Customer and product segmentation basics

MACHINE LEARNING FOR MARKETING IN PYTHON

Karolis Urbonas
Head of Analytics & Science, Amazon

datacamp

Data format

```
# Customer by product/service matrix
wholesale.head()
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	12669	9656	7561	214	2674	1338
1	7057	9810	9568	1762	3293	1776
2	6353	8808	7684	2405	3516	7844
3	13265	1196	4221	6404	507	1788
4	22615	5410	7198	3915	1777	5185

Unsupervised learning models

- Hierarchical clustering
- K-means
- Non-negative matrix factorization (NMF)
- Biclustering
- Gaussian mixture models (GMM)
- And many more

Unsupervised learning models

- Hierarchical clustering
- **K-means**
- **Non-negative matrix factorization (NMF)**
- Biclustering
- Gaussian mixture models (GMM)
- And many more

Unsupervised learning steps

1. **Initialize** the model
2. **Fit** the model
3. **Assign** cluster values
4. **Explore** results

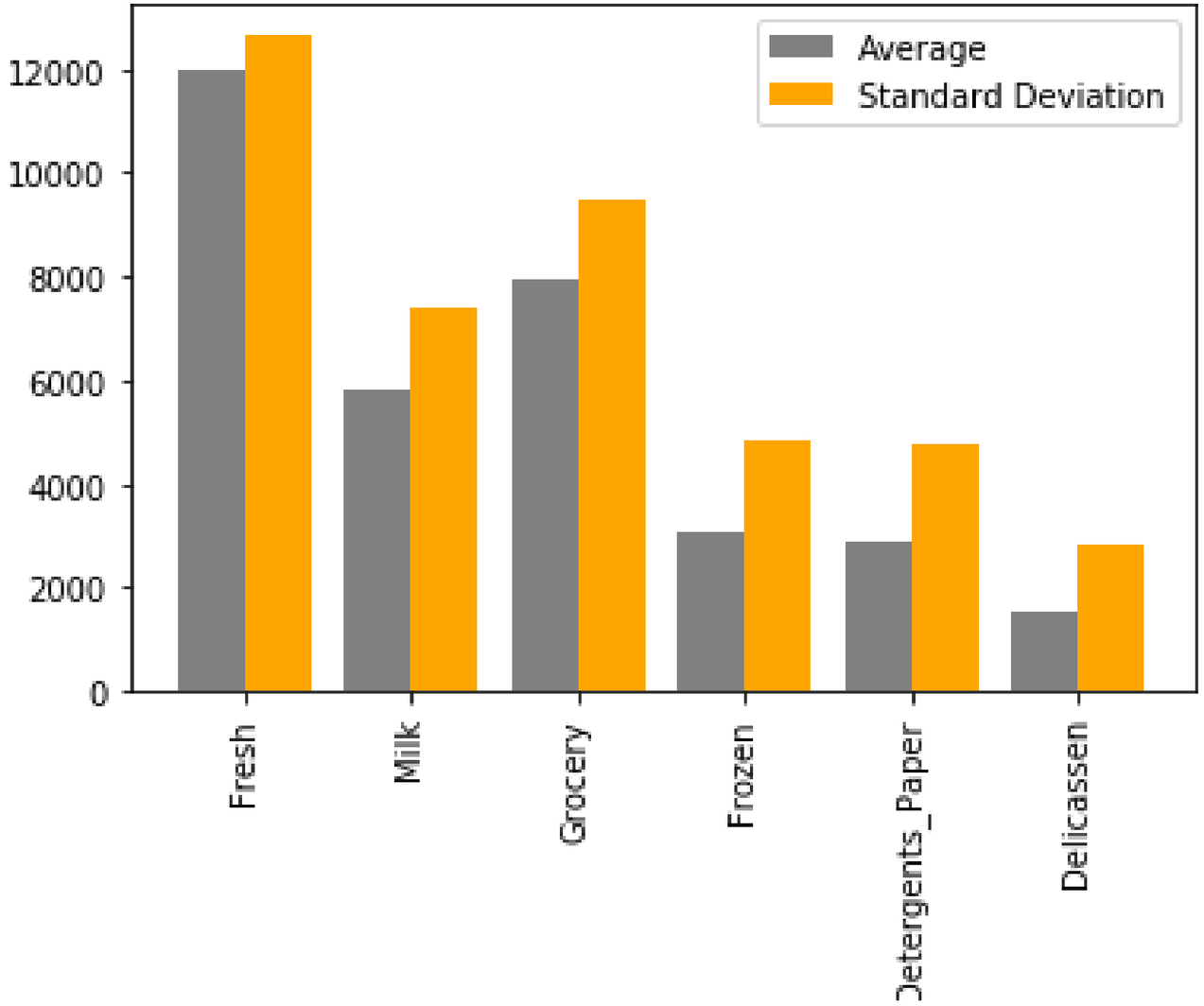
Explore variables

```
wholesale.agg(['mean','std']).round(0)
```

```
        Fresh    Milk  Grocery  Frozen  Detergents_Paper  Delicassen
mean  12000.0  5796.0   7951.0  3072.0            2881.0      1525.0
std   12647.0  7380.0   9503.0  4855.0            4768.0      2820.0
```

```
# Get the statistics
averages = wholesale.mean()
st_dev = wholesale.std()
x_names = wholesale.columns
x_ix = np.arange(wholesale.shape[1])
# Plot the data
import matplotlib.pyplot as plt
plt.bar(x_ix-0.2, averages, color='grey', label='Average', width=0.4)
plt.bar(x_ix+0.2, st_dev, color='orange', label='Standard Deviation', width=0.4)
plt.xticks(x_ix, x_names, rotation=90)
plt.legend()
plt.show()
```

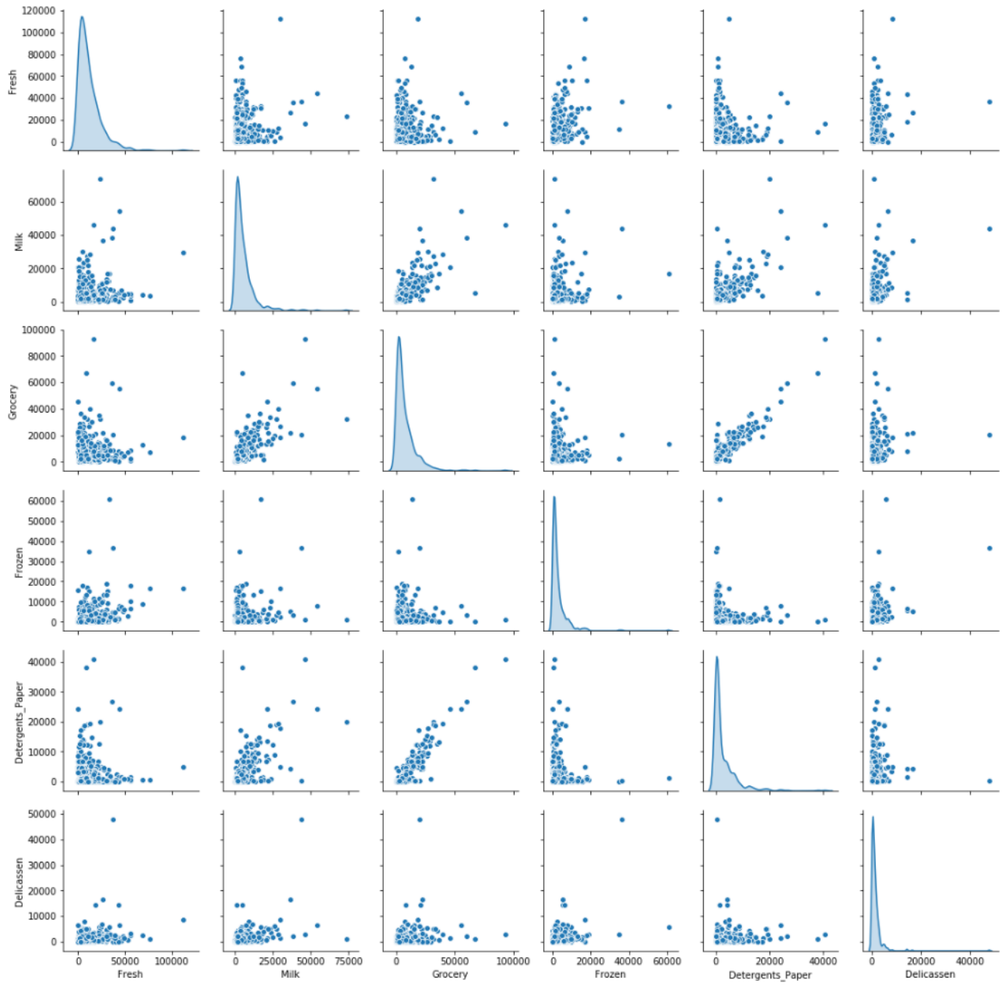
Bar chart of averages and standard deviations



Visualize pairwise plot to explore distributions

```
import seaborn as sns
sns.pairplot(wholesale, diag_kind='kde')
plt.show()
```

Pairwise plot review



Let's practice!

MACHINE LEARNING FOR MARKETING IN PYTHON

Data preparation for segmentation

MACHINE LEARNING FOR MARKETING IN PYTHON

Karolis Urbonas
Head of Analytics & Science, Amazon

datacamp

Model assumptions

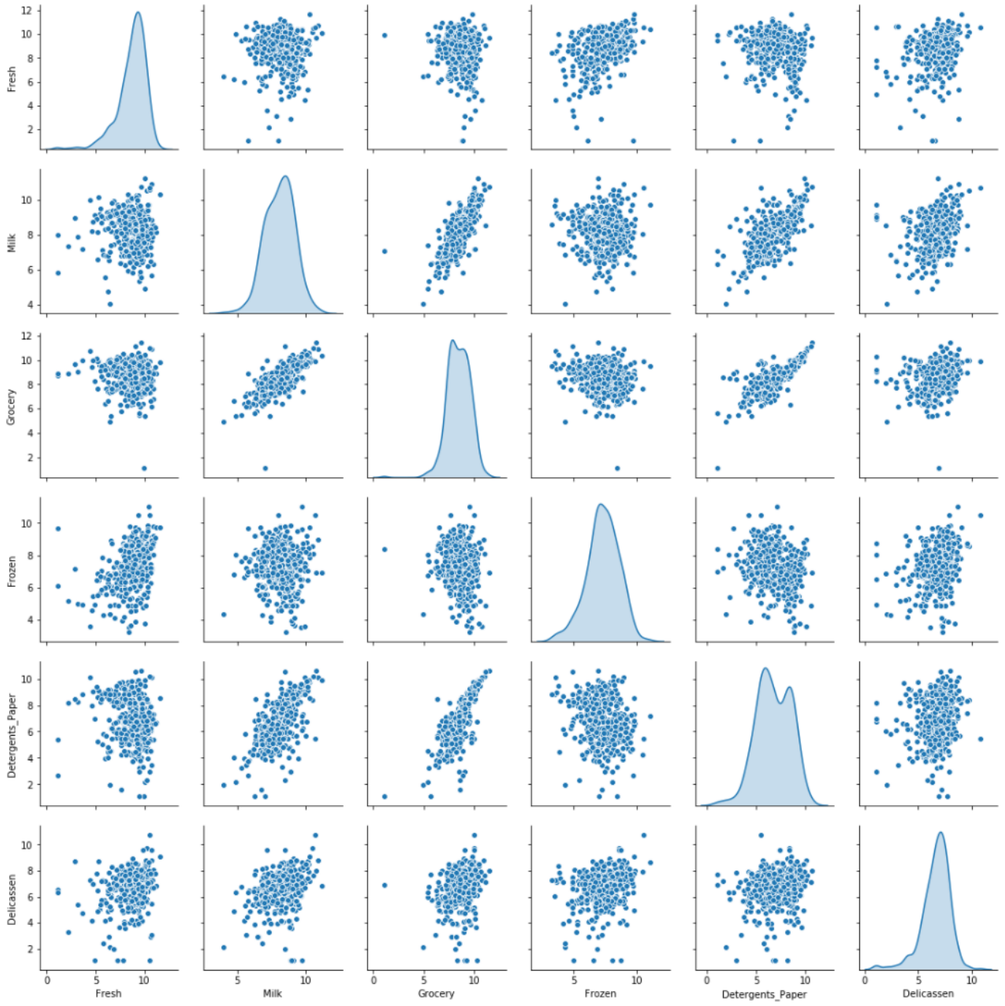
- First we'll start with K-means
- K-means clustering works well when data is 1) ~normally distributed (no skew), and 2) standardized (mean = 0, standard deviation = 1)
- Second model - NMF - can be used on raw data, especially if the matrix is sparse

Unskewing data with log-transformation

```
# First option - log transformation
wholesale_log = np.log(wholesale)
```

```
sns.pairplot(wholesale_log, diag_kind='kde')
plt.show()
```

Explore log-transformed data



Unskewing data with Box-Cox transformation

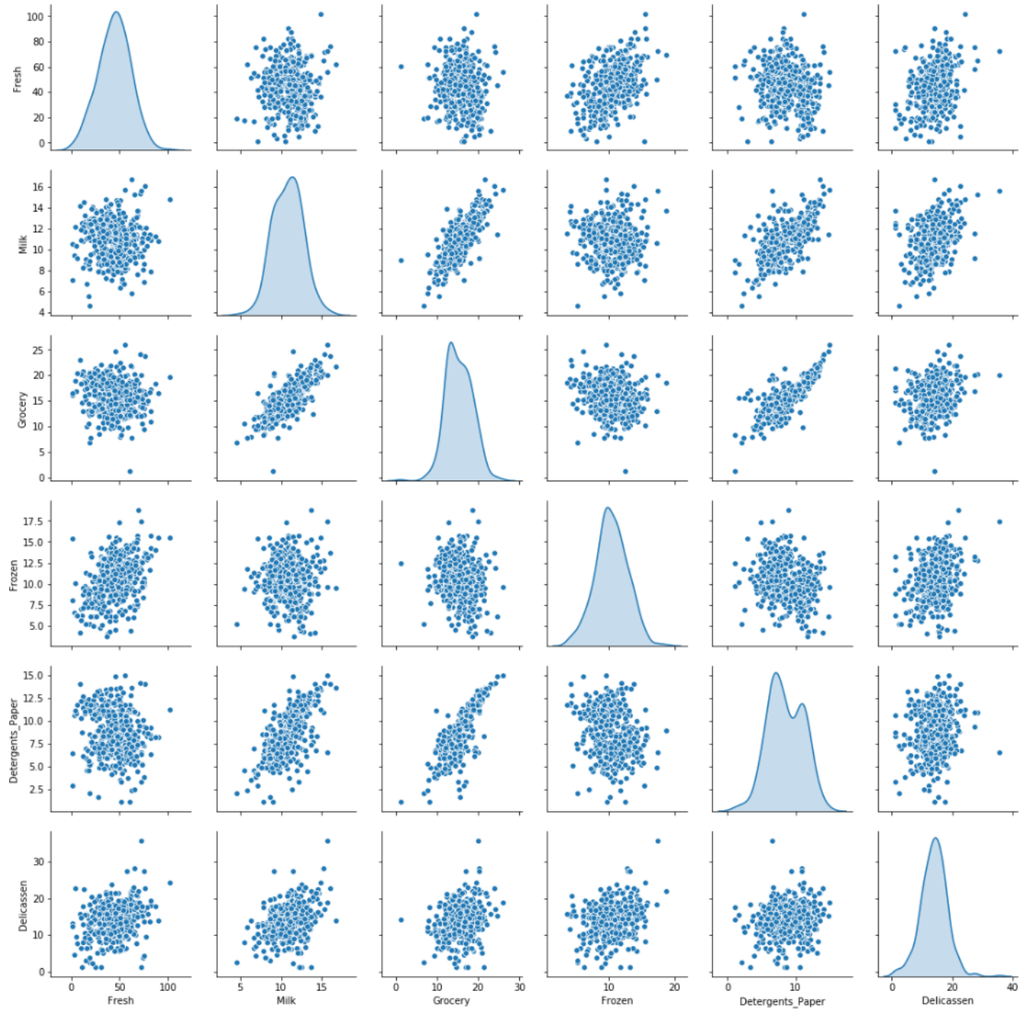
```
# Second option - Box-Cox transformation
from scipy import stats

def boxcox_df(x):
    x_boxcox, _ = stats.boxcox(x)
    return x_boxcox

wholesale_boxcox = wholesale.apply(boxcox_df, axis=0)
```

```
sns.pairplot(wholesale_boxcox, diag_kind='kde')
plt.show()
```

Explore Box-Cox transformed data



Scale the data

- Subtract column average from each column value
- Divide each column value by column standard deviation
- Will use `StandardScaler()` module from `sklearn`

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(wholesale_boxcox)
wholesale_scaled = scaler.transform(wholesale_boxcox)
wholesale_scaled_df = pd.DataFrame(data=wholesale_scaled,
                                   index=wholesale_boxcox.index,
                                   columns=wholesale_boxcox.columns)
wholesale_scaled_df.agg(['mean','std']).round()
```

```
      Fresh  Milk  Grocery  Frozen  Detergents_Paper  Delicassen
mean   -0.0   0.0      0.0     0.0              -0.0         0.0
std     1.0   1.0      1.0     1.0               1.0         1.0
```

Let's practice!

MACHINE LEARNING FOR MARKETING IN PYTHON

Build customer and product segmentation

MACHINE LEARNING FOR MARKETING IN PYTHON

Karolis Urbonas
Head of Analytics & Science, Amazon

datacamp

Segmentation steps with K-means

Segmentation with K-means (for `k` number of clusters):

```
from sklearn.cluster import KMeans
kmeans=KMeans(n_clusters=k)
kmeans.fit(wholesale_scaled_df)
wholesale_kmeans4 = wholesale.assign(segment = kmeans.labels_)
```

Segmentation steps with NMF

Segmentation with NMF (`k` number of clusters):

```
from sklearn.decomposition import NMF
nmf = NMF(k)
nmf.fit(wholesale)
components = pd.DataFrame(nmf.components_, columns=wholesale.columns)
```

Extracting segment assignment:

```
segment_weights = pd.DataFrame(nmf.transform(wholesale, columns=components.index)
segment_weights.index = wholesale.index
wholesale_nmf = wholesale.assign(segment = segment_weights.idxmax(axis=1))
```

How to initialize the number of segments?

- Both K-means and NMF require to set a number of clusters (`k`)
- Two ways to define `k` : 1) Mathematically, 2) Test & learn
- We'll explore mathematical elbow criterion method to get a ball-park estimate

Elbow criterion method

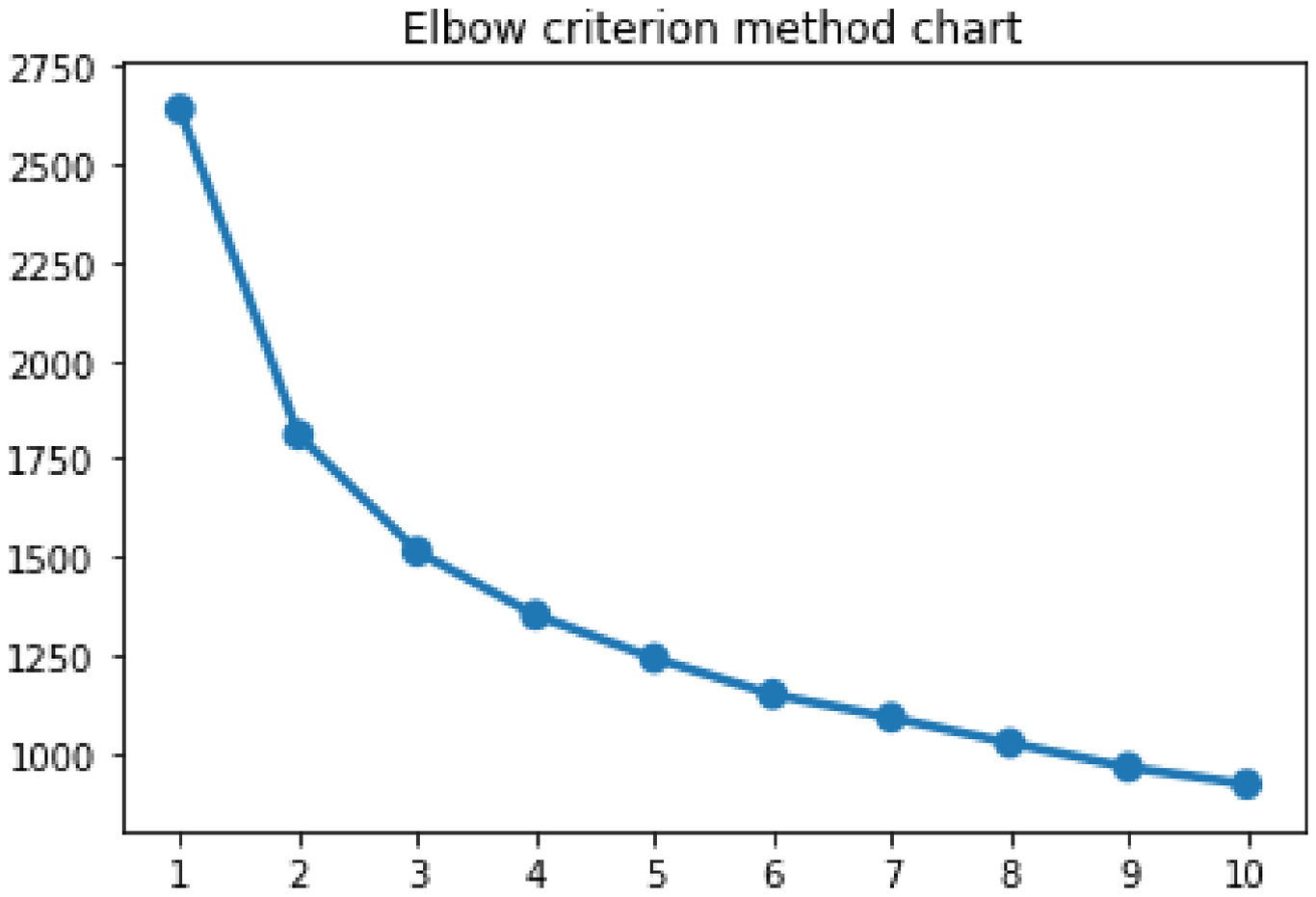
- Iterate through a number of `k` values
- Run clustering for each on the same data
- Calculate sum of squared errors (`SSE`) for each
- Plot `SSE` against `k` and identify the "elbow" - diminishing incremental improvements in error reduction

Calculate sum of squared errors and plot the results

```
sse = {}
for k in range(1, 11):
    kmeans=KMeans(n_clusters=k, random_state=333)
    kmeans.fit(wholesale_scaled_df)
    sse[k] = kmeans.inertia_
```

```
plt.title('Elbow criterion method chart')
sns.pointplot(x=list(sse.keys()), y=list(sse.values()))
plt.show()
```

Identifying the optimal number of segments



Test & learn method

- First, calculate mathematically optimal number of segments
- Build segmentation with multiple values around the optimal `k` value
- Explore the results and choose one with most business relevance (Can you name the segments? Are they ambiguous / overlapping?)

Let's build customer segments!

MACHINE LEARNING FOR MARKETING IN PYTHON

Visualize and interpret segmentation solutions

MACHINE LEARNING FOR MARKETING IN PYTHON

Karolis Urbonas
Head of Analytics & Science, Amazon

datacamp

Methods to explore segments

- Calculate average / median / other percentile values for each variable by segment
- Calculate relative importance for each variable by segment
- We can explore the data table or plot it (heatmap is a good choice)

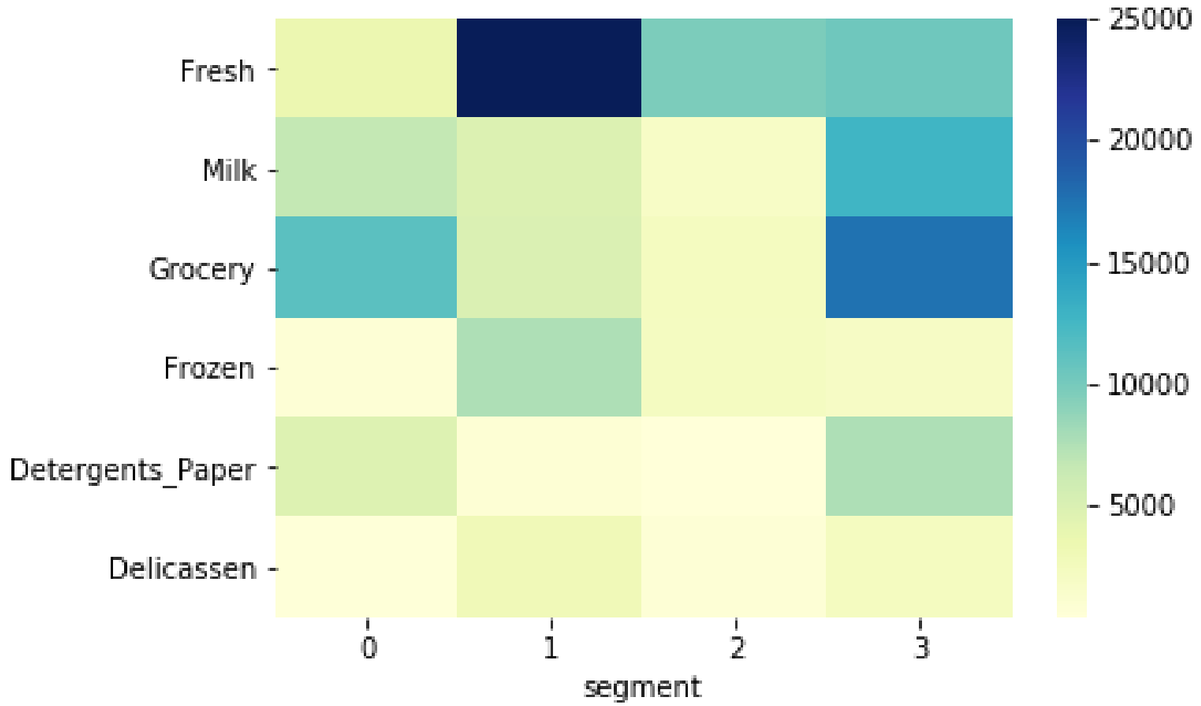
Analyze average K-means segmentation attributes

```
kmeans4_averages = wholesale_kmeans4.groupby(['segment']).mean().round(0)
print(kmeans4_averages)
```

cluster	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	3654.0	6654.0	11413.0	707.0	4731.0	573.0
1	24969.0	4852.0	5024.0	7690.0	844.0	2918.0
2	9766.0	1911.0	2403.0	2312.0	407.0	712.0
3	10452.0	12771.0	17591.0	1968.0	7622.0	2403.0

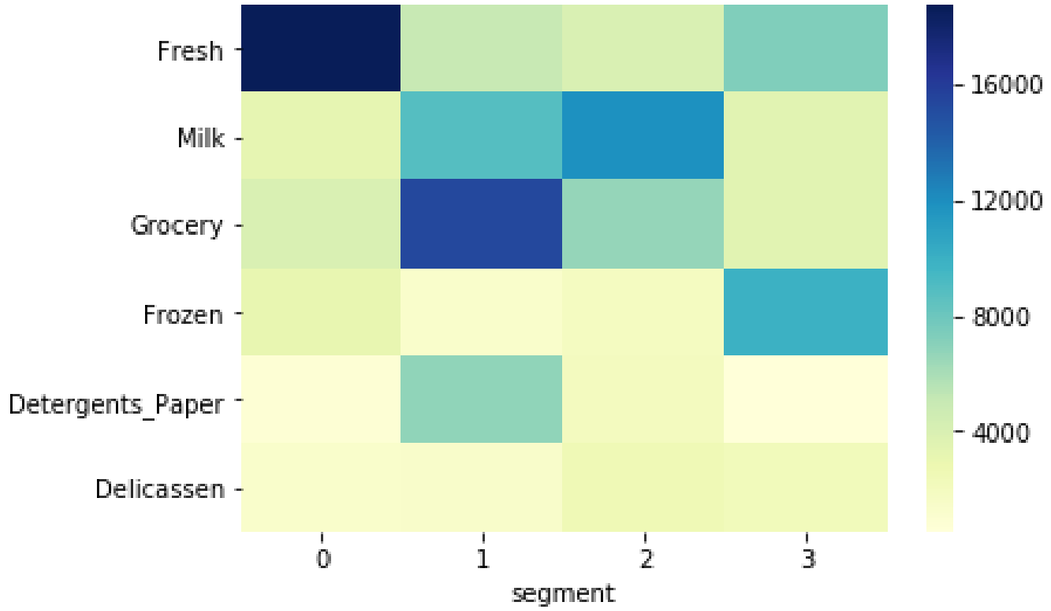
Plot average K-means segmentation attributes

```
sns.heatmap(kmeans4_averages.T, cmap='YlGnBu')
plt.show()
```



Plot average NMF segmentation attributes

```
nmf4_averages = wholesale_nmf4.groupby('segment').mean().round(0)
sns.heatmap(nmf4_averages.T, cmap='YlGnBu')
plt.show()
```



Let's build 3-segment solutions!

MACHINE LEARNING FOR MARKETING IN PYTHON

Congratulations!

MACHINE LEARNING FOR MARKETING IN PYTHON

Karolis Urbonas
Head of Analytics & Science, Amazon

What have we learned?

- Different types of machine learning - supervised, unsupervised, reinforcement
- Machine learning steps
- Data preparation techniques for different kinds of models
- Predict telecom customer churn with logistic regression and decision trees
- Calculate customer lifetime value
- Predict next month transactions with linear regression
- Measure model performance with multiple metrics
- Segment customers based on their product purchase history with K-means and NMF

What's next?

- Dive deeper into each topic
- Explore the datasets, change the parameters and try to improve model accuracy, or segmentation interpretability
- Take on a project with other dataset, and build models with comments by yourself
- Write a blog post with link to GitHub code once you finish your project
- Test your knowledge in your job

Thank you and great learning!

MACHINE LEARNING FOR MARKETING IN PYTHON