

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 [1]:

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

import warnings

warnings.filterwarnings('ignore')

# 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() # print the first 5 rows

data.describe()
```

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

Out[1]:

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
count	440.000000	440.000000	440.000000	440.000000	440.000000	440.000000
mean	12000.297727	5796.265909	7951.277273	3071.931818	2881.493182	2881.493182
std	12647.328865	7380.377175	9503.162829	4854.673333	4767.854448	4767.854448
min	3.000000	55.000000	3.000000	25.000000	3.000000	3.000000
25%	3127.750000	1533.000000	2153.000000	742.250000	256.750000	256.750000
50%	8504.000000	3627.000000	4755.500000	1526.000000	816.500000	816.500000
75%	16933.750000	7190.250000	10655.750000	3554.250000	3922.000000	3922.000000
max	112151.000000	73498.000000	92780.000000	60869.000000	40827.000000	40827.000000

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: We can expect that PCA will detect primary and secondary vectors that explains the data variance.

Data will probably be explained in just first two dimensions. By using PCA we will be able to reduce these 6 variables to just the two of them that best captures that information. By having smaller number of dimensions we will be able to visualize the data better.

Based on Standard Deviation of the values Fresh column has highest value followed by grocery and Milk. The Primary vector will be composed based on this variation.

ICA will find vectors which are statistically separate from each other by "whitening" the overlap in data and separating each vector into a separate signal.

Using ICA will be useful in separating data in preparation for clustering if we can not find clearly separable boundaries.

```
In [3]: from sklearn import preprocessing
std = preprocessing.StandardScaler()
data_std = std.fit_transform(data)
```

###PCA

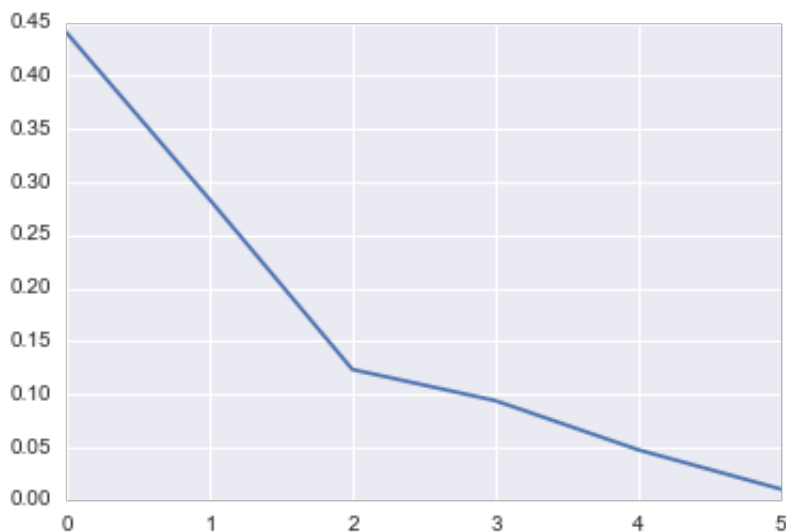
```
In [46]: # TODO: Apply PCA with the same number of dimensions as variables in the
from sklearn.decomposition import PCA
pca = PCA(n_components=data.shape[1])

pca.fit(data_std)
pca_dt = pca.transform(data_std)

pca_df = pd.DataFrame(pca_dt , columns=['PC1','PC2', 'PC3', 'PC4', 'PC5',
#restrict the Dataframe to first 3

# Print the components and the amount of variance in the data contained i
#print 'components_', pca.components_
print 'explained_variance_ratio_', pca.explained_variance_ratio_
print plt.plot(pca.explained_variance_ratio_)
#plt.legend()
```

```
explained_variance_ratio_ [ 0.44082893  0.283764    0.12334413  0.0939
5504  0.04761272  0.01049519]
[<matplotlib.lines.Line2D object at 0x10e193110>]
```



To interpret each component, we must compute the correlations between the original data for each variable and each principal component.

These correlations are obtained using the correlation procedure. In the variable statement we will include the first three principal components, PC1, PC2, and PC3, in addition to all six of the original variables. We will use these correlations between the principal components and the original variables to interpret these principal components.

Because of standardization, all principal components will have mean 0. The standard deviation is also given for each of the components and these will be the square root of the eigenvalue.

We want to examine correlations between Principal components and the original values.

To interpret principal components we look at which variables are most strongly correlated to each other (which number are large in magnitude - furthest from zero in either direction)

Here a correlation value above ± 0.5 is deemed important.

PC1

The first principal component is strongly correlated to 3 of the 6 original variables. The first principal component decreases with Milk, Detergent & Paper and Grocery. This suggest that these 3 categories vary together.

This component can be viewed as retailers interest in Milk, Detergent & Paper and Grocery

PC2

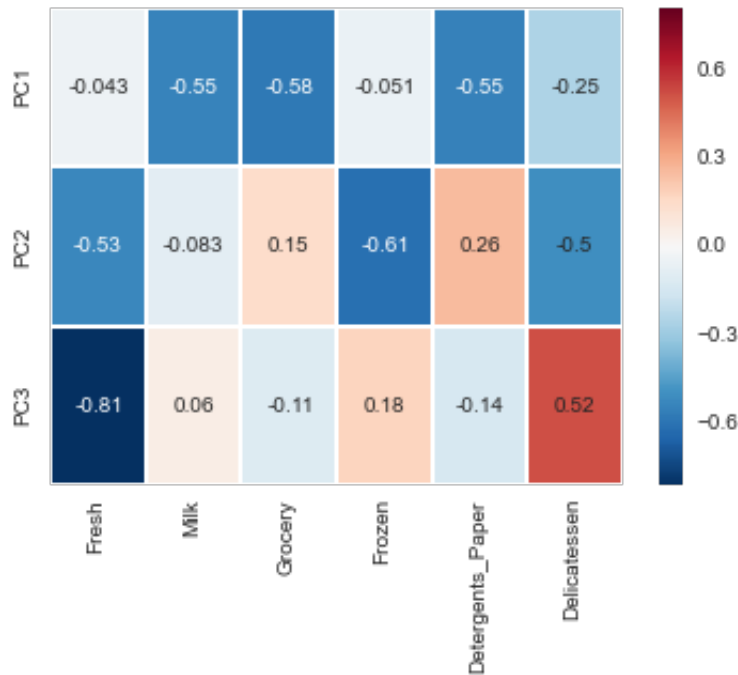
The second Principal Component decreases with Delicatessen Fresh, Frozen and Frozen.

PC3

Is made up from Delicatessen and on the opposite side fresh.

```
In [19]: pca_correlations = pd.DataFrame(pca.components_, columns=data.columns, in
sns.heatmap(pca_correlations, annot=True)
```

```
Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x10d795b10>
```



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?

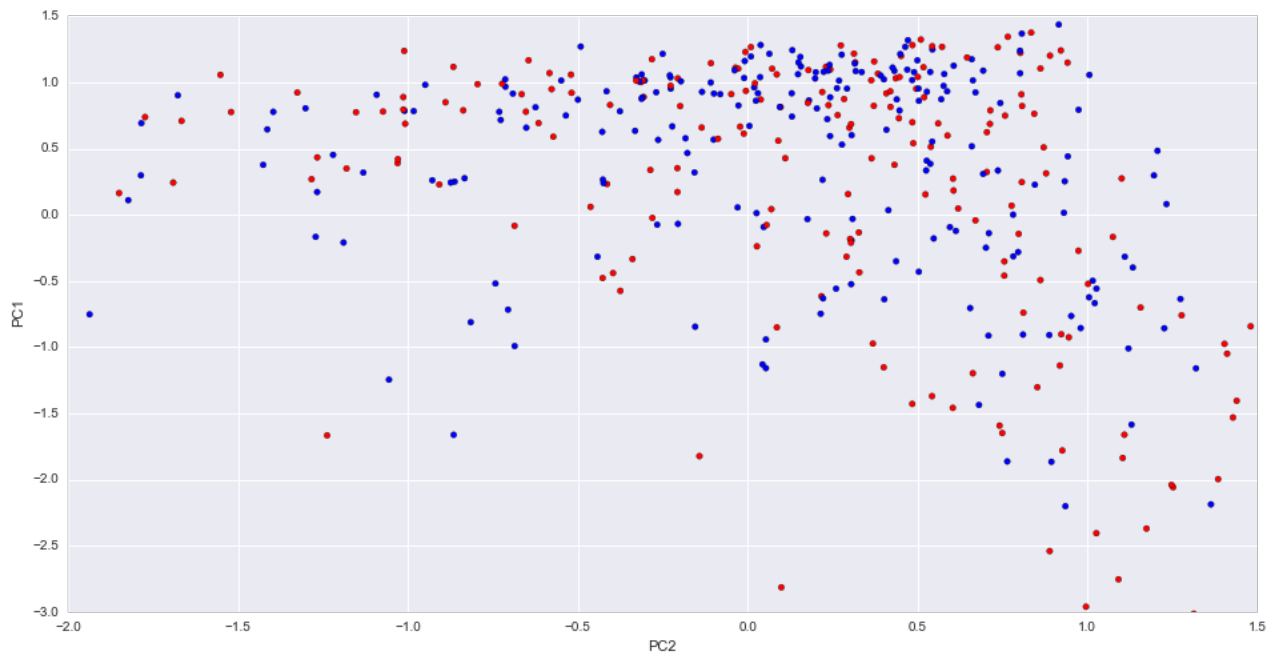
Answer: We see that the first PC already explains 44% of the variance, while the second one accounts for another 28% for a total of almost 72% between the two of them. Third dimension is responsible for 12% for 84%.

I am going to choose 3 dimensions for my analyses as they explain 88% of variance.

In [7]:

```
ax = pca_df.plot(kind='scatter', x='PC2', y='PC1', figsize=(16,8), c=['b'  
ax.set_xlim (-2, 1.5)  
ax.set_ylim (-3, 1.5)  
  
# most of the variance is along the PC1 axis
```

Out[7]: (-3, 1.5)



```
In [8]: #attempting to make 3D plot with our 3 variables

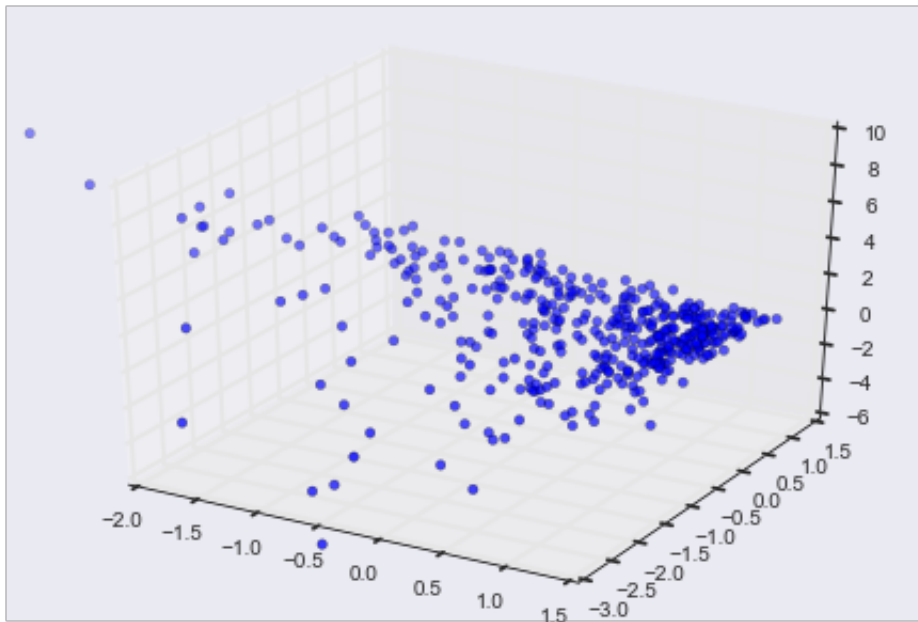
from matplotlib import pyplot
import pylab
from mpl_toolkits.mplot3d import Axes3D
import random

fig = pylab.figure()
ax = Axes3D(fig)

ax.scatter(pca_df.PC1, pca_df.PC2, pca_df.PC3, c=['b', 'r'], marker='o',

ax.set_xlim (-2, 1.5)
ax.set_ylim (-3, 1.5)

pyplot.show()
```




```
In [9]: from sklearn.cluster import KMeans

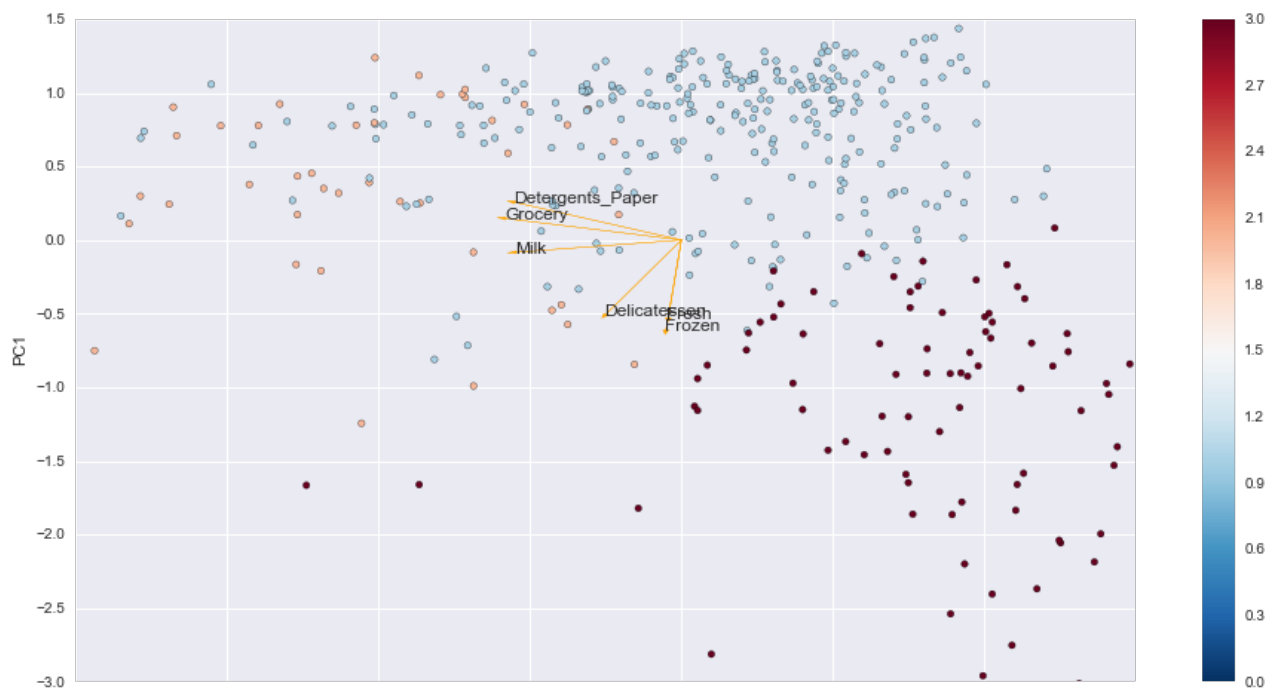
df = pd.DataFrame(data_std, columns=['Fresh', 'Milk', 'Grocery', 'Frozen'])

kmeans = KMeans(n_clusters=4)
clusters = kmeans.fit(data)

pca_df['cluster'] = pd.Series(clusters.labels_, index=pca_df.index)

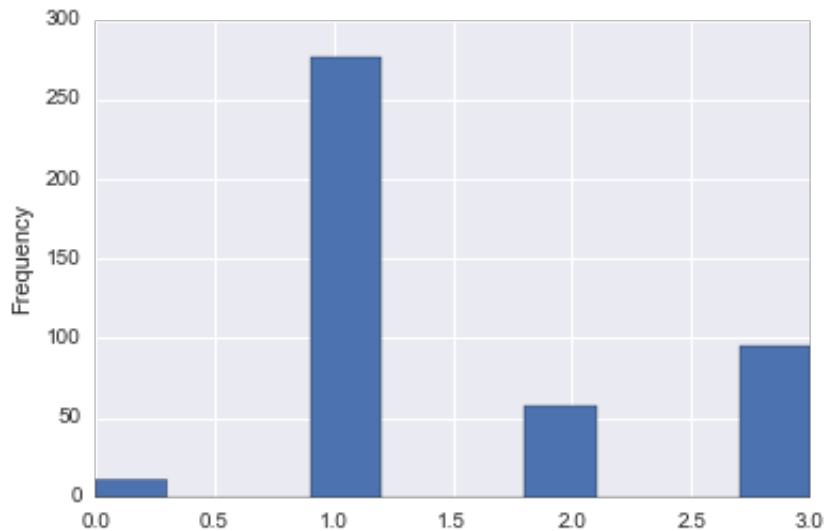
ax = pca_df.plot(
    kind='scatter',
    x='PC2', y='PC1',
    c=pca_df.cluster.astype(np.float),
    colormap = 'RdBu_r',
    figsize=(16,8),
    xlim = [-2, 1.5],
    ylim = [-3, 1.5] )

for i, (pc1, pc2) in enumerate(zip(pca.components_[0], pca.components_[1])):
    ax.arrow(0, 0, pc1, pc2, width=0.001, fc='orange', ec='orange')
    ax.annotate(df.columns[i], (pc1, pc2), size=12)
```



```
In [10]: pca_df['cluster'].plot(kind='hist')
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x10c9cc6d0>
```



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

Answer: After applying PCA I have applied clustering. Fine tuning the number of clusters parameters, I can observe that the data is separated in four different clusters. Majority of the custores are in the Detergent_Paper and Grocery

First two Dimmensions of PCA are two dimmensions that explain majority (86%) of variance in data. This is done by combining information from number of different original dimmensions to produce new dimmensions. We can use this new dimmensions to represent the data in further machine learning algoritams, having advantage that we have reduced dimmensionality.

From Above biplot we can see that customers can be grouped in 3-4 groups.

Detergent_Paper and Grocery Milk Delicatessen Frozen

The First Component places approximatly equal weight to Detergent_Paper and Grocery Milk and much less weight to Delicatessen and Frozen

The second component places most of its weight on Delicatessen & Frozen

From this we can see that Detergent_Paper, Grocery and Milk are correlated to each other while Delicatesen and Frozen are in a category of their own.

By examining the differences between the customers via two principal components. Vectors suggest that Customers with high demand for Detergent / paper / Grocery and milk have little demand for frozen and about 25% of demand for Delicatessen

We can use PCA components to visualize the data in two dimensions while retaining most of the characteristics of the original data.

###ICA

In [20]:

```

# TODO: Fit an ICA model to the data
# Note: Adjust the data to have center at the origin first!
from sklearn.decomposition import FastICA
from sklearn import preprocessing

ica = FastICA(random_state=311)

stnd = preprocessing.StandardScaler()
ica_scale = preprocessing.StandardScaler()
ica_data = stnd.fit_transform(data_stnd)

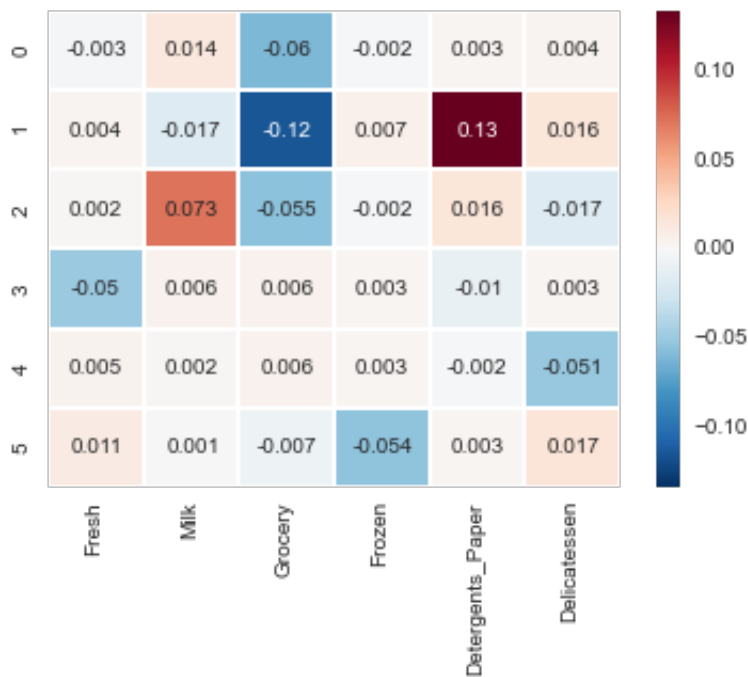
S_ = ica.fit_transform(data_stnd)
A_ = ica.mixing_

assert np.allclose(ica_data, np.dot(S_, A_.T) + ica.mean_)

ica_data= pd.DataFrame(ica.components_,columns=data.columns).round(3)
sns.heatmap(ica_data, annot=True)

```

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x10de0eb50>



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?

From the heat map above we will examine how each of the original components relate to recovered data through ICA.

We can see that:

- First component primarily consists of Grocery
- Second component consists of Detergent_Paper and inversely of Grocery
- Third component consists of Milk and Grocery
- Fourth Component is primarily Fresh
- Fifth component mostly consists of Delicates
- Final Sixth component is mostly Frozen.

These components can be used to reduce the feature set into new ones that are statistically independent from one another. This can be useful in Clustering as it can help visually separating the customer segments.

##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

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

Answer:

Advantages of K Means clustering:

- With a large number of variables, K-Means may be computationally faster than hierarchical clustering (if K is small).
- K-Means may produce tighter clusters than hierarchical clustering, especially if the clusters are globular.

- K-means is simple and converges quickly to a local optimum. It has linear time complexity $O(I K N)$ where I is the number of iterations

Advantages of Gaussian Mixture Models:

- It is the fastest algorithm for learning mixture models
- The GMM algorithm is a good algorithm to use for the classification of static postures and non-temporal pattern recognition.
- the total run time of the algorithm is $O(k^2 n)$ (or $O(kmn)$ if $k < m$).

Comparison of the K-means algorithm with the EM algorithm for Gaussian mixtures shows that there is a close similarity. Whereas the K-means algorithm

algorithm
 performs a hard assignment of data points to clusters, in which each data point is associated uniquely with one cluster, the EM algorithm makes a soft assignment based on the posterior probabilities.

Which do you think is more appropriate for this problem?
 Because dataset is not clearly separable Gaussian Mixtures model will provide more flexible boundaries, and as such is more appropriate for this problem

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) (http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html) from the sklearn documentation.

```
In [21]: # Import clustering modules
from sklearn.cluster import KMeans
from sklearn.mixture import GMM
```

```
In [22]: # TODO: First we reduce the data to two dimensions using PCA to capture v
reduced_data = pca_df[['PC1', 'PC2']]
print reduced_data[:10] # print upto 10 elements
```

	PC1	PC2
0	-0.193291	0.305100
1	-0.434420	0.328413
2	-0.811143	-0.815096
3	0.778648	-0.652754
4	-0.166287	-1.271434
5	0.156170	0.295141
6	0.335288	0.525003
7	-0.140586	0.230993
8	0.517320	0.659363
9	-1.592109	0.741011

```
In [23]: # TODO: Implement your clustering algorithm here, and fit it to the reduced data
# The visualizer below assumes your clustering object is named 'clusters'

clusters = KMeans(n_clusters=3)
clusters.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, tol=0.0001,
       verbose=0)
```

```
In [24]: # Plot the decision boundary by building a mesh grid to populate a graph.
x_min, x_max = reduced_data['PC1'].min() , reduced_data['PC1'].max()
y_min, y_max = reduced_data['PC2'].min() , reduced_data['PC2'].max()
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()])
```

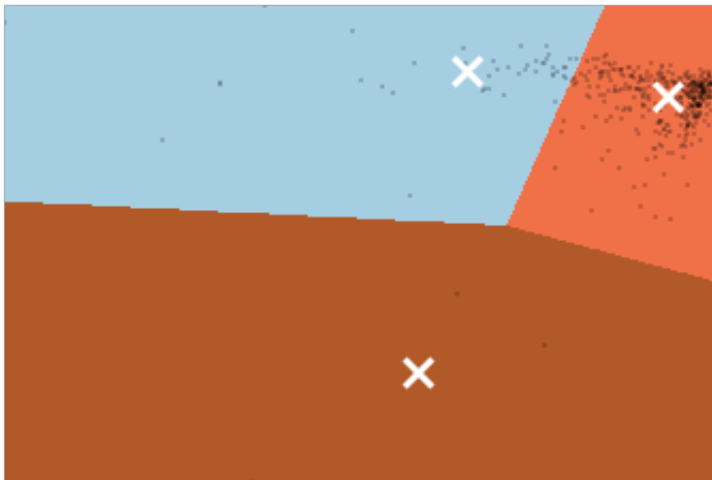
```
In [25]: # TODO: Find the centroids for KMeans or the cluster means for GMM

centroids = clusters.cluster_centers_
print centroids
```

```
In [26]: # 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['PC1'], reduced_data['PC2'], '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)
          'Centroids are marked with white cross')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

Clustering on the wholesale grocery dataset (PCA-reduced data)
Centroids are marked with white cross




```
In [27]: #TODO: Implement your clustering algorithm here, and fit it to the reduce
# The visualizer below assumes your clustering object is named 'clusters'

clusters = GMM(n_components=3).fit(reduced_data)
# clusters = GMM(n_components=3).fit(reduced_data)
# clusters = KMeans(n_clusters=2).fit(reduced_data)
# clusters = KMeans(n_clusters=3).fit(reduced_data)
print clusters

GMM(covariance_type='diag', init_params='wmc', min_covar=0.001,
    n_components=3, n_init=1, n_iter=100, params='wmc', random_state=None,
    thresh=None, tol=0.001)
```

```
In [28]: # Plot the decision boundary by building a mesh grid to populate a graph.
x_min, x_max = reduced_data['PC1'].min() , reduced_data['PC1'].max()
y_min, y_max = reduced_data['PC2'].min() , reduced_data['PC2'].max()
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 [17]: # TODO: Find the centroids for KMeans or the cluster means for GMM

centroids = clusters.means_
# centroids = clusters.cluster_centers_
print centroids

[[-2.87924272 -1.70246154]
 [ 0.81499208 -0.09020858]
 [-0.73775307  0.63075796]]
```

```
In [29]: Put the result into a color plot
= Z.reshape(xx.shape)
t.figure(1)
t.clf()
t.imshow(Z, interpolation='nearest',
        extent=(xx.min(), xx.max(), yy.min(), yy.max()),
        cmap=plt.cm.Paired,
        aspect='auto', origin='lower')

t.plot(reduced_data['PC1'], reduced_data['PC2'], 'k.', markersize=2)
t.scatter(centroids[:, 0], centroids[:, 1],
        marker='x', s=169, linewidths=3,
        color='w', zorder=10)
t.title('Clustering on the wholesale grocery dataset (PCA-reduced data)\n'
        'Centroids are marked with white cross')
t.xlim(x_min, x_max)
t.ylim(y_min, y_max)
t.xticks(())
t.yticks(())
t.show()
```

Clustering on the wholesale grocery dataset (PCA-reduced data)
Centroids are marked with white cross



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

Answer: From PCA we identified 3 grouping / correlations and I used this value as K . We can see that certain our customers can be divided in 3 such groups as well.

Majority of the customers are in tow top right cluster, I would describe this as wholesale / large customer.

Second group of the customers I would describe as Grocery stores mainly purchasing Detergents & paper, Groceries and Milk.

Third group I would classify as Convenience stores (7 Eleven's, etc) purchasing wide range of products in smaller quantities

###Conclusions

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

Answer: PCA gave me most insight by enabling visualization of the customers in addition to KMeans clustering where I was able to assign clusters to individual customers.

Overall I found that Gaussian Mixture Model (GMM) provides more logical clustering of the data as Kmeans wasn't able to find a clear divide in the clusters.

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

Answer: We could use results of PCA as input parameters in regression, or other types of clusters. Having the ability to cluster the customers we can treat them as groups of customers on which we can apply different experiments. Such as trying to predict how would spending change if demand for one of the groups of products is changed.

Additionally by assigning customers to cluster we can design A/B tests where we are able to target specific market segments. What is important to wholesalers may not be important to convenience store owners.

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

Answer: By seeing what type of products customers are interested in and this can influence how we interact with them. For example we can target only specific segment of our customers with special discounts and incentives.

Using the cluster assignment in addition to previous purchase we can predict the effect of certain incentives such as discounts to total increase of sales.

References: K-Means Clustering Overview -

http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/K-Means_Clustering_Overview.htm

(http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/K-Means_Clustering_Overview.htm)

