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

# 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
```

```
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
```

```
/Users/seanhegarty/Library/Enthought/Canopy_64bit/User/lib/python
2.7/site-packages/matplotlib/font_manager.py:273: UserWarning: Mat
plotlib is building the font cache using fc-list. This may take a
moment.
  warnings.warn('Matplotlib is building the font cache using fc-li
st. This may take a moment.')
```

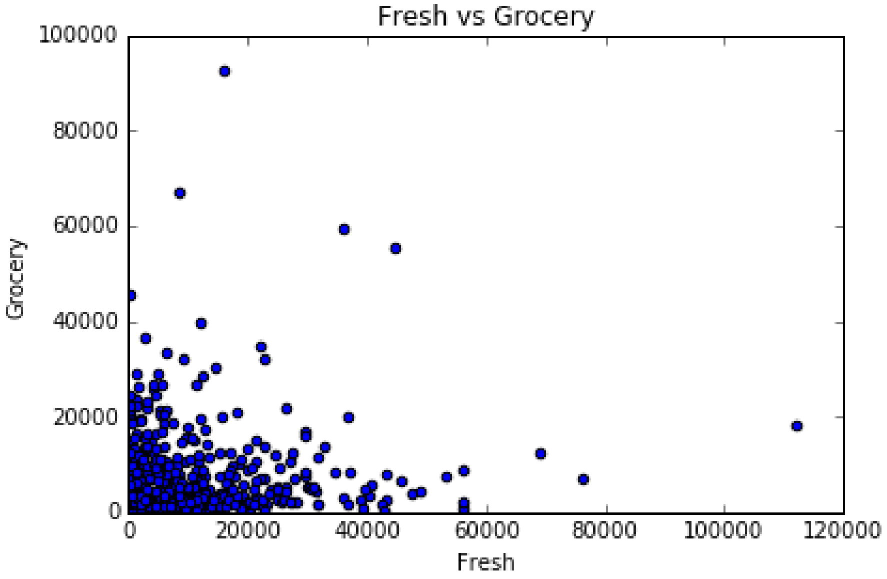
Feature Transformation

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: I expect that PCA will find its first dimensions within the fresh/ perishable items (Fresh, Milk, Grocery), given they have a standard deviation roughly double the three other items in the list. Seeing as all the columns are measured in the same units, normalizing the columns is not completely necessary. I expect not normalizing the data to further enforce the correlation between the strength of This makes sense to me because smaller stores would have a problem with selling perishable products in bulk.

ICA, on the other hand, I would expect to report two transformations representing different types of products, if it is told to find two signals. For example, maybe the Milk and the Groceries are bought in more similar frequencies compared to items like Detergent and Frozen items. One reason for this might be the rate at which people consume items like milk and groceries that have experation dates on the order of days and weeks, while Detergent and Frozen items can last for months or more.

```
In [182]: plt.scatter(data['Fresh'],data['Grocery'])
          plt.title("Fresh vs Grocery")
          plt.xlabel("Fresh")
          plt.ylabel("Grocery")
          plt.ylim([0,100000])
          plt.xlim([0,120000])
          plt.show()
```



PCA

In [360]:

```
# TODO: Apply PCA with the same number of dimensions as variables i
n the dataset
from sklearn.decomposition import PCA
pca = PCA(n_components=6)
pca.fit(data).transform(data)
# Print the components and the amount of variance in the data conta
ined in each dimension
components = pd.DataFrame(pca.components_)
components.columns = ["Fresh", "Milk", "Grocery", "Frozen", "Deterg
ents_Paper", "Delicatessen"]
print components
print pca.explained_variance_ratio_
```

```
      Fresh      Milk   Grocery    Frozen  Detergents_Paper  Delic
atessen
0 -0.976537 -0.121184 -0.061540 -0.152365          0.007054     -
0.068105
1 -0.110614  0.515802  0.764606 -0.018723          0.365351      
0.057079
2 -0.178557  0.509887 -0.275781  0.714200         -0.204410      
0.283217
3 -0.041876 -0.645640  0.375460  0.646292          0.149380     -
0.020396
4  0.015986  0.203236 -0.160292  0.220186          0.207930     -
0.917077
5 -0.015763  0.033492  0.410939 -0.013289         -0.871284     -
0.265417
0.934815965411
```

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 variance drops off fairly quickly as the principle component number increases. It would appear that most of the variance, as expected, comes from the features with the highest initial standard deviation.

Without having a supervised learning problem to determine the effectiveness of the PCA dimensions, I would probably got with a rule of thumb that I have come across on multiple sources that say to use enough dimensions to explain roughly 95% of the variance to limit the information loss. Seeing as using the first three PCA dimensions would be 93.5% of the original variance, there is not a significant amount more information to be gained from more dimensions.

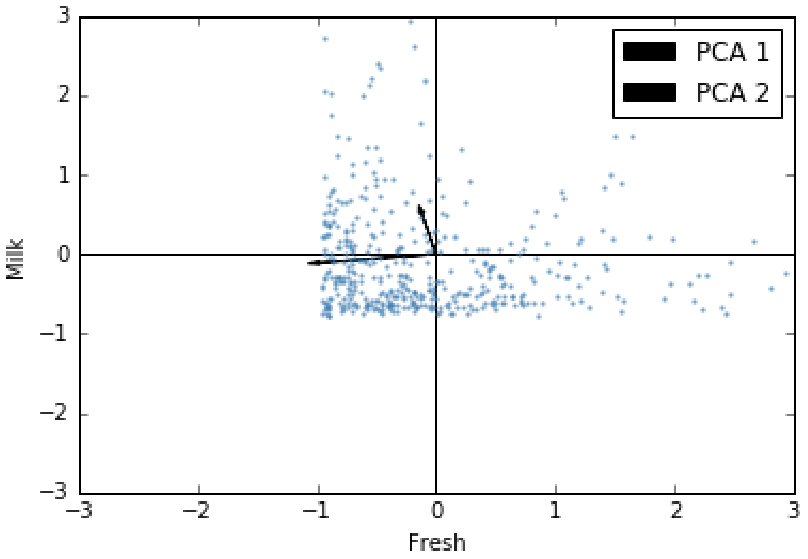
3) What do the dimensions seem to represent? How can you use this information?

Answer:

The first dimension mainly represents the purchases of *Fresh* products, whereas the second dimension represents the *Groceries*, *Milk*, and *Detergents_Paper*. This information indicates that these are products that differ largely in purchasing patterns between large and small customers. The wholesale supplier might be able to continue using a bulk delivery for the items with smaller variance while retaining smaller customers by continuing the daily deliveries.

In [365]:
```
#Plot Two dimensions of the data
def plot_samples(S, pca, n_comps=2):
    plt.scatter(S['Fresh'], S['Milk'], s=2, marker='o', zorder=10,
                color='steelblue', alpha=0.5)
    plt.hlines(0, -3, 3)
    plt.vlines(0, -3, 3)
    plt.xlim(-3, 3)
    plt.ylim(-3, 3)
    plt.xlabel('Fresh')
    plt.ylabel('Milk')
    #Plot Principal Component vectors
    pca_comps = []
    pca_dim = []
    for i in range(n_comps):
        pca_comps.append(
            plt.arrow(0,0,pca.components_.T[i][0],pca.components_.T
[i][1],
                      head_width=0.05, head_length=0.1, fc='k', ec
='k')
            )
        pca_dim.append("PCA {}".format(i+1))
    plt.legend(pca_comps,pca_dim)
ax = plot_samples((data-np.mean(data))/np.std(data), pca, n_comps=
2)

plt.show()
```



ICA

In [362]:
```
# TODO: Fit an ICA model to the data
# Note: Adjust the data to have center at the origin first!
from sklearn.decomposition import FastICA
ica = FastICA(n_components=4)
ica.fit_transform( (data - np.mean(data))/ np.std(data) )

# Print the independent components
ica_comps = pd.DataFrame(ica.mixing_.T)
ica_comps.columns = ['Fresh', 'Milk', 'Grocery', 'Frozen', 'Deterge
nts_Paper', 'Delicatessen']
print (ica_comps)
```

```
       Fresh       Milk    Grocery     Frozen  Detergents_Paper  D
elicatessen
0   2.943383   8.581862   2.014258   6.872764         -0.906468
20.437553
1 -20.694793  -0.766649   0.986095  -4.910462          2.505549
-2.067553
2  -0.463758 -16.726952 -20.183838   0.578851        -20.076084
-2.043857
3   1.585272   0.228237  -0.823184  19.188865         -1.522040
1.117699
```

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?

Answer:

It appears from the mixing matrix vectors shown transposed above that while *Fresh* products, *Frozen* products, and *Delicatessen* products all seem to run independently of one another, *Milk*, *Grocery*, and *Detergents_Paper* share similar transformations from the original data in sign and magnitude.

The 0th component correlates very highly with *Delicatessen* followed by *Milk*. These products tend to have close expiration dates compared to almost all other products, which could mean that it represents a trend of purchasing patterns that are products purchased in short periods.

The 1st component is highly negatively correlated with *Fresh* products. It is weakly correlated with all other features. This could suggest that the *Fresh* products are often bought in trends that run against the trends of buying all other products. In other words, customers that buy mostly fresh products tend to stray away from purchasing many other type of product.

The 2nd component strongly negatively correlates with *Milk*, *Grocery*, and *Detergents_Paper* products. This component could be representative of products that are bought more often by families, whether they be at a smaller or larger supplier.

The 3rd component very strongly correlates with *Frozen* items. This component seems to deal solely with the trends set by the purchases of *Frozen* food.

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:

Clustering algorithms will allow segmentation of the data. The interpretation is pretty straight forward in that it separates the points based on their distance to a given amount of points. Specifically, if some points are closer to marker A and farther from marker B, then they will be in cluster A as all those points are similar to each other, as determined by a distance metric.

K Means has a minimal number of parameters to set accoring to sklearn clustering (http://scikit-learn.org/stable/modules/clustering.html). Given that I am not an expert in the field of wholesale products, this minimal number of parameters in K Means is a minor advantage over GMMs. Also, for any supervised analysis, we are going to assign customers to clusters. K Means already does this, where GMMs provide a probability with which to make the decision.

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 [368]:

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

In [502]:

```
# TODO: First we reduce the data to two dimensions using PCA to capture variation
pca = PCA(n_components=2)
reduced_data =  pca.fit_transform(data)
#reduced_data =  pca.fit_transform(np.log(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 [554]:

```
# TODO: Implement your clustering algorithm here, and fit it to the
reduced data for visualization
# The visualizer below assumes your clustering object is named 'clu
sters'
n_clusters = 3
clf = KMeans(n_clusters=n_clusters, n_init=20)
#clf = GMM(n_clusters)
clusters = clf.fit( reduced_data )
print clusters
```

```
KMeans(copy_x=True, init='k-means++', max_iter=300, n_clusters=3,
n_init=20,
    n_jobs=1, precompute_distances='auto', random_state=None, tol=
0.0001,
    verbose=0)
```

In [555]:

```
# 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 [558]:

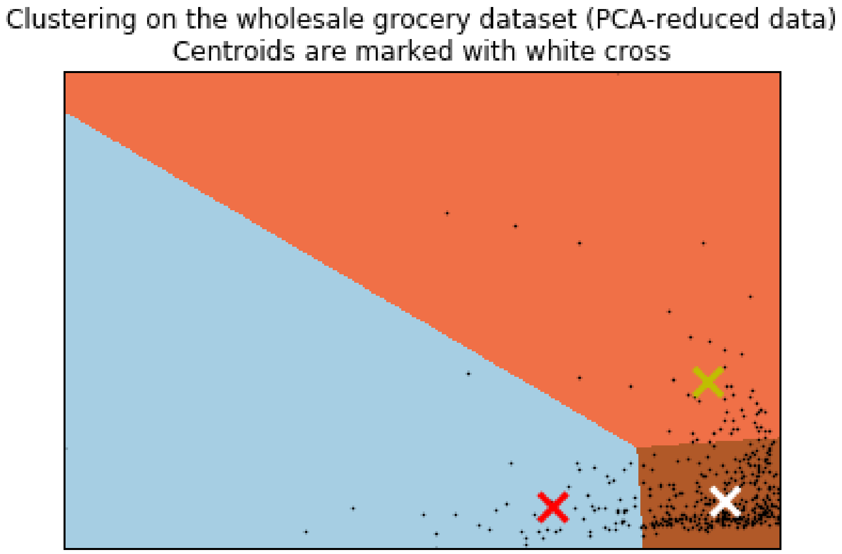
```
# TODO: Find the centroids for KMeans or the cluster means for GMM
centroids = clusters.cluster_centers_
#centroids = clusters.means_ #Uncomment this line if using a GMM fo
r clustering. Also, comment out the line above.
print centroids
```

```
[[-23978.86566553  -4445.56611772]
 [  1341.31124554  25261.39189714]
 [  4165.1217824   -3105.15811456]]
```

In [557]:

```
# 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=['r','y','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()
```



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

Answer:

It is first important to understand the dimensions of the graph shown above. The X-axis, as discussed in the PCA section, is largely representative of the *Fresh* products purchased by the customers, while the Y-axis is largely represented by *Milk*, *Groceries*, and *Detergent_Paper* products purchased by the customers.

Mathematically, each central point represents the average location of all points classified to that cluster. Conceptually, the central objects are the clustering algorithm's representation of a typical customer in that cluster, or group, according to the above two dimensions. The clusters, shown based on the 2-dimensional representation provided by PCA, are inclusive of all customers similar in purchasing patterns discussed in the previous paragraph.

After trying a few different numbers of clusters, it seems that three clusters best fits the data. Two clusters was not enough because it did not account for the large variation in the y-axis of the graph shown above. There is a number of metrics for determining the number of clusters that one should use when using a clustering algorithm, but according to Machine Learning expert Andrew Ng, "the most common [method for choosing number of clusters] is to choose the number of clusters by hand." (https://www.coursera.org/learn/machine-learning/lecture/Ks0E9/choosing-the-number-of-clusters)

The cluster represented by the yellow X in the above graph represents customers that tend to purchase more *Milk*, *Groceries*, and *Detergent_Paper* products than they do *Fresh* products. The cluster represented by the red X in the above graph represents customers that tend to purchase more *Fresh* products than they do *Milk*, *Groceries*, and *Detergent_Paper* products. Finally, the cluster represented in the above graph by a white X represents customers that buy roughly similar amounts of *Fresh*, *Milk*, *Groceries*, and *Detergent_Paper* products.

Increasing the number of clusters past 3 will show that, with K Means clustering, more clusters are built based around the same rules discussed in the previous paragraph. There will be one customer with similar purchases of *Fresh*, *Milk*, *Groceries*, and *Detergent_Paper* products, and all other clusters will be divide the dimensions segmented in the white and yellow X clusters.

Conclusions

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

Answer:

Given that PCA was able to reveal what products might be of more concern to the supplier by indicating each products' contribution to the first, second, and third principal components, I would have to say that PCA has a good start. In addition, PCA offered a 2-dimensional representation that could be used for visualization in later clustering. Visualizations are always easier to interpret and better for presentation to a non-technical audience.

On the other hand, clustering offers simpler, more easily interpreted lables. If over this threshold, this is a **large *Fresh* product customer**, and otherwise. This information is clearly actionable, especially after PCA covers the preprocessing step. Clustering, using K-Means, gives the most insight into this data because it offers clear A/B distinctions for future experiments.

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

Answer:

Using the cluster labels, I would design different experiments based on either of the two individual segments rather than grouping them as having the same needs. This way the results of an experiment would be more specified. The control group would be a section of the cluster that will not experience the change, and the test group will be the other fraction of that same cluster that will experience the change. The variation that will be examined is the difference between costs between the test and control groups.

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

Answer:

The data could be used to build regressions based on customer type, where type is determined by which cluster they belong to, that would predict how much a customer of a certain type might be expected to purchase of a particular product in the future.

Ideally, clustering customers into different groups will allow the results to be viewed independently of different clusters. This should more accurately represent smaller collections of customers, because different types of customers will have purchasing patterns that could skew the results of a regression downward or upward. For example, the customers in the white X cluster that is largely representative of large buyers of *Fresh* products will not be able to skew the results of an experiment done on customers represented by the red or yellow X clusters.

```
In [ ]:
```