice, pause: Developing a framework for evaluating smart domestic product engagement. *AMS Review*, *8*(1), 58-74.
[2] https://www.investopedia.com/terms/p/purchasingpower.asp

[3] Hu, H., Djebarni, R., Zhao, X., Xiao, L., & Flynn, B. (2017). Effect of different food recall strategies on consumers' reaction to different recall norms: A comparative study. *Industrial Management & Data Systems*.
[4] https://www.weatherads.io/blog/the-impact-of-weather-on-retail-sector-in-the-uk

[5] Fris, P. (1997). Disappearing clouds: Weather influences on retail sales. In *New Operational Approaches for Financial Modelling* (pp. 195-206). Physica, Heidelberg.

CV data clustering:

Data preprocessing:
Only actions above score threshold were taken.
Score threshold was set to 2/(number of labels) in order to get significant actions only.
Actions were split into 2 groups: actions which engage with the product:

```
'hold_product': 1, 'pick_up': 2, 'put_to_cart': 3, 'touch_product': 4
```

And actions which do not engage with the product:

```
'stand': 7, 'walk': 8, 'put_back': 6
```

Stop at category was dropped from all actions due to redundancy(high probability and low informativity).
Division to groups was done due to inconsistent CV actions above score threshold.

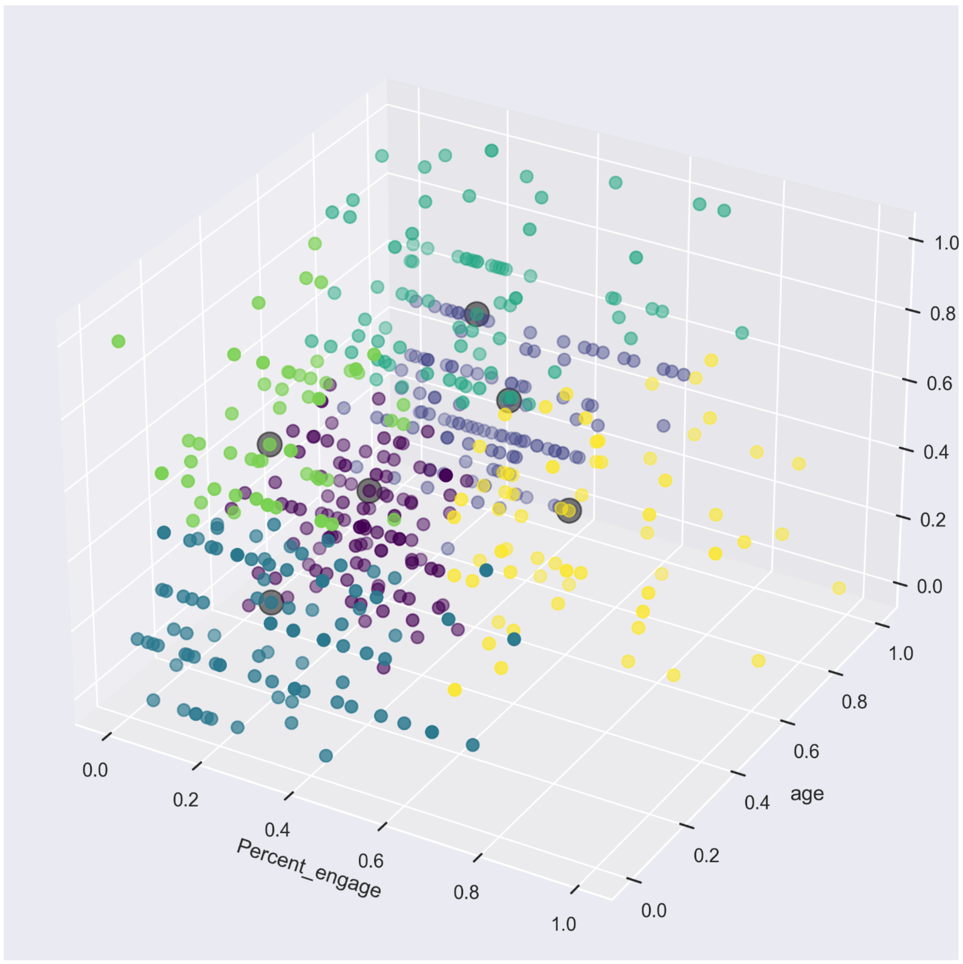
Matrices chosen are:

- Percent actions of engagement with the product out of all actions.
- Dayin week
- Age of costumer

Additional information extracted from data and used for combined model:
- Mean location (based on x,y) of actions which engage with the product
- Mean location (based on x,y) of actions which don’t engage with the product
- Date

Clustering results:

Clustering method K-mediods was used due to high modulation of the algorithm. I used built centroids initialization and multiple initialization epochs due to datas’ high heterogeneity. This method has a high probability of clustering which would be consistent.



Results show a few groups of low engagement with the product. Those groups were consistent with 4-7 clustering groups. And one group in yellow shows a high engagement percentage. These active users are especially engaged with the product and are mainly an older group who shops later in the week.

BI Modeling:
Data preprocessing:

Excels' data was converted with external features added per branch for each row to Pandas Dataframe. A run of all the excels took a few hours. Loading bar was added to monitor progress:

```
|█████████                        | ▆▃ 4/16 [25%] in 26:55 (0.0/s, eta: 59:02)
```

Pandas dataframe was saved to .csv in saved_data folder to allow faster revival of data.
All concatenation data was saved in “saved data” in a .csv file for faster load.
Labels - Pidyon column
Features matrix -

All string data was factorized to categorical integer per unique value of column.
Index and BarCode column was dropped.

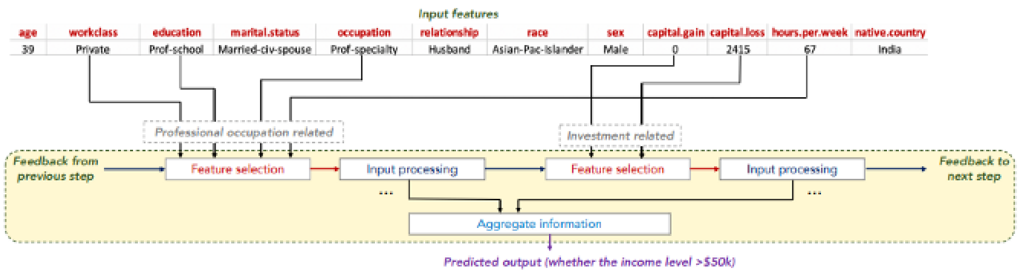
All data values were normalized: $$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Features added:

- Weather. Based on location of store. Including temperature, wind and rain. Weather was used with geopy. To get location I used meteosat. Better to use GoogleMaps but it requires payment(or at least registering with a credit card).
- Google trends - War and recall. Change in interest of the last 2 days compared to the last two weeks.

Modeling:

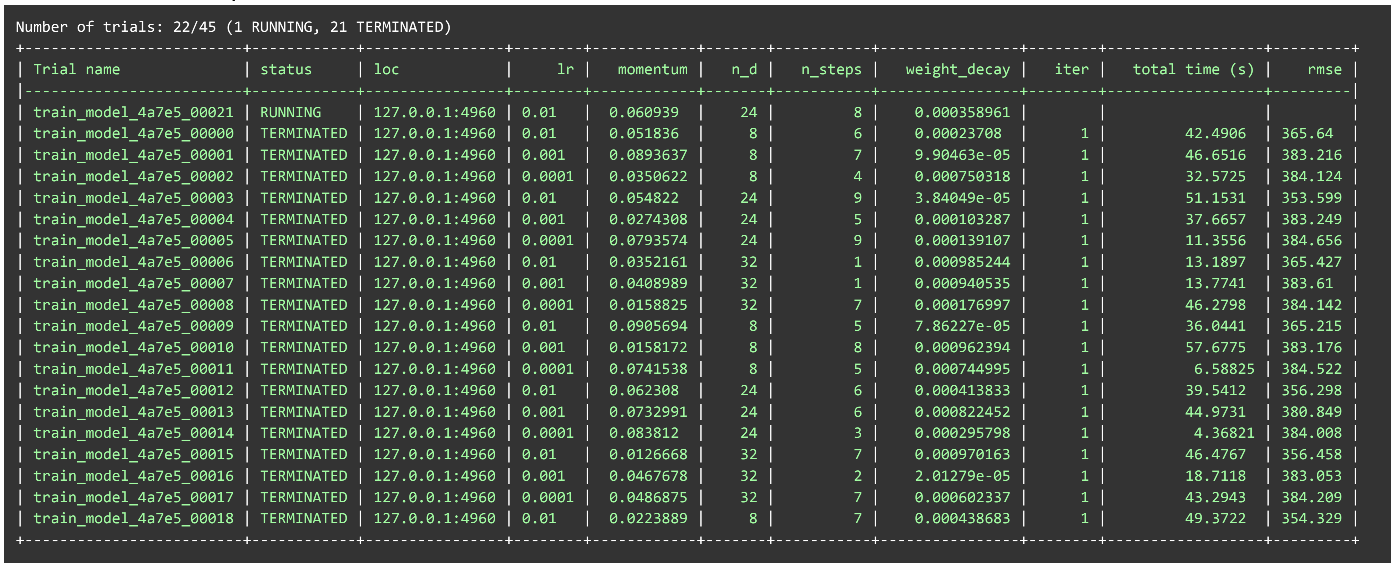
Since labeling is continuous I used regression. For tabular data I use tabnet which is based on pytorch.
Tabnet[6] uses DL to optimize short term forecasting [7]
Tabnet model:



In addition in order to avoid overfitting I used k fold cross validation (CV).
Hyperparameters were optimized with grid search & random tuning using Ray tuning:

[6] Arık, S. O., & Pfister, T. (2021). Tabnet: Attentive interpretable tabular learning. In *AAAI* (Vol. 35, pp. 6679-6687).
[7] Borghini, E., & Giannetti, C. (2021). Short Term Load Forecasting Using TabNet: A Comparative Study with Traditional State-of-the-Art Regression Models. *Engineering Proceedings*, *5*(1), 6.

Example:



```
Number of trials: 22/45 (1 RUNNING, 21 TERMINATED)
+-------------------------+------------+----------------+--------+-----------+-------+-----------+----------------+--------+------------------+---------+
| Trial name              | status     | loc            |     lr |  momentum |   n_d |   n_steps |   weight_decay |   iter |   total time (s) |    rmse |
|-------------------------+------------+----------------+--------+-----------+-------+-----------+----------------+--------+------------------+---------|
| train_model_4a7e5_00021 | RUNNING    | 127.0.0.1:4960 | 0.01   | 0.060939  |    24 |         8 |    0.000358961 |        |                  |         |
| train_model_4a7e5_00000 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.051836  |     8 |         6 |    0.00023708  |      1 |         42.4906  | 365.64  |
| train_model_4a7e5_00001 | TERMINATED | 127.0.0.1:4960 | 0.001  | 0.0893637 |     8 |         7 |    9.90463e-05 |      1 |         46.6516  | 383.216 |
| train_model_4a7e5_00002 | TERMINATED | 127.0.0.1:4960 | 0.0001 | 0.0350622 |     8 |         4 |    0.000750318 |      1 |         32.5725  | 384.124 |
| train_model_4a7e5_00003 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.054822  |    24 |         9 |    3.84049e-05 |      1 |         51.1531  | 353.599 |
| train_model_4a7e5_00004 | TERMINATED | 127.0.0.1:4960 | 0.001  | 0.0274308 |    24 |         5 |    0.000103287 |      1 |         37.6657  | 383.249 |
| train_model_4a7e5_00005 | TERMINATED | 127.0.0.1:4960 | 0.0001 | 0.0793574 |    24 |         9 |    0.000139107 |      1 |         11.3556  | 384.656 |
| train_model_4a7e5_00006 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.0352161 |    32 |         1 |    0.000985244 |      1 |         13.1897  | 365.427 |
| train_model_4a7e5_00007 | TERMINATED | 127.0.0.1:4960 | 0.001  | 0.0408989 |    32 |         1 |    0.000940535 |      1 |         13.7741  | 383.61  |
| train_model_4a7e5_00008 | TERMINATED | 127.0.0.1:4960 | 0.0001 | 0.0158825 |    32 |         7 |    0.000176997 |      1 |         46.2798  | 384.142 |
| train_model_4a7e5_00009 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.0905694 |     8 |         5 |    7.86227e-05 |      1 |         36.0441  | 365.215 |
| train_model_4a7e5_00010 | TERMINATED | 127.0.0.1:4960 | 0.001  | 0.0158172 |     8 |         8 |    0.000962394 |      1 |         57.6775  | 383.176 |
| train_model_4a7e5_00011 | TERMINATED | 127.0.0.1:4960 | 0.0001 | 0.0741538 |     8 |         5 |    0.000744995 |      1 |          6.58825 | 384.522 |
| train_model_4a7e5_00012 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.062308  |    24 |         6 |    0.000413833 |      1 |         39.5412  | 356.298 |
| train_model_4a7e5_00013 | TERMINATED | 127.0.0.1:4960 | 0.001  | 0.0732991 |    24 |         6 |    0.000822452 |      1 |         44.9731  | 380.849 |
| train_model_4a7e5_00014 | TERMINATED | 127.0.0.1:4960 | 0.0001 | 0.083812  |    24 |         3 |    0.000295798 |      1 |          4.36821 | 384.008 |
| train_model_4a7e5_00015 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.0126668 |    32 |         7 |    0.000970163 |      1 |         46.4767  | 356.458 |
| train_model_4a7e5_00016 | TERMINATED | 127.0.0.1:4960 | 0.001  | 0.0467678 |    32 |         2 |    2.01279e-05 |      1 |         18.7118  | 383.053 |
| train_model_4a7e5_00017 | TERMINATED | 127.0.0.1:4960 | 0.0001 | 0.0486875 |    32 |         7 |    0.000602337 |      1 |         43.2943  | 384.209 |
| train_model_4a7e5_00018 | TERMINATED | 127.0.0.1:4960 | 0.01   | 0.0223889 |     8 |         7 |    0.000438683 |      1 |         49.3722  | 354.329 |
+-------------------------+------------+----------------+--------+-----------+-------+-----------+----------------+--------+------------------+---------+
```

Test data can be split from data to measure accuracy at a higher prediction than validation.

Results:
Results were measure in 3 ways:

- RMSE value - the lower the more accurate the predictions were
- Ttest t value - the lower the value is the lower the difference between the teacher and

$$T = \frac{mean1 - mean2}{\frac{s(\text{diff})}{\sqrt{(n)}}}$$

  the predictions were
- Ttest p-value - the confidence level of weather the two groups are difference. The closer to 1 the more accurate the predictions were.

Results:

```
The CV score is 110.42551
RMSE ttest result: 0.006068847438167256
RMSE ttest pvalue: 0.995157953331592
```

Tuning results:

```
best config:  {'lr': 0.01, 'momentum': 0.06869814311075284, 'n_d': 8,
'weight_decay': 9.390065766228706e-05, 'n_steps': 2}
```

Combined CV & BI:

This model is similar to BI model, but uses a features matrix from the Rishon branch only and uses additional features from CV data.
The features added from CV data are:

- Percent actions of engagement with the product out of all actions.
- Dayin week
- Age of costumer
- Mean location (based on x,y) of actions which engage with the product
- Mean location (based on x,y) of actions which don't engage with the product

Date from CV data and BI data matched per row of feature matrix.

Tabnet hyperparameters were reoptimized using ray tuning for this model.

Results:

```
The CV score is 334.95236
RMSE ttest result: -0.5382598877087904
RMSE ttest pvalue: 0.5910276308855287
```

Tuning results:

```
best config:  {'lr': 0.01, 'momentum': 0.017508676653025766, 'n_d': 32,
'weight_decay': 9.73785653440169e-05, 'n_steps': 9}
```

Total runtime:

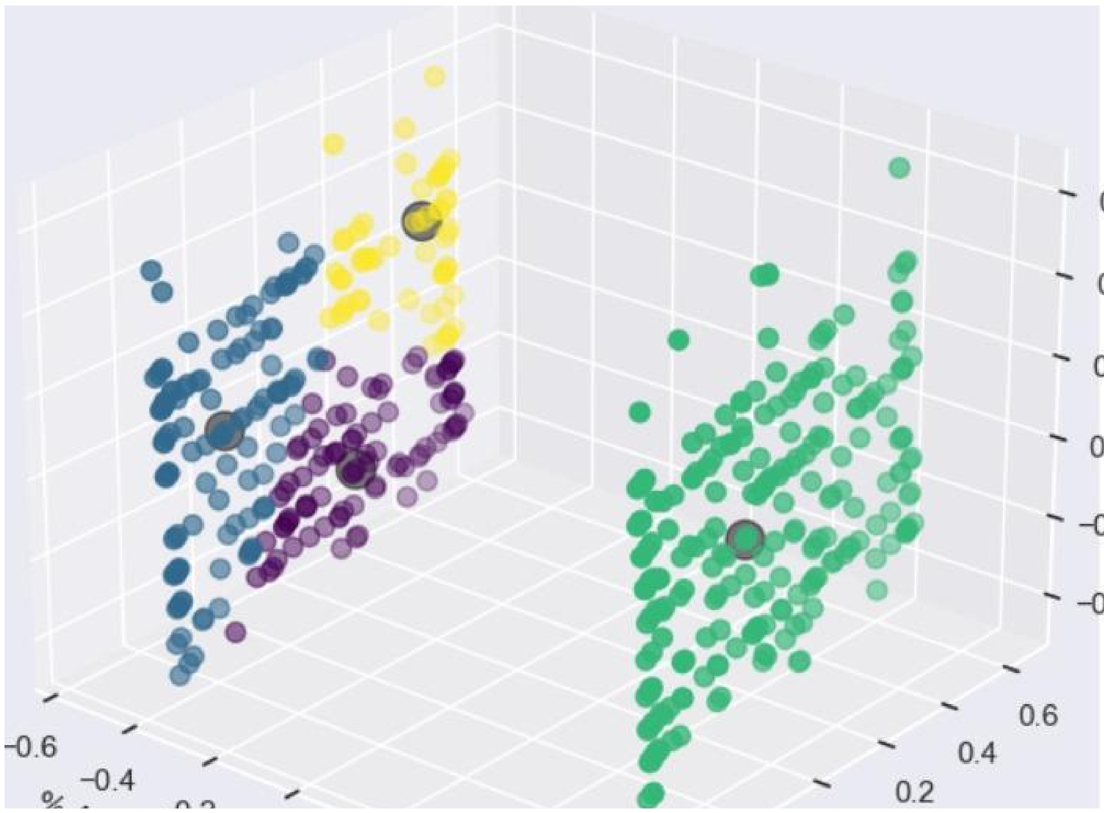
```
Runtime 4579.555155038834
```

Since the data is much smaller we see a much less accurate predicting.

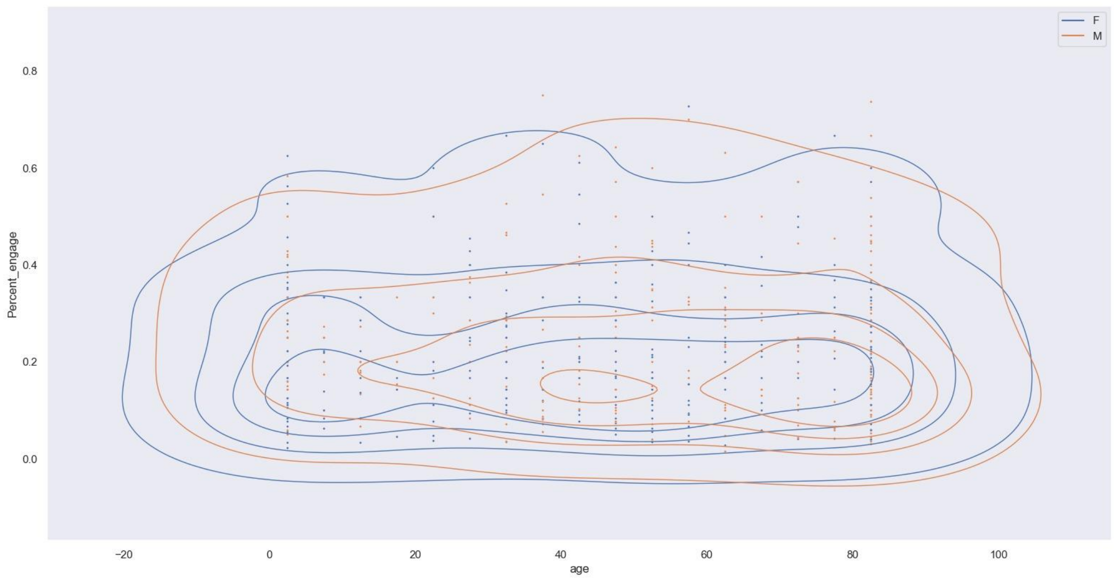
Appendix:
Epochs in tuning and running were reduced due to high run time. Multiple GPUs and bigger VRAM would reduce runtime and allow it to increase..

CV data:
If data was a continuous procedure of shopping with consistent scoring per item purchase. Ideally I would use the “actions_process” function. This function returns the actions the customer did until they put the item in the cart. By looking at the number of actions the consumer did until they chose the item the model would be able to detect behavior that leads to item purchase. Binary returns an array of each costumer how many engagements with products they did before put to cart. While multiclass returns a list of all the actions done before put to cart.
Other clustering methods that were considered:

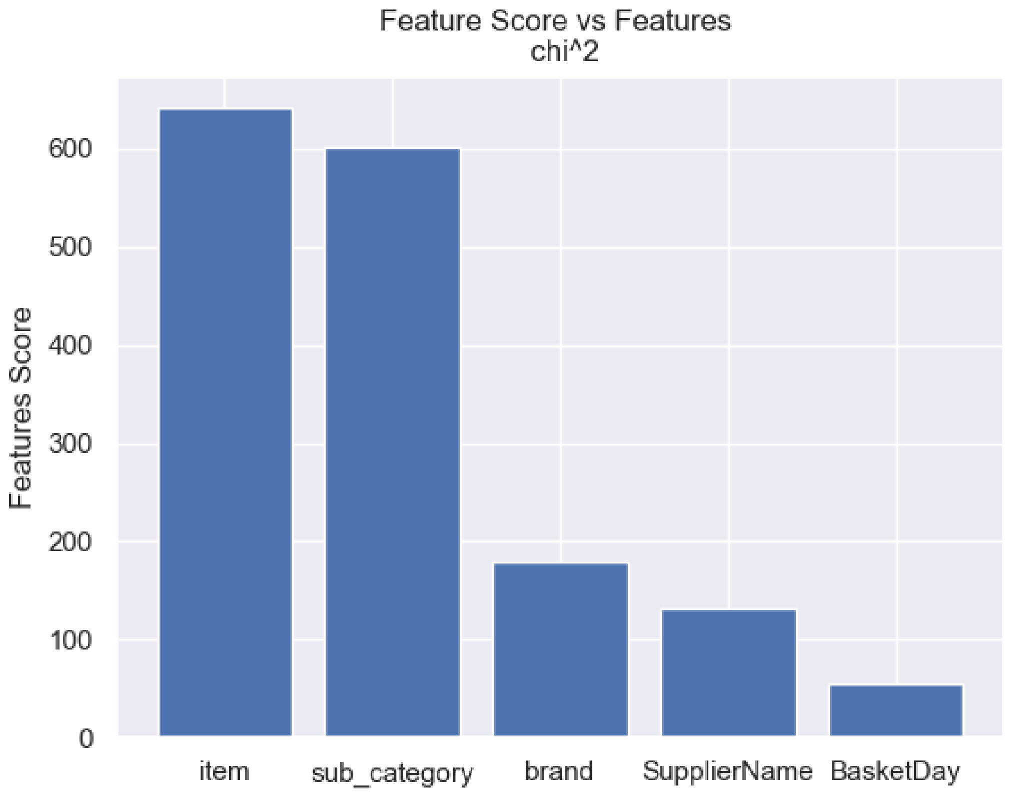
- PCA - in order to use all CV data and reduce dimension. This was not chosen due to difficulty to explain results.



- No clustering visualization - we can see that there is age and gender difference in engagement.



- Highest feature correlation to Pidyun:



BI features:

- I started from looking at two weeks before and after.

```
timeframe=(date-datetime.timedelta(days=delta_days)).strftime("%Y-%m-%d") + ' '+
((date+datetime.timedelta(days=delta_days)).strftime("%Y-%m-%d"))
```

Afterwards I realized the future cannot affect people’s shopping habits in the present so I only looked at 14 days in the past. The variable chosen is the mean slope of google trends of a word 3 days prior to the date.

- Other external features ideas: Traffic, special event (e.g sports, terror ect.), construction around the branch or in it.

General:

To save run time only "משומר בשר/דגים" category was run. To change to all data delete line:

```
df = df[df.category == category_get_only]  # Get only specific category
```

- the candidate needs to *choice* - needs to be corrected to choose*.
- BI data from sales (15 excel files in Drive each represents a *signle* day). - needs to be corrected to single*
- Using database (SQL/NoSQL) is a bunos -needs to be corrected to bonus*
- Excels are long and big and take a long time to run - recommended to work with .csv