Creating Customer Segments

In this project you, will analyze a dataset containing annual spending amounts for internal structure, to understand the variation in the different types of customers that a wholesale distributor interacts with.

Instructions:

- Run each code block below by pressing **Shift+Enter**, making sure to implement any steps marked with a TODO.
- Answer each question in the space provided by editing the blocks labeled "Answer:".
- When you are done, submit the completed notebook (.ipynb) with all code blocks executed, as well as a .pdf version (File > Download as).

In [2]:

```
# Import libraries: NumPy, pandas, matplotlib
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Tell iPython to include plots inline in the notebook
%matplotlib inline

# Read dataset
data = pd.read_csv("wholesale-customers.csv")
print("Dataset has {} rows, {} columns".format(*data.shape))
print(data.head(5))  # print the first 5 rows
```

```
Dataset has 440 rows, 6 columns
   Fresh  Milk  Grocery  Frozen  Detergents_Paper  Delicatessen
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
```

##Feature Transformation

1) In this section you will be using PCA and ICA to start to understand the structure of the data. Before doing any computations, what do you think will show up in your computations? List one or two ideas for what might show up as the first PCA dimensions, or what type of vectors will show up as ICA dimensions.

Answer: **The first PCA dimension is the dimension in the data with highest variance. Intuitively, it corresponds to the 'longest' vector one can find in the 6-dimensional feature space that captures the data, that is, the eigenvector with the largest eigenvalue.**

I suppose the first component will carry a high load of the 'Fresh' feature, as this feature seems to vary more than any of the other features (according to the README.md-file, 'Fresh' has the highest variance). Moreover, this feature seems to vary independently of the others, that is, a high or low value of 'Fresh' is not very informative for the values of the other features. A pratical interpretation of these observations could be that some of the supplied customers focus on fresh items, whereas other focus on non-fresh items.

ICA, as opposed to PCA, finds the subcomponents that are statistically independent. I suppose ICA also finds 'Fresh' as one of the first components. The other components, however, may differ, as they need not be orthogonal in the feature space (in contrast to PCA).

###PCA

In [3]:

```
# (DONE) TODO: Apply PCA with the same number of dimensions as variables in th
from sklearn.decomposition import PCA
pca = PCA(n_components=6) # 6 components for 6 variables
pca.fit(data)

# Print the components and the amount of variance in the data contained in eac
print(pca.components_)
print(pca.explained_variance_ratio_)
```

```
[[-0.97653685 -0.12118407 -0.06154039 -0.15236462  0.00705417
-0.06810471]
 [-0.11061386  0.51580216  0.76460638 -0.01872345  0.36535076
0.05707921]
 [-0.17855726  0.50988675 -0.27578088  0.71420037 -0.20440987
0.28321747]
 [-0.04187648 -0.64564047  0.37546049  0.64629232  0.14938013
-0.02039579]
 [ 0.015986    0.20323566 -0.1602915   0.22018612  0.20793016
-0.91707659]
 [-0.01576316  0.03349187  0.41093894 -0.01328898 -0.87128428
-0.26541687]]
[ 0.45961362  0.40517227  0.07003008  0.04402344  0.01502212  0.0
0613848]
```

2) How quickly does the variance drop off by dimension? If you were to use PCA on this dataset, how many dimensions would you choose for your analysis? Why?

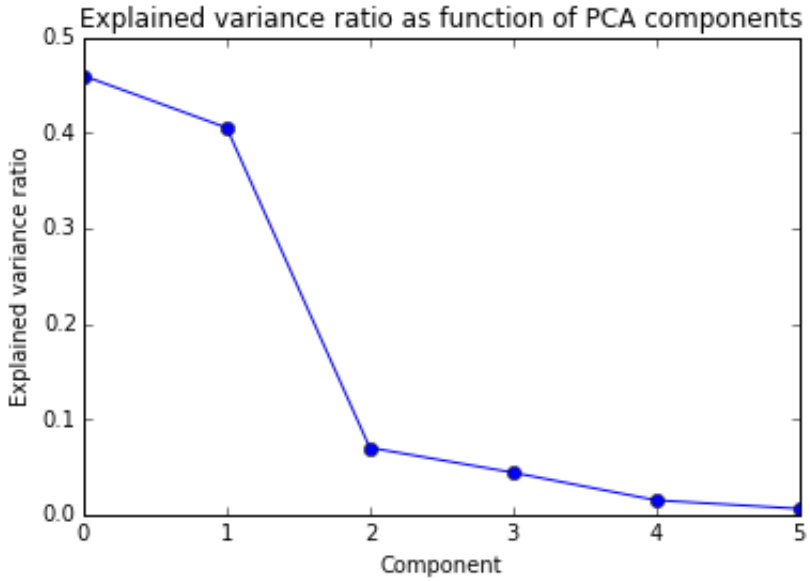
Answer: **The explained variance is high for the first two dimensions (45.96 % and 40.52 %, respectively), but drops significantly beginning with the third dimension (7.00 % for the third, 4.40 % for the fourth dimension). Thus, the first two components explain already 86.5 % of the variation in the data.**

How many dimension to choose for the analysis really depends on the goal of the analysis. Even though PCA reduces the feature space (with all advantages that brings, such as faster computations) and makes interpreting the data easier for us by projecting them down to a lower dimension, it necessarily comes with a loss of information that may or may not be desired.

It the case at hand, assuming interpretation is the goal (creating customer segments) and given the sharp drop of the explained variance after the second component, I would choose the first two dimensions for analysis.

In [4]:

```
plt.plot(list(pca.explained_variance_ratio_),'-o')
plt.title('Explained variance ratio as function of PCA components')
plt.ylabel('Explained variance ratio')
plt.xlabel('Component')
plt.show()
```



3) What do the dimensions seem to represent? How can you use this information?

Answer: **The first dimension seems to basically represent only the 'fresh'-feature, as this feature has a strong negative projection on the first dimension. The other features have rather weak (mostly negative) projections on the first dimension. That is, the first dimension basically tells us whether the 'fresh'-feature value is high or low, mixed with a little bit of information from the other features.**

The second dimension is mainly represented by the features 'Grocery', 'Milk' and 'Detergents', in the order of decreasing importance, and has rather low correlation with the other features.

There are two main uses of this information. The first use is feature interpretation and hypothesis formation. We could form initial conjectures about the customer segments contained in the data. One conjecture could be that the bulk of customers can be split into customers ordering mainly 'fresh' items and customers mainly ordering 'Grocery', 'Milk' and 'Detergents' from the wholesale distributor. The second use is that, given knowledge of the PCA components, new features can be engineered for further analysis of the problem. These features could be generated by applying an exact PCA-transformation or by using some heuristic based on the feature combinations recovered in PCA.

###ICA

In [124]:

```
# (DONE) TODO: Fit an ICA model to the data
# Note: Adjust the data to have center at the origin first!
def center_data(data, rescale = 0):
    centeredData = data.copy()
    for col in centeredData.columns:
        centeredData[col] = (centeredData[col] - np.mean(centeredData[col]))/
    return centeredData

from sklearn.decomposition import FastICA
#data_centered = center_data(data)

ica = FastICA(n_components=6, whiten=True)
ica.fit(center_data(data,0))

# Print the independent components
print(ica.components_)

# Print the independent components (rescaled again)
print('Independent components scaled with mean')
print(np.multiply(ica.components_,list(np.mean(data))))
```

```
[[ -3.97580684e-06   8.56819919e-07   6.20011762e-07   6.78097508
e-07
   -2.04990965e-06   1.04660364e-06]
 [  1.51685332e-07   9.85644833e-06  -5.78069673e-06  -3.69646977
e-07
    3.22189924e-06  -6.07175115e-06]
 [  3.00627916e-07  -2.26192941e-06  -1.21219680e-05   1.45884725
e-06
    2.82232539e-05   5.72200539e-06]
 [  3.86495882e-07   2.20185508e-07   5.99622279e-07   5.22444732
e-07
   -5.06771020e-07  -1.80929400e-05]
 [ -8.65248555e-07  -1.39907725e-07   7.73408854e-07   1.11460292
e-05
   -5.55920895e-07  -5.95202609e-06]
 [  2.12455184e-07  -1.88359462e-06   6.33499450e-06   4.22130065
e-07
   -6.12082773e-07  -1.42483237e-06]]
Independent components scaled with mean
[[-0.04771087  0.00496636  0.00492989  0.00208307 -0.0059068
0.00159593]
 [ 0.00182027  0.0571306  -0.04596392 -0.00113553  0.00928388
-0.00925863]
 [ 0.00360762 -0.01311074 -0.09638513  0.00448148  0.08132511
0.00872532]
 [ 0.00463807  0.00127625  0.00476776  0.00160491 -0.00146026
-0.02758939]
 [-0.01038324 -0.00081094  0.00614959  0.03423984 -0.00160188
-0.00907607]
 [ 0.00254953 -0.01091782  0.0503713   0.00129675 -0.00176371
-0.00217268]]
```

4) For each vector in the ICA decomposition, write a sentence or two explaining what sort of object or property it corresponds to. What could these components be used for?

property it corresponds to? What could these components be used for?

Answer: **The first vector [-0.04771087 0.00496636 0.00492989 0.00208307 -0.0059068 0.00159593] again represents mainly the 'fresh'-feature, with a coefficient of -0.0477. The other features have a rather weak projection on the first dimension.**

The second vector [0.00182027 0.0571306 -0.04596392 -0.00113553 0.00928388 -0.00925863] corresponds mainly to the features 'Milk' and 'Grocery', but in different directions. This indicates that, other things equal, high 'Milk'-spending is associated with low 'Grocery'-spending and vice versa.

The third vector [0.00360762 -0.01311074 -0.09638513 0.00448148 0.08132511 0.00872532] has as strong association with the 'Grocery'- and 'Detergents_Paper'-features, again in opposite directions. This indicates a negative association between these features across the wholesalers customers.

The main charactistic of the fourth vector [0.00463807 0.00127625 0.00476776 0.00160491 -0.00146026 -0.02758939] is that this vector has a relatively strong negative association with 'delicatessen' (and only rather weak associations with the other features). Even though the coefficient are very low, the vector permits the interpretation that 'delicatessen' are negativly related to the 'fresh'- and 'grocery'-features.

##Clustering

In this section you will choose either K Means clustering or Gaussian Mixed Models clustering, which implements expectation-maximization. Then you will sample elements from the clusters to understand their significance.

###Choose a Cluster Type K Means Clustering

5) What are the advantages of using K Means clustering or Gaussian Mixture Models?

Answer: **Before discussing the advantages of K Means vs Gaussian Mixture models, it is helpful to observe that both methods are actually very similar. The main difference is that Gaussian Mixture models make a probabilistic assignment of points to classes depending on some distance metric, whereas K Means makes a deterministic assignment depending on some metric. Now, when the variance of the Gaussian mixtures is very small, this method becomes very similar to K Means, since the assignment probabilities to a specific cluster converge to 0 or 1 for any point in the domain. Because of the probabilistic assignment, Gaussian Mixtures (in contrast to K Means) are often characterized as soft clustering algorithms.**

An advantage of Gaussian Mixture models is that, if there is some a priori uncertainty about the assignment of a point to a cluster, this uncertainty is inherently reflected in the probabilistic model (soft assignment) and assignment probabilities can be computed for any data point after the model is trained. On the other hand, if a priori the clusters assignments are expected to be deterministic, K Means has advantages. An example would be a data generating process that actually is a mixture of Gaussians. Applying a Gaussian mixture model is more natural given this data generating process. When it comes to processing speed, the EM algorithm with Gaussian mixtures is generally slightly slower than Lloyd's algorithm for K Means, since computing the

normal probability (EM) is generally slower than computing the L2-norm (K Means). A disadvantage of both methods is that they can get stuck in local minima (this can be considered as the cost of solving NP-hard problems (global min for k-means) approximately).

Since there is no strong indication that the data are generated from a mixture of normals (this assesment may be different given more information about the nature of the spending data) and the goal is to "hard"-cluster them (and not assign probabilities), I decided the use the general-purpose k-means algorithm.

A decision on the number of clusters will be made by visualizing the final clustering and deciding whether k equals the number of data centers found by visual inspection. Note that many other approaches for this task could be utilized, such as silhoutte analysis (see for example http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html (http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html)).

6) Below is some starter code to help you visualize some cluster data. The visualization is based on this demo (http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html) from the sklearn documentation.

In [15]:

```
# Import clustering modules
from sklearn.cluster import KMeans
from sklearn.mixture import GMM
```

In [27]:

```
# (DONE) TODO: First we reduce the data to two dimensions using PCA to capture
pca = PCA(n_components=2)
reduced_data = pca.fit_transform(data)
print(reduced_data[:10])  # print upto 10 elements
```

```
[[  -650.02212207   1585.51909007]
 [  4426.80497937   4042.45150884]
 [  4841.9987068    2578.762176  ]
 [  -990.34643689  -6279.80599663]
 [-10657.99873116  -2159.72581518]
 [  2765.96159271   -959.87072713]
 [   715.55089221  -2013.00226567]
 [  4474.58366697   1429.49697204]
 [  6712.09539718  -2205.90915598]
 [  4823.63435407  13480.55920489]]
```

In [65]:

```
# (Done) TODO: Implement your clustering algorithm here, and fit it to the red
# The visualizer below assumes your clustering object is named 'clusters'

# TRIED OUT 2,3,4,5,6 CLUSTERS AND CONCLUDED THAT 3 CLUSTERS ARE A SENSIBLE CH
# WE OBTAIN ONE CENTRAL CLUSTER AND TWO CLUSTERS THAT SPREAD FAR OUT IN TWO DI
kmeans = KMeans(n_clusters=3)
clusters = kmeans.fit(reduced_data)
print(clusters)
```

```
KMeans(copy_x=True, init='k-means++', max_iter=300, n_clusters=3,
n_init=10,
    n_jobs=1, precompute_distances='auto', random_state=None, to
l=0.0001,
    verbose=0)
```

In [66]:

```
# Plot the decision boundary by building a mesh grid to populate a graph.
x_min, x_max = reduced_data[:, 0].min() - 1, reduced_data[:, 0].max() + 1
y_min, y_max = reduced_data[:, 1].min() - 1, reduced_data[:, 1].max() + 1
hx = (x_max-x_min)/1000.
hy = (y_max-y_min)/1000.
xx, yy = np.meshgrid(np.arange(x_min, x_max, hx), np.arange(y_min, y_max, hy))

# Obtain labels for each point in mesh. Use last trained model.
Z = clusters.predict(np.c_[xx.ravel(), yy.ravel()])
```

In [67]:

```
# (Done) TODO: Find the centroids for KMeans or the cluster means for GMM

centroids = kmeans.cluster_centers_
print('*** K MEANS CENTROIDS ***')
print(centroids)

# TRANSFORM DATA BACK TO ORIGINAL SPACE FOR ANSWERING 7
print('*** CENTROIDS TRANSFERED TO ORIGINAL SPACE ***')
print(pca.inverse_transform(centroids))
```

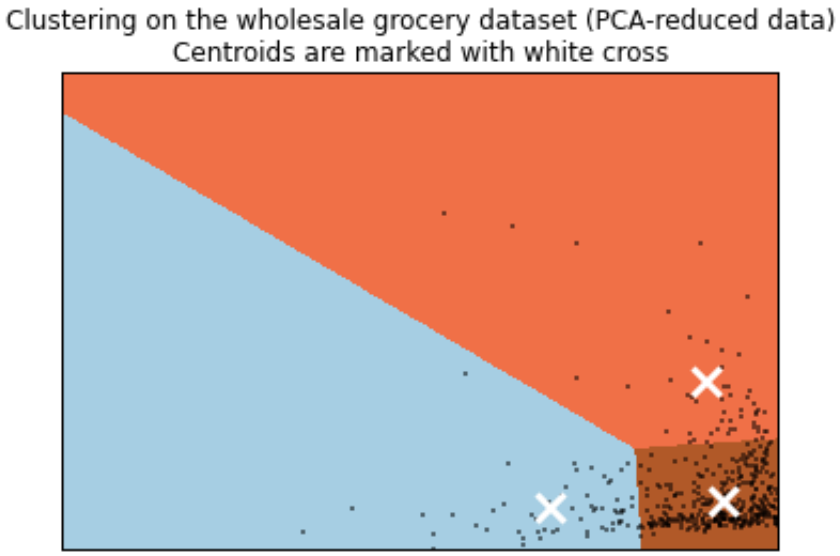
```
*** K MEANS CENTROIDS ***
[[-23978.86566553  -4445.56611772]
 [  1341.31124554  25261.39189714]
 [  4165.1217824   -3105.15811456]]
*** CENTROIDS TRANSFERED TO ORIGINAL SPACE ***
[[ 35908.284778     6409.08986458   6027.8378528    6808.69891192
    1088.15113313   2904.19473686]
 [  7896.19789901  18663.60082354  27183.75398875   2394.58291695
   12120.22381513   2875.42121485]
 [  8276.37635401   3689.87223746   5320.73032038   2495.4539104
    1776.40278857   1063.96606028]]
```

In [68]:

```
# Put the result into a color plot
Z = Z.reshape(xx.shape)
plt.figure(1)
plt.clf()
plt.imshow(Z, interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap=plt.cm.Paired,
           aspect='auto', origin='lower')

plt.plot(reduced_data[:, 0], reduced_data[:, 1], 'k.', markersize=2)
plt.scatter(centroids[:, 0], centroids[:, 1],
            marker='x', s=169, linewidths=3,
            color='w', zorder=10)
plt.title('Clustering on the wholesale grocery dataset (PCA-reduced data)\n'
          'Centroids are marked with white cross')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

```
/usr/local/lib/python3.4/site-packages/matplotlib/collections.p
y:590: FutureWarning: elementwise comparison failed; returning sc
alar instead, but in the future will perform elementwise comparis
on
  if self._edgecolors == str('face'):
```



7) What are the central objects in each cluster? Describe them as customers.

Answer: **In order to answer this question, I transfered the cluster centers back to the original space. The results can be found in the output above and at the end of this answer. I also used the feature statistics provided in README.md.**

The first cluster contains customers that have vastly (around 3 times) higher spendings in the 'Fresh'-category compared to the average, indicating that those customers specialize in selling fresh products. Also, customers in this cluster tend to place many orders in the 'Frozen'- and

'Delicatessen'-Category, but relatively few in the 'Detergents and Paper'-category.

Customers in the second cluster tend to spend the most overall, with particularly high spendings in the categories 'Milk', 'Grocery' and 'Detergents and Paper' and relatively low spendings in the 'Fresh' and 'Frozen' categories. Overall, this indicates that customers in this segment sell products that are more durable (i.e. not fresh).

The last cluster reflects small customers that have below-average annual spendings for each of the items. Appart from the low total spending, it is apparent that the spending distribution across categories is not pathological, that is, there is no category for which spendings are particularly low or high (given that spendings are low overall).

Regarding the question targeted at distinguishing the clusters visually: I generally had no problems distinguishing the clusters. Besides that, one observation is that the PCA reduction does not result in clusters that are well separated from each other. Reducing the data to three or four dimensions only (instead of two) may result in clusters that have more separation, but adds the complexity of having to visually represent the data using an (hyper-)cube instead of a plane. Of course, one could try to improve cluster representation using a 3-component PCA and a cube.

CENTROIDS TRANSFERED TO ORIGINAL SPACE

[[35908.28; 6409.09; 6027.84; 6808.70; 1088.15; 2904.19] (first cluster)

[7896.20; 18663.60; 27183.75; 2394.58; 12120.22; 2875.42] (second cluster)

[8276.38; 3689.87; 5320.73; 2495.45; 1776.40; 1063.97]] (third cluster)

###Conclusions

8) Which of these techniques did you feel gave you the most insight into the data?

Answer: **PCA in combination with K-means-clustering was the most insightful solution technique. Even though only six features are present in the data, PCA identified the the two features with the highest variance and enabled me two cluster the data in a lower dimensional space. I used PCA over ICA for dimensionality reduction, since the PCA eigenvalues provide a convenient ordering of the most important variance directions (eigenvectors) of the data that can later be used in clustering. ICA does not provide a similar rank order for the most important components. After preprocessing the data with PCA, K-means uncovered (clustered) interpretable customer groups that are helpful to the client. As argued before, I 'hard-clustered' the data with K-means (instead of GMM) because there is no clear indication about the underlying data generating process and hence I prefered a method that does not assume Gausssian distributions. Due to the clustering, we can now tell the client that, among the large customers, one group focusses on the 'fresh' category and the other on more durable products. The small customers, however, have a more balanced assortment (and we also concluded that small customers actually exist.).**

9) How would you use that technique to help the company design new experiments?

Answer: **If the client plans to change, retire or introduce new (delivery) strategies in the future, she could use the segmentation k-means found to try the strategies on subsets of the segments. That is, the client could choose a small number of customers in each segment and introduce the**

change for those customers only. It could then, after some time, see how the customer segments respond to the change (for example by asking them for feedback). Based on this, the client may conclude to not proceed with the new strategy, proceed only for one or two customer segments, or implement it for the whole customer base. The ability to receive feedback on a new strategy from a small subset of customers before making a final decision is advantageous for the client.

10) How would you use that data to help you predict future customer needs?

Answer: **(I assume this question relates to the data gathered in question 9)**

If the client implemented a strategy for a subset of each of the customer segements and asked them to provide feedback on a, say, 1 (very poor) to 5 (great) scale, the client could then run a linear regression of the feedback provided on a set of cluster membership features (dummy variables) to identify the needs of each of the customer segments (note that this linear regression basically results in segment averages for the response). This will help the client to determine whether the new strategy suits the customers needs (by segment) or not.

In [ ]: