Report: Clustering

1.Process of making features description

```
#Examples of my features

median_price_size = df.groupby('dbi_item_size')['post_discount_price'].median().reset_i
df = df.merge(median_price_size, on='dbi_item_size', suffixes=("", 'median_price_per_si

median_price_person = df.groupby('dd_card_number')['post_discount_price'].median()
df = df.merge(median_price_person, on='dd_card_number', suffixes=("", 'median_price_per

median_price_store = df.groupby('store_number')['post_discount_price'].median()
df = df.merge(median_price_store, on='store_number', suffixes=("", 'median_price_per_st

median_price_store_on_sum = df.groupby('store_number')['post_discount_price'].median()
df = df.merge(median_price_store_on_sum, on='store_number', suffixes=("", 'median_price

median_price_person_on_sum = df.groupby('dd_card_number')['post_discount_price'].median
df = df.merge(median_price_person_on_sum, on='dd_card_number', suffixes=("", 'median_pr

median_price_person_per_day = df.groupby(['dd_card_number','date'])['post_discount_pric
df = df.merge(median_price_person_per_day, on=['dd_card_number','date'], suffixes=("",
```

Features description:

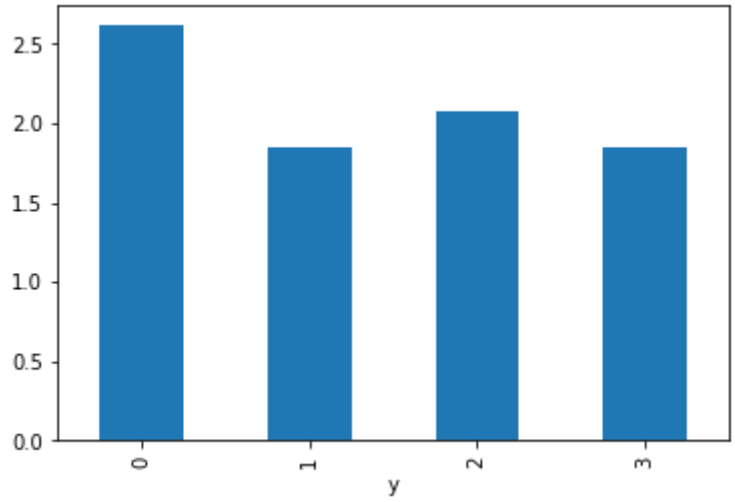
- median_price_per_size: Median product price in specific dbi_item_size category
- median_price_per_person: Median product price peoples buy
- median_price_per_store: Median product price in specific store
- median_price_store_on_sum: Median product price in specific store divided on total sum on store
- median_price_person_on_sum: Median product price peoples buy divided by total sum on person
- median_price_person_per_day: Median product price people buy per day

```
#We have 4 clusters with numbers of element:
df.groupby('y')['index'].count()
```

```
y
0      855
1    19990
2       41
3    31050
Name: index, dtype: int64
```

```
#People from 1 cluster spend more money in stores and has higher median price per trans
#People from 2 cluster spends less money in stores and per single transactions
df.groupby('y').mean()['post_discount_pricemedian_price_per_store'].plot.bar()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd642964d30>
```



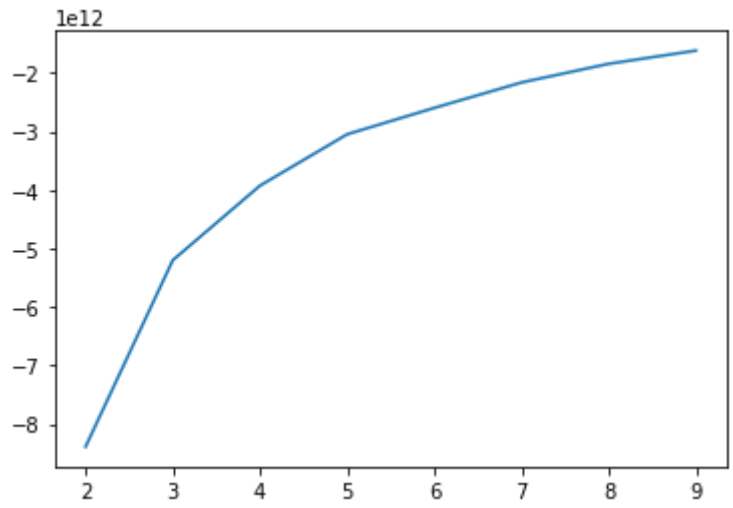
Basically, seems that clusters has this interpretation:

- 0 cluster: people buying only drinks
- 1 cluster: people buying only food
- 2 cluster: people buying mostly food + some drinks
- 3 cluster: people buying mostly food + some drinks

Mostly on morning

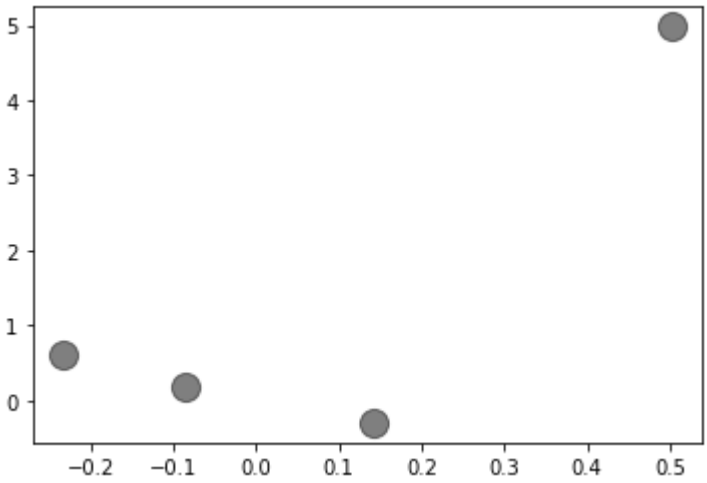
I have chosen 5 clusters based on the Kmeans score. It can be seen that this score tends to rise more slide after 5 clusters. It shows that 5 clusters could be optimal for this task

```
ns = list(range(2, 10, 1))
df_norm = StandardScaler().fit_transform(df)
plt.plot(ns, [KMeans(n_clusters=n).fit(df).score(df) for n in ns]);
```



Here also the representation of the cluster's center's distribution in the 2D graph

```
centers = model.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5);
```


5
4
3
2
1
0
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4
0.5