Machine Learning Engineer Nanodegree

Unsupervised Learning

Project: Creating Customer Segments

Welcome to the third project of the Machine Learning Engineer Nanodegree! In this notebook, some template code has already been provided for you, and it will be your job to implement the additional functionality necessary to successfully complete this project. Sections that begin with **'Implementation'** in the header indicate that the following block of code will require additional functionality which you must provide. Instructions will be provided for each section and the specifics of the implementation are marked in the code block with a `'TODO'` statement. Please be sure to read the instructions carefully!

In addition to implementing code, there will be questions that you must answer which relate to the project and your implementation. Each section where you will answer a question is preceded by a **'Question X'** header. Carefully read each question and provide thorough answers in the following text boxes that begin with **'Answer:'**. Your project submission will be evaluated based on your answers to each of the questions and the implementation you provide.

> **Note:** Code and Markdown cells can be executed using the **Shift + Enter** keyboard shortcut. In addition, Markdown cells can be edited by typically double-clicking the cell to enter edit mode.

Getting Started

In this project, you will analyze a dataset containing data on various customers' annual spending amounts (reported in *monetary units*) of diverse product categories for internal structure. One goal of this project is to best describe the variation in the different types of customers that a wholesale distributor interacts with. Doing so would equip the distributor with insight into how to best structure their delivery service to meet the needs of each customer.

The dataset for this project can be found on the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Wholesale+customers). For the purposes of this project, the features `'Channel'` and `'Region'` will be excluded in the analysis — with focus instead on the six product categories recorded for customers.

Run the code block below to load the wholesale customers dataset, along with a few of the necessary Python libraries required for this project. You will know the dataset loaded successfully if the size of the dataset is reported.

In [40]:
```
# Import libraries necessary for this project
import numpy as np
import pandas as pd
from IPython.display import display # Allows the use of display() for DataFram
es

# Import supplementary visualizations code visuals.py
import visuals as vs

# Pretty display for notebooks
%matplotlib inline

# Load the wholesale customers dataset
try:
    data = pd.read_csv("customers.csv")
    data.drop(['Region', 'Channel'], axis = 1, inplace = True)
    print "Wholesale customers dataset has {} samples with {} features each.".
format(*data.shape)
except:
    print "Dataset could not be loaded. Is the dataset missing?"
```

```
Wholesale customers dataset has 440 samples with 6 features each.
```

Data Exploration

In this section, you will begin exploring the data through visualizations and code to understand how each feature is related to the others. You will observe a statistical description of the dataset, consider the relevance of each feature, and select a few sample data points from the dataset which you will track through the course of this project.

Run the code block below to observe a statistical description of the dataset. Note that the dataset is composed of six important product categories: **'Fresh'**, **'Milk'**, **'Grocery'**, **'Frozen'**, **'Detergents_Paper'**, and **'Delicatessen'**. Consider what each category represents in terms of products you could purchase.

In [41]:
```
# Display a description of the dataset
display(data.describe())
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Del
count	440.000000	440.000000	440.000000	440.000000	440.000000	440
mean	12000.297727	5796.265909	7951.277273	3071.931818	2881.493182	152
std	12647.328865	7380.377175	9503.162829	4854.673333	4767.854448	282
min	3.000000	55.000000	3.000000	25.000000	3.000000	3.0
25%	3127.750000	1533.000000	2153.000000	742.250000	256.750000	408
50%	8504.000000	3627.000000	4755.500000	1526.000000	816.500000	965
75%	16933.750000	7190.250000	10655.750000	3554.250000	3922.000000	182
max	112151.000000	73498.000000	92780.000000	60869.000000	40827.000000	479

Implementation: Selecting Samples

To get a better understanding of the customers and how their data will transform through the analysis, it would be best to select a few sample data points and explore them in more detail. In the code block below, add **three** indices of your choice to the `indices` list which will represent the customers to track. It is suggested to try different sets of samples until you obtain customers that vary significantly from one another.

In [42]:
```
# TODO: Select three indices of your choice you wish to sample from the datase
t
indices = [3,350,285]

# Create a DataFrame of the chosen samples
samples = pd.DataFrame(data.loc[indices], columns = data.keys()).reset_index(d
rop = True)
print "Chosen samples of wholesale customers dataset:"
display(samples)
```

```
Chosen samples of wholesale customers dataset:
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	13265	1196	4221	6404	507	1788
1	3521	1099	1997	1796	173	995
2	40254	640	3600	1042	436	18

Question 1

Consider the total purchase cost of each product category and the statistical description of the dataset above for your sample customers.
What kind of establishment (customer) could each of the three samples you've chosen represent?
Hint: Examples of establishments include places like markets, cafes, and retailers, among many others. Avoid using names for establishments, such as saying *"McDonalds"* when describing a sample customer as a restaurant.

Answer: I will assume that the monetary unit is Euros and using the data description from the Github repository:

```
              (Minimum, Maximum, Mean, Std. Deviation)
```

FRESH (3, 112151, 12000.30, 12647.329)

MILK (55, 73498, 5796.27, 7380.377)

GROCERY (3, 92780, 7951.28, 9503.163)

FROZEN (25, 60869, 3071.93, 4854.673)

DETERGENTS_PAPER (3, 40827, 2881.49, 4767.854)

DELICATESSEN (3, 47943, 1524.87, 2820.106)

I calculated in Excel the proportion of each spending category on the total spendign for each sample/customer. This would give me a better sense on what the main spending categories are, hence, allowing me to use them as a proxy to identify their business purpose.

In this case, all the samples I selected have a very high percentage of their spending in the Fresh category.

Sample 0:

Fresh Milk Grocery Frozen Deterg_Paper Delicatessen 48% 4% 15% 23% 2% 7% This sample has its major spending on categories FRESH, GROCERY and FROZEN (87%). From the statistical description, I can also see this place is above the 50% percentile but close to the 75 percentile for FRESH, 25% for MILK, 50% for GROCERY, and above 75% percentile for FROZEN and DELICATESSEN. Looking at this data and their relationships, this leads me to think that this is a large Restaurant or maybe a local chain of a few restaurants.

Sample 1:

Fresh Milk Grocery Frozen Deterg_Paper Delicatessen 37% 11% 21% 19% 2% 10%

This sample has a relatively balanced spending among categories when compared to my other two samples which leads me to think that this business uses/sells all of them. Also, looking at the statistical description for the categories I see this business is below the 25% percentile (for FRESH, MILK and GROCERY) so it is not a large busines from our dataset's point of view, however, FROZEN and Delicatessen were around their 50% percentile and these two categories comprise 29% of the spending of this business. Considering all the information: relative high spending in FRESH produce, GROCERY and FROZEN, very high spending in MILK and Delicatessen, this leads me to think that this may be a small delicatessen cafe that also offers food to serve.

Sample 2:

Fresh Milk Grocery Frozen Deterg_Paper Delicatessen 88% 1% 8% 2% 1% 0%

The relative spending for this particular sample is mainly in FRESH and GROCERY with a 96% of its annual spending. Comparing its total anual spending with the statistical description, I can see this business is well above the 75% percentile for FRESH, below 25 percentile for MILK, GROCERY, and FROZEN, and DELICATESSEN

and detergents slightly above 25 percentile.

Looking at this data, This leads me to think that this business must be a large fresh market.

Implementation: Feature Relevance

One interesting thought to consider is if one (or more) of the six product categories is actually relevant for understanding customer purchasing. That is to say, is it possible to determine whether customers purchasing some amount of one category of products will necessarily purchase some proportional amount of another category of products? We can make this determination quite easily by training a supervised regression learner on a subset of the data with one feature removed, and then score how well that model can predict the removed feature.

In the code block below, you will need to implement the following:

- Assign `new_data` a copy of the data by removing a feature of your choice using the `DataFrame.drop` function.
- Use `sklearn.cross_validation.train_test_split` to split the dataset into training and testing sets.
  - Use the removed feature as your target label. Set a `test_size` of `0.25` and set a `random_state`.
- Import a decision tree regressor, set a `random_state`, and fit the learner to the training data.
- Report the prediction score of the testing set using the regressor's `score` function.

In [43]:
```
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeRegressor
# TODO: Make a copy of the DataFrame, using the 'drop' function to drop the gi
ven feature
new_data = data.drop(['Fresh'], axis = 1)
#display(new_data.describe())
labels = data[['Fresh']]
# TODO: Split the data into training and testing sets using the given feature
 as the target
X_train, X_test, y_train, y_test = train_test_split(new_data, labels, test_siz
e=0.25, random_state=42)

# TODO: Create a decision tree regressor and fit it to the training set
regressor = DecisionTreeRegressor(random_state=42)
regressor.fit(X_train, y_train)

# TODO: Report the score of the prediction using the testing set
score = regressor.score(X_test,y_test)
print "Score is: ",score
```

```
Score is:  -0.385749710204
```

Question 2

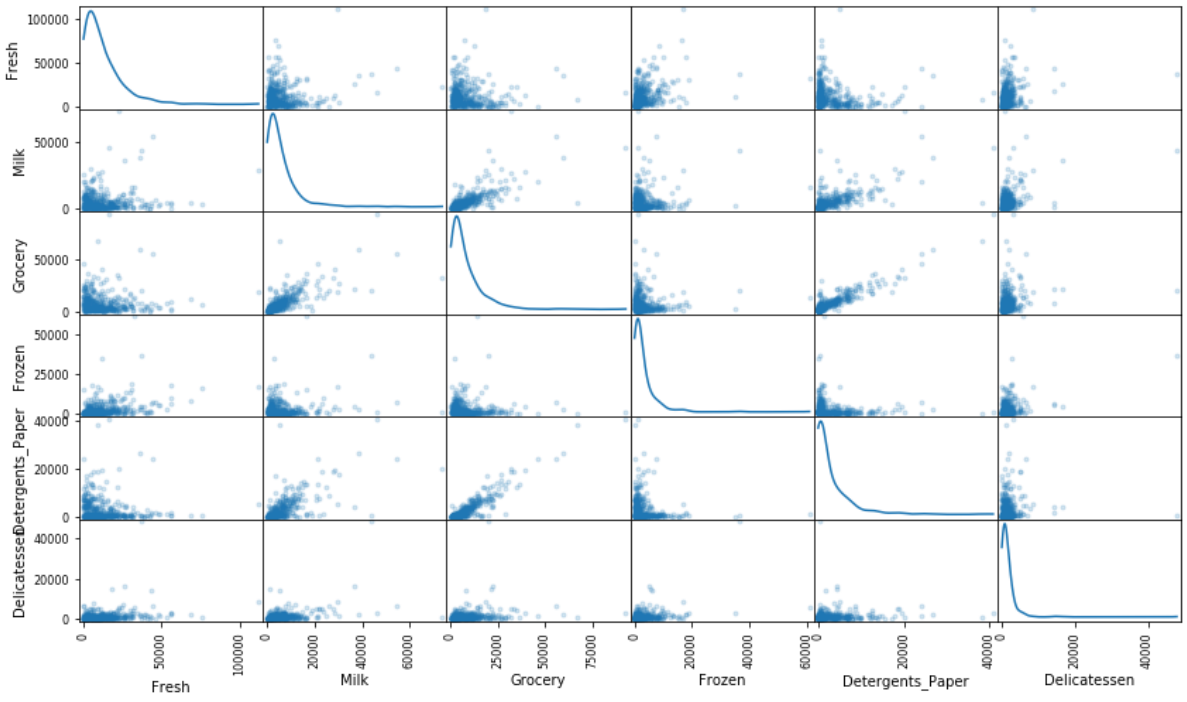
Which feature did you attempt to predict? What was the reported prediction score? Is this feature necessary for identifying customers' spending habits?
Hint: The coefficient of determination, `R^2`, is scored between 0 and 1, with 1 being a perfect fit. A negative `R^2` implies the model fails to fit the data.

Answer: I attempted 'FRESH'. The result was -.038 which means the model does not predict the data well without the use of this feature. Having a negative R^2 also means that this regression model does not have an intercept. This means this feature will be important(relevant) for the regression process.
https://stats.stackexchange.com/questions/183265/what-does-negative-r-squared-mean (https://stats.stackexchange.com/questions/183265/what-does-negative-r-squared-mean)

Visualize Feature Distributions

To get a better understanding of the dataset, we can construct a scatter matrix of each of the six product features present in the data. If you found that the feature you attempted to predict above is relevant for identifying a specific customer, then the scatter matrix below may not show any correlation between that feature and the others. Conversely, if you believe that feature is not relevant for identifying a specific customer, the scatter matrix might show a correlation between that feature and another feature in the data. Run the code block below to produce a scatter matrix.

In [44]:
```
# Produce a scatter matrix for each pair of features in the data
pd.scatter_matrix(data, alpha = 0.18, figsize = (14,8), diagonal = 'kde');
```



Question 3

Are there any pairs of features which exhibit some degree of correlation? Does this confirm or deny your suspicions about the relevance of the feature you attempted to predict? How is the data for those features distributed?
Hint: Is the data normally distributed? Where do most of the data points lie?

Answer:

Are there any pairs of features which exhibit some degree of correlation?

Yes, there are a few, for instance: Grocery/Milk, detergent/grocery, detergent/milk,

Does this confirm or deny your suspicions about the relevance of the feature you attempted to predict?

The feature I selected, Fresh, does not show a clear correlation with any other feature in the set, hence, it would still be an important feature to use for regression purposes.

How is the data for those features distributed?

Since there's no possibility to obtain negative spending, the distributions observed do not show a normal distribution (Gaussian curve). Distributions shapes for any of the features have their maximum frequency close or around low spending amounts with a sharp fall once spending increases.

Data Preprocessing

In this section, you will preprocess the data to create a better representation of customers by performing a scaling on the data and detecting (and optionally removing) outliers. Preprocessing data is often times a critical step in assuring that results you obtain from your analysis are significant and meaningful.

Implementation: Feature Scaling

If data is not normally distributed, especially if the mean and median vary significantly (indicating a large skew), it is most often appropriate (http://econbrowser.com/archives/2014/02/use-of-logarithms-in-economics) to apply a non-linear scaling — particularly for financial data. One way to achieve this scaling is by using a Box-Cox test (http://scipy.github.io/devdocs/generated/scipy.stats.boxcox.html), which calculates the best power transformation of the data that reduces skewness. A simpler approach which can work in most cases would be applying the natural logarithm.

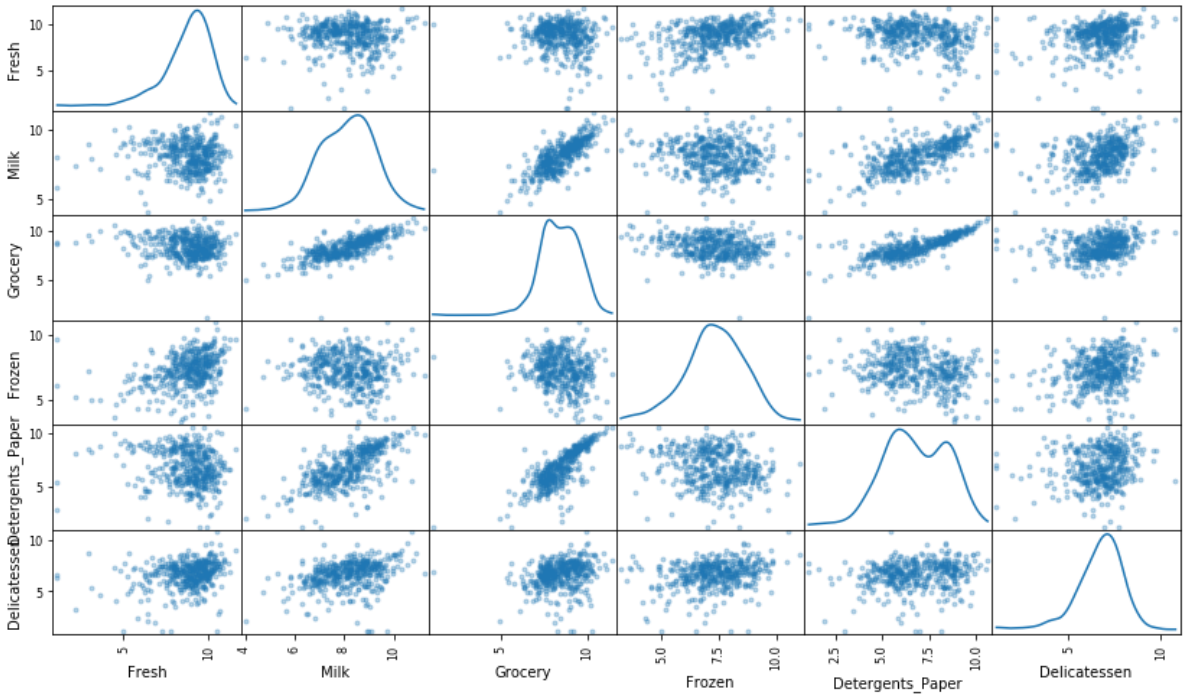
In the code block below, you will need to implement the following:

- Assign a copy of the data to `log_data` after applying logarithmic scaling. Use the `np.log` function for this.
- Assign a copy of the sample data to `log_samples` after applying logarithmic scaling. Again, use `np.log`.

In [45]:
```
# TODO: Scale the data using the natural logarithm
log_data = np.log(data)

# TODO: Scale the sample data using the natural logarithm
log_samples = np.log(samples)

# Produce a scatter matrix for each pair of newly-transformed features
pd.scatter_matrix(log_data, alpha = 0.3, figsize = (14,8), diagonal = 'kde');
```



Observation

After applying a natural logarithm scaling to the data, the distribution of each feature should appear much more normal. For any pairs of features you may have identified earlier as being correlated, observe here whether that correlation is still present (and whether it is now stronger or weaker than before).

Run the code below to see how the sample data has changed after having the natural logarithm applied to it.

In [46]:
```
# Display the log-transformed sample data
display(log_samples)
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	9.492884	7.086738	8.347827	8.764678	6.228511	7.488853
1	8.166500	7.002156	7.599401	7.493317	5.153292	6.902743
2	10.602965	6.461468	8.188689	6.948897	6.077642	2.890372

Implementation: Outlier Detection

Detecting outliers in the data is extremely important in the data preprocessing step of any analysis. The presence of outliers can often skew results which take into consideration these data points. There are many "rules of thumb" for what constitutes an outlier in a dataset. Here, we will use Tukey's Method for identfying outliers (http://datapigtechnologies.com/blog/index.php/highlighting-outliers-in-your-data-with-the-tukey-method/): An *outlier step* is calculated as 1.5 times the interquartile range (IQR). A data point with a feature that is beyond an outlier step outside of the IQR for that feature is considered abnormal.

In the code block below, you will need to implement the following:

- Assign the value of the 25th percentile for the given feature to `Q1`. Use `np.percentile` for this.
- Assign the value of the 75th percentile for the given feature to `Q3`. Again, use `np.percentile`.
- Assign the calculation of an outlier step for the given feature to `step`.
- Optionally remove data points from the dataset by adding indices to the `outliers` list.

NOTE: If you choose to remove any outliers, ensure that the sample data does not contain any of these points! Once you have performed this implementation, the dataset will be stored in the variable `good_data`.

In [47]:
```
# For each feature find the data points with extreme high or low values
for feature in log_data.keys():

    # TODO: Calculate Q1 (25th percentile of the data) for the given feature
    Q1 = np.percentile(log_data[feature],25)

    # TODO: Calculate Q3 (75th percentile of the data) for the given feature
    Q3 = np.percentile(log_data[feature],75)

    # TODO: Use the interquartile range to calculate an outlier step (1.5 time
s the interquartile range)
    step = 1.5*(Q3-Q1)

    # Display the outliers
    print "Data points considered outliers for the feature '{}':".format(featu
re)
    display(log_data[~((log_data[feature] >= Q1 - step) & (log_data[feature] <
= Q3 + step))])

# OPTIONAL: Select the indices for data points you wish to remove
outliers  = [65,66,128,154,75]

# Remove the outliers, if any were specified
good_data = log_data.drop(log_data.index[outliers]).reset_index(drop = True)
```

Data points considered outliers for the feature 'Fresh':

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
65	4.442651	9.950323	10.732651	3.583519	10.095388	7.260523
66	2.197225	7.335634	8.911530	5.164786	8.151333	3.295837
81	5.389072	9.163249	9.575192	5.645447	8.964184	5.049856
95	1.098612	7.979339	8.740657	6.086775	5.407172	6.563856
96	3.135494	7.869402	9.001839	4.976734	8.262043	5.379897
128	4.941642	9.087834	8.248791	4.955827	6.967909	1.098612
171	5.298317	10.160530	9.894245	6.478510	9.079434	8.740337
193	5.192957	8.156223	9.917982	6.865891	8.633731	6.501290
218	2.890372	8.923191	9.629380	7.158514	8.475746	8.759669
304	5.081404	8.917311	10.117510	6.424869	9.374413	7.787382
305	5.493061	9.468001	9.088399	6.683361	8.271037	5.351858
338	1.098612	5.808142	8.856661	9.655090	2.708050	6.309918
353	4.762174	8.742574	9.961898	5.429346	9.069007	7.013016
355	5.247024	6.588926	7.606885	5.501258	5.214936	4.844187
357	3.610918	7.150701	10.011086	4.919981	8.816853	4.700480
412	4.574711	8.190077	9.425452	4.584967	7.996317	4.127134

Data points considered outliers for the feature 'Milk':

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
86	10.039983	11.205013	10.377047	6.894670	9.906981	6.805723
98	6.220590	4.718499	6.656727	6.796824	4.025352	4.882802
154	6.432940	4.007333	4.919981	4.317488	1.945910	2.079442
356	10.029503	4.897840	5.384495	8.057377	2.197225	6.306275

Data points considered outliers for the feature 'Grocery':

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
75	9.923192	7.036148	1.098612	8.390949	1.098612	6.882437
154	6.432940	4.007333	4.919981	4.317488	1.945910	2.079442

Data points considered outliers for the feature 'Frozen':

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
38	8.431853	9.663261	9.723703	3.496508	8.847360	6.070738
57	8.597297	9.203618	9.257892	3.637586	8.932213	7.156177
65	4.442651	9.950323	10.732651	3.583519	10.095388	7.260523
145	10.000569	9.034080	10.457143	3.737670	9.440738	8.396155
175	7.759187	8.967632	9.382106	3.951244	8.341887	7.436617
264	6.978214	9.177714	9.645041	4.110874	8.696176	7.142827
325	10.395650	9.728181	9.519735	11.016479	7.148346	8.632128
420	8.402007	8.569026	9.490015	3.218876	8.827321	7.239215
429	9.060331	7.467371	8.183118	3.850148	4.430817	7.824446
439	7.932721	7.437206	7.828038	4.174387	6.167516	3.951244

Data points considered outliers for the feature 'Detergents_Paper':

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
75	9.923192	7.036148	1.098612	8.390949	1.098612	6.882437
161	9.428190	6.291569	5.645447	6.995766	1.098612	7.711101

Data points considered outliers for the feature 'Delicatessen':

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
66	2.197225	7.335634	8.911530	5.164786	8.151333	3.295837
109	7.248504	9.724899	10.274568	6.511745	6.728629	1.098612
128	4.941642	9.087834	8.248791	4.955827	6.967909	1.098612
137	8.034955	8.997147	9.021840	6.493754	6.580639	3.583519
142	10.519646	8.875147	9.018332	8.004700	2.995732	1.098612
154	6.432940	4.007333	4.919981	4.317488	1.945910	2.079442
183	10.514529	10.690808	9.911952	10.505999	5.476464	10.777768
184	5.789960	6.822197	8.457443	4.304065	5.811141	2.397895
187	7.798933	8.987447	9.192075	8.743372	8.148735	1.098612
203	6.368187	6.529419	7.703459	6.150603	6.860664	2.890372
233	6.871091	8.513988	8.106515	6.842683	6.013715	1.945910
285	10.602965	6.461468	8.188689	6.948897	6.077642	2.890372
289	10.663966	5.655992	6.154858	7.235619	3.465736	3.091042
343	7.431892	8.848509	10.177932	7.283448	9.646593	3.610918

Question 4

Are there any data points considered outliers for more than one feature based on the definition above? Should these data points be removed from the dataset? If any data points were added to the `outliers` list to be removed, explain why.

Answer: Yes, there are several points cosidered outliers for each of the features. The features with more outliers are FRESH, FROZEN and DELICATESSEN. For this exercise, I have removed the common outlier points ocurring more than 2 times since avoiding these, I avoid skewing my model in later stages. I could have also removed extreme outliers for teh same reason, but for this exercise, I'll maintain the criteria of removal to common outliers between features (more than 2)

Feature Transformation

In this section you will use principal component analysis (PCA) to draw conclusions about the underlying structure of the wholesale customer data. Since using PCA on a dataset calculates the dimensions which best maximize variance, we will find which compound combinations of features best describe customers.

Implementation: PCA

Now that the data has been scaled to a more normal distribution and has had any necessary outliers removed, we can now apply PCA to the `good_data` to discover which dimensions about the data best maximize the variance of features involved. In addition to finding these dimensions, PCA will also report the *explained variance ratio* of each dimension — how much variance within the data is explained by that dimension alone. Note that a component (dimension) from PCA can be considered a new "feature" of the space, however it is a composition of the original features present in the data.

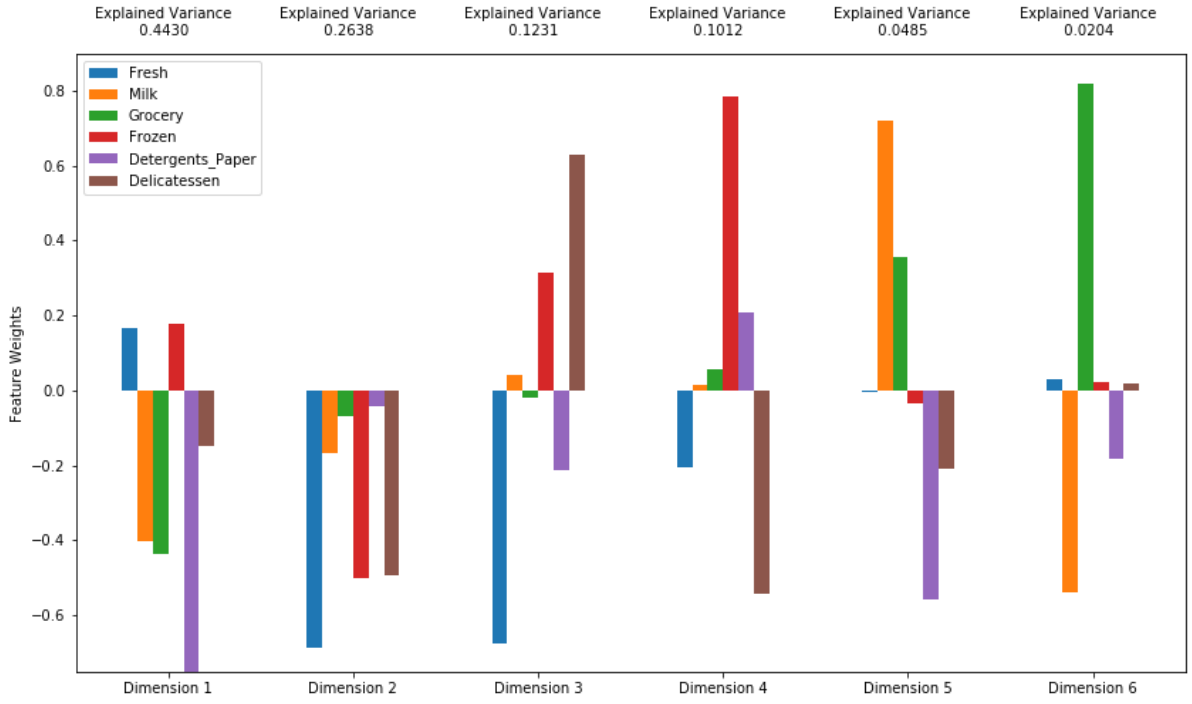
In the code block below, you will need to implement the following:

- Import `sklearn.decomposition.PCA` and assign the results of fitting PCA in six dimensions with `good_data` to `pca`.
- Apply a PCA transformation of `log_samples` using `pca.transform`, and assign the results to `pca_samples`.

In [48]:
```
from sklearn.decomposition import PCA
# TODO: Apply PCA by fitting the good data with the same number of dimensions
 as features
pca = PCA(n_components=6).fit(good_data)
print(pca.explained_variance_ratio_)
# TODO: Transform log_samples using the PCA fit above
pca_samples = pca.transform(log_samples)
#display(log_samples)

# Generate PCA results plot
pca_results = vs.pca_results(good_data, pca)
#pca_sample_results = vs.pca_results(samples, pca)
```

```
[ 0.44302505  0.26379218  0.1230638   0.10120908  0.04850196  0.02040793]
```



Question 5

*How much variance in the data is explained **in total** by the first and second principal component? What about the first four principal components? Using the visualization provided above, discuss what the first four dimensions best represent in terms of customer spending.*

Hint: A positive increase in a specific dimension corresponds with an *increase* of the *positive-weighted* features and a *decrease* of the *negative-weighted* features. The rate of increase or decrease is based on the individual feature weights.

Answer:

The first two components explain over 70% of the variance in the data. The first four components explain 93% of the variance of the data.

I used the following links to understand how to explain a PCA result:
https://onlinecourses.science.psu.edu/stat505/node/54 (https://onlinecourses.science.psu.edu/stat505/node/54)
http://webspace.ship.edu/pgmarr/Geo441/Lectures/Lec%2017%20-%20Principal%20Component%20Analysis.pdf (http://webspace.ship.edu/pgmarr/Geo441/Lectures/Lec%2017%20-%20Principal%20Component%20Analysis.pdf)

From the Forums: https://discussions.udacity.com/t/question-5-what-the-first-four-dimensions-best-represent-in-terms-of-customer-spending/177181/5 (https://discussions.udacity.com/t/question-5-what-the-first-four-dimensions-best-represent-in-terms-of-customer-spending/177181/5)

https://discussions.udacity.com/t/q5-observation-consistent-with-with-your-initial-interpretation-of-the-sample-points/171396/2 (https://discussions.udacity.com/t/q5-observation-consistent-with-with-your-initial-interpretation-of-the-sample-points/171396/2)

What the first four dimensions best represent in terms of customer spending?

For this analysis I'll assume correlations higher than 0.5 or lower than -0.5 are considered as strong correlations.

PCA is used to decompose a multivariate dataset in a set of successive orthogonal components that explain a maximum amount of the variance (http://scikit-learn.org/stable/modules/decomposition.html#pca (http://scikit-learn.org/stable/modules/decomposition.html#pca)). Principal componentes (PCs) are the linear combinations of the individual features that form my data set to create "new" features, generally less than the original feature set, that also represent our dataset well (https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues (https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues)).

PCA 1

This dimension explains 44.3% of the variability, from which DETERGENT is highly correlated with this dimension, followed by GROCERY and MILK (and on the same negative direction). From this it is observed that the linear combination of these features gives more weight to non-fresh/food related items

PCA 2

In PCA 2, it's seen how FRESH is strongly negatively correlated to it (taking most of the weight for the linear combination), followed by FROZEN and DELICATESSEN who are also negatively correlated in lessen terms; moreover, in PC 2 it can be observed how more food-related items take more relevance than non-food ones.

PCA 3

This dimension explains slightly over 12% of the variability, hence, much lower than the first two. In this case though, two variables are strongly correlated FRESH (negatively) and DELICATESSEN (positively) followed by FROZEN. It is worth noting that the weight for FRESH is very similar to the one in PC 2 and, from a type of

product (feature) point of view, even though FROZEN and DELICATESSEN have different signs on their weights, PC 3 is in some ways related to PC 2 since they are driven by the same features.

PCA 4

This dimension explains slightly over 10% of the variability, hence, much lower than the first two. In this case two variables are strongly correlated FROZEN (positively) and DELICATESSEN(negatively) followed by FRESH. Once again, it is observed the same "relevant" feature set, though with different weights for each feature, as in PCA 2 and PCA 3; this is greater weights in food-related products.

I finally see it!

Without entering in the very details of what each PC may be or signify, it is clear that PC 2, PC 3 and PC 4 are correlated and are mainly driven by food-related items. In contrast, PC 1 while it has some grocery type (and MILK) it is mainly driven by non-food items. This opens the case to state we might be in the presence of two distinct customer types or segments.

Observation

Run the code below to see how the log-transformed sample data has changed after having a PCA transformation applied to it in six dimensions. Observe the numerical value for the first four dimensions of the sample points. Consider if this is consistent with your initial interpretation of the sample points.

In [49]:
```
# Display sample log-data after having a PCA transformation applied
#print pca_results.index.values
#display(np.round(pca_samples, 4))
display(samples)
display(pd.DataFrame(np.round(pca_samples, 4), columns = pca_results.index.values))
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	13265	1196	4221	6404	507	1788
1	3521	1099	1997	1796	173	995
2	40254	640	3600	1042	436	18

	Dimension 1	Dimension 2	Dimension 3	Dimension 4	Dimension 5	Dimension 6
0	1.1553	-1.4052	0.5422	0.4127	-0.6865	0.6409
1	1.9642	0.5433	0.9106	-0.2652	-0.2450	0.1923
2	2.1408	1.1368	-3.6622	1.2051	-0.0894	0.7799

Implementation: Dimensionality Reduction

When using principal component analysis, one of the main goals is to reduce the dimensionality of the data — in effect, reducing the complexity of the problem. Dimensionality reduction comes at a cost: Fewer dimensions used implies less of the total variance in the data is being explained. Because of this, the *cumulative explained variance ratio* is extremely important for knowing how many dimensions are necessary for the problem. Additionally, if a signifiant amount of variance is explained by only two or three dimensions, the reduced data can be visualized afterwards.

In the code block below, you will need to implement the following:

- Assign the results of fitting PCA in two dimensions with `good_data` to `pca`.
- Apply a PCA transformation of `good_data` using `pca.transform`, and assign the results to `reduced_data`.
- Apply a PCA transformation of `log_samples` using `pca.transform`, and assign the results to `pca_samples`.

In [50]:
```
# TODO: Apply PCA by fitting the good data with only two dimensions
pca = PCA(n_components=2).fit(good_data)

# TODO: Transform the good data using the PCA fit above
reduced_data = pca.transform(good_data)

# TODO: Transform log_samples using the PCA fit above
pca_samples = pca.transform(log_samples)

# Create a DataFrame for the reduced data
reduced_data = pd.DataFrame(reduced_data, columns = ['Dimension 1', 'Dimension
 2'])
```

Observation

Run the code below to see how the log-transformed sample data has changed after having a PCA transformation applied to it using only two dimensions. Observe how the values for the first two dimensions remains unchanged when compared to a PCA transformation in six dimensions.

In [51]:
```
# Display sample log-data after applying PCA transformation in two dimensions
display(pd.DataFrame(np.round(pca_samples, 4), columns = ['Dimension 1', 'Dime
nsion 2']))
```

	Dimension 1	Dimension 2
0	1.1553	-1.4052
1	1.9642	0.5433
2	2.1408	1.1368

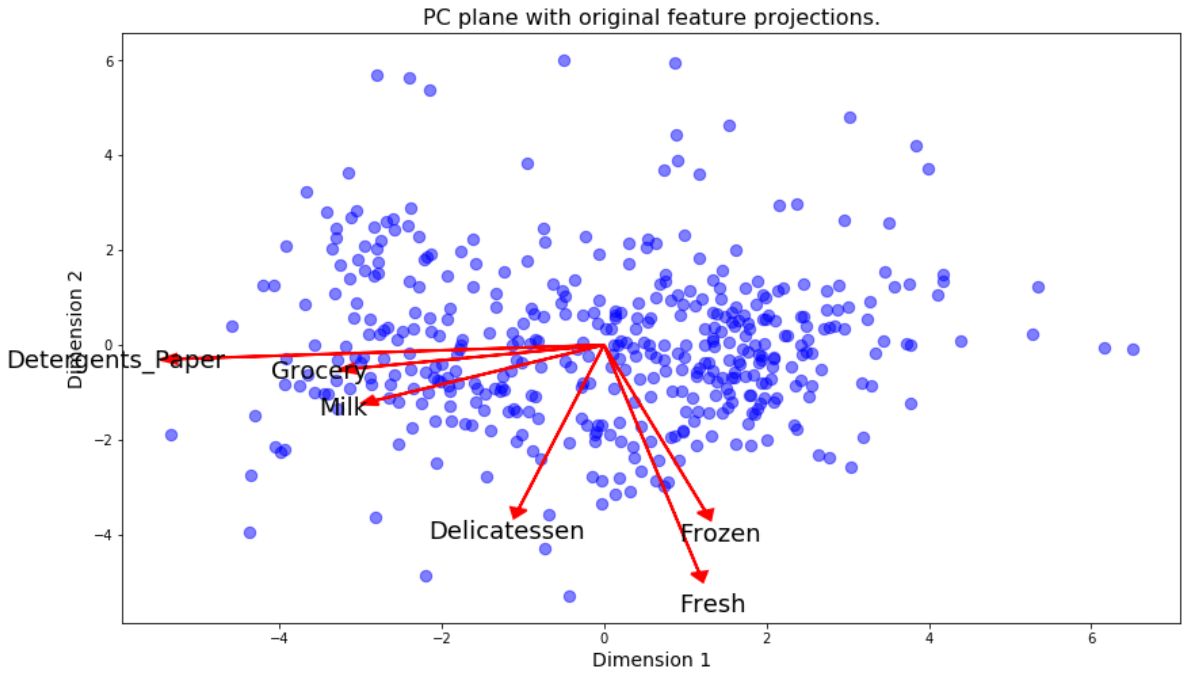
Visualizing a Biplot

A biplot is a scatterplot where each data point is represented by its scores along the principal components. The axes are the principal components (in this case `Dimension 1` and `Dimension 2`). In addition, the biplot shows the projection of the original features along the components. A biplot can help us interpret the reduced dimensions of the data, and discover relationships between the principal components and original features.

Run the code cell below to produce a biplot of the reduced-dimension data.

In [52]:
```
# Create a biplot
vs.biplot(good_data, reduced_data, pca)
```

Out[52]: `<matplotlib.axes._subplots.AxesSubplot at 0xee924a8>`



Observation

Once we have the original feature projections (in red), it is easier to interpret the relative position of each data point in the scatterplot. For instance, a point the lower right corner of the figure will likely correspond to a customer that spends a lot on `'Milk'`, `'Grocery'` and `'Detergents_Paper'`, but not so much on the other product categories.

From the biplot, which of the original features are most strongly correlated with the first component? What about those that are associated with the second component? Do these observations agree with the pca_results plot you obtained earlier?

Clustering

In this section, you will choose to use either a K-Means clustering algorithm or a Gaussian Mixture Model clustering algorithm to identify the various customer segments hidden in the data. You will then recover specific data points from the clusters to understand their significance by transforming them back into their original dimension and scale.

Question 6

What are the advantages to using a K-Means clustering algorithm? What are the advantages to using a Gaussian Mixture Model clustering algorithm? Given your observations about the wholesale customer data so far, which of the two algorithms will you use and why?

Answer: U used the following links to better support my answer: https://www.quora.com/What-is-the-difference-between-K-means-and-the-mixture-model-of-Gaussian (https://www.quora.com/What-is-the-difference-between-K-means-and-the-mixture-model-of-Gaussian)
http://www.cse.psu.edu/~rtc12/CSE586Spring2010/lectures/cse586gmmemPart1_6pp.pdf (http://www.cse.psu.edu/~rtc12/CSE586Spring2010/lectures/cse586gmmemPart1_6pp.pdf)
https://www.quora.com/What-are-the-advantages-to-using-a-Gaussian-Mixture-Model-clustering-algorithm (https://www.quora.com/What-are-the-advantages-to-using-a-Gaussian-Mixture-Model-clustering-algorithm)

http://scikit-learn.org/stable/modules/mixture.html (http://scikit-learn.org/stable/modules/mixture.html) http://scikit-learn.org/stable/modules/clustering.html#k-means (http://scikit-learn.org/stable/modules/clustering.html#k-means) https://stats.stackexchange.com/questions/58855/why-do-we-use-k-means-instead-of-other-algorithms (https://stats.stackexchange.com/questions/58855/why-do-we-use-k-means-instead-of-other-algorithms)

K-Means clustering algorithm

Characteristics and Con's:

- Requires pre knowledge of K count
- Uses Hard Assigment
- Does not work well on elongated clusters, it works better on spherical clusters
- It may converge to a local minimum so initial centroids should be randomized several times to avoid local minimums
- May be affected by high dimensionality, hence, it may need PCA.
- Sensitive to Outlies and Noise

  Pro's: https://www.quora.com/What-are-the-advantages-of-K-Means-clustering (https://www.quora.com/What-are-the-advantages-of-K-Means-clustering) 1)Practically, it works well even when some assumptions are broken 2)It is simple and easy to implement 3)It is easy to interpret the clustering results 4)It is fast and efficient in terms of computational cost

Gaussian Mixture Model clustering algorithm

https://www.quora.com/What-are-the-advantages-to-using-a-Gaussian-Mixture-Model-clustering-algorithm (https://www.quora.com/What-are-the-advantages-to-using-a-Gaussian-Mixture-Model-clustering-algorithm)

Characteristics and Con's:

- Requires pre knowledge of K count
- Uses Soft assignment
- It is the fastest algorithm for learning mixture models
- It helps express the uncertainty for whicha a point belongs to a group
- It can represent multi-modal datasets using multiple unimodal gaussian-like clusters

Pro's:

- As expressed in the lessons, it allows to provide a sense of truth to the data since there always be a probability (even if very slow) that the point may be of another group, in other words, it allows point to belong to different groups at different membership levels based on tose calculated probabilities.
- It is more flexible by allowing clusters of shapes other than spheres

Given your observations about the wholesale customer data so far, which of the two algorithms will you use and why?

Even though there's a slight cluster structure on the righ side of the biplot (darker blue points), I can see there is no clear definition -decision line- of groups or clusters in the data nor I can clearly see their clusters' shape (spherical vs other), hence, making it more diffult to make a definitive decision about number of clusters for the k-means algorithm. Because of this, I'll select the GM Model because it will give me more flexibility including the cluster shapes, yet relatively simple and fast to execute.

Implementation: Creating Clusters

Depending on the problem, the number of clusters that you expect to be in the data may already be known. When the number of clusters is not known *a priori*, there is no guarantee that a given number of clusters best segments the data, since it is unclear what structure exists in the data — if any. However, we can quantify the "goodness" of a clustering by calculating each data point's *silhouette coefficient*. The silhouette coefficient (http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html) for a data point measures how similar it is to its assigned cluster from -1 (dissimilar) to 1 (similar). Calculating the *mean* silhouette coefficient provides for a simple scoring method of a given clustering.

In the code block below, you will need to implement the following:

- Fit a clustering algorithm to the `reduced_data` and assign it to `clusterer`.
- Predict the cluster for each data point in `reduced_data` using `clusterer.predict` and assign them to `preds`.
- Find the cluster centers using the algorithm's respective attribute and assign them to `centers`.
- Predict the cluster for each sample data point in `pca_samples` and assign them `sample_preds`.
- Import `sklearn.metrics.silhouette_score` and calculate the silhouette score of `reduced_data` against `preds`.
    - Assign the silhouette score to `score` and print the result.

In [53]:
```
from sklearn.mixture import GaussianMixture
from sklearn.metrics import silhouette_score
# TODO: Apply your clustering algorithm of choice to the reduced data
clusterer = GaussianMixture(n_components=2, covariance_type='full',random_stat
e = 42).fit(reduced_data)

# TODO: Predict the cluster for each data point
preds = clusterer.predict(reduced_data)

# TODO: Find the cluster centers
centers = clusterer.means_

# TODO: Predict the cluster for each transformed sample data point
sample_preds = clusterer.predict(pca_samples)

# TODO: Calculate the mean silhouette coefficient for the number of clusters c
hosen
score = silhouette_score(reduced_data,preds)

print score
```

```
0.421916846463
```

Question 7

Report the silhouette score for several cluster numbers you tried. Of these, which number of clusters has the best silhouette score?

Answer:

Clusters = 2 : 0.421916846463

Clusters = 3 : 0.404248738241

Clusters = 4 : 0.293269564847

Clusters = 5 : 0.300456388725

Clusters = 6 : 0.326139450471

Clusters = 7 : 0.324227205384

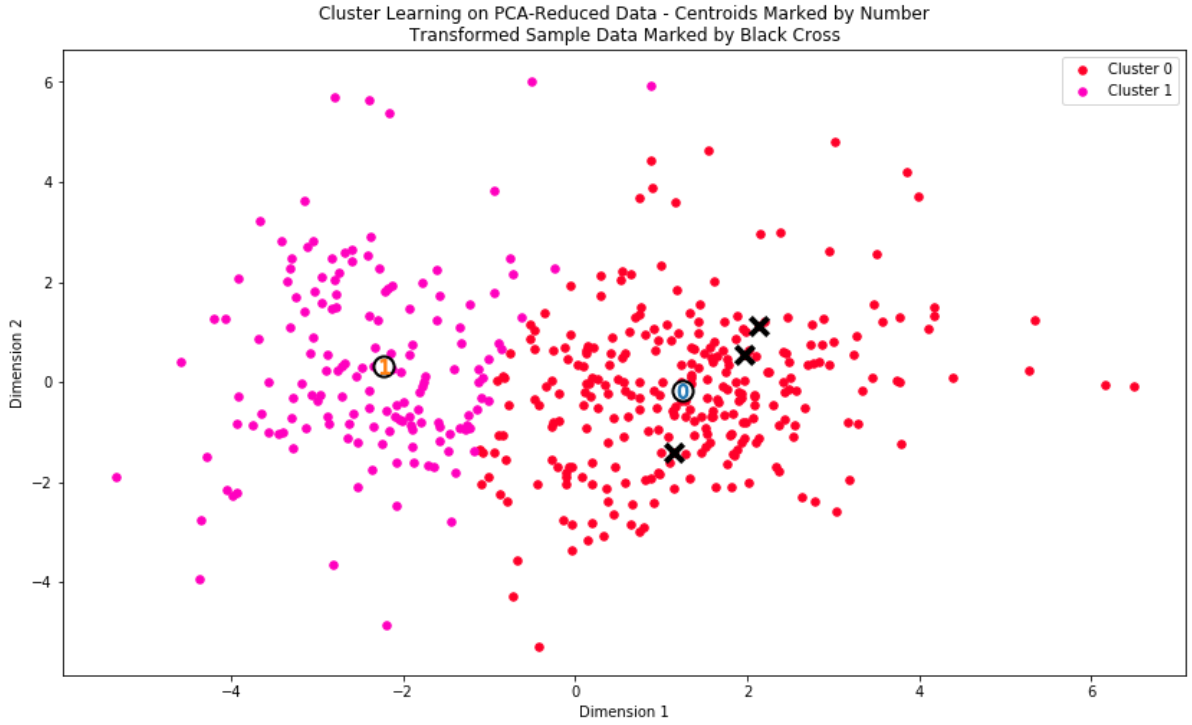
Clusters = 8 : 0.296476656397

From the results obtained, I see that choosing ***two clusters*** provides the best silouhtte score, hence, improving my chances to a better clustering of my data.

Cluster Visualization

Once you've chosen the optimal number of clusters for your clustering algorithm using the scoring metric above, you can now visualize the results by executing the code block below. Note that, for experimentation purposes, you are welcome to adjust the number of clusters for your clustering algorithm to see various visualizations. The final visualization provided should, however, correspond with the optimal number of clusters.

In [54]:
```
# Display the results of the clustering from implementation
vs.cluster_results(reduced_data, preds, centers, pca_samples)
```



Implementation: Data Recovery

Each cluster present in the visualization above has a central point. These centers (or means) are not specifically data points from the data, but rather the *averages* of all the data points predicted in the respective clusters. For the problem of creating customer segments, a cluster's center point corresponds to *the average customer of that segment*. Since the data is currently reduced in dimension and scaled by a logarithm, we can recover the representative customer spending from these data points by applying the inverse transformations.

In the code block below, you will need to implement the following:

- Apply the inverse transform to `centers` using `pca.inverse_transform` and assign the new centers to `log_centers`.
- Apply the inverse function of `np.log` to `log_centers` using `np.exp` and assign the true centers to `true_centers`.

In [55]:
```
# TODO: Inverse transform the centers
log_centers = pca.inverse_transform(centers)

# TODO: Exponentiate the centers
true_centers = np.exp(log_centers)
#display(true_centers)
# Display the true centers
segments = ['Segment {}'.format(i) for i in range(0,len(centers))]
true_centers = pd.DataFrame(np.round(true_centers), columns = data.keys())
true_centers.index = segments
display(true_centers)
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
Segment 0	8953.0	2114.0	2765.0	2075.0	353.0	732.0
Segment 1	3552.0	7837.0	12219.0	870.0	4696.0	962.0

Question 8

Consider the total purchase cost of each product category for the representative data points above, and reference the statistical description of the dataset at the beginning of this project. *What set of establishments could each of the customer segments represent?*
Hint: A customer who is assigned to `'Cluster X'` should best identify with the establishments represented by the feature set of `'Segment X'`.

Answer:

Segment 0:

This customer spends 81% of its spending in FRESH, GROCERY and MILK, or in general, in food related products. In terms of the statistical description, this business shows:

- 50+% percentile for FRESH
- 25% percentile for MILK,
- 25+% percentile for GROCERY and
- 50+% percentile for FROZEN
- 25+% percentile for DETERGENTS
- 25+% percentile for DELICATESSEN

, hence, I would consider this customer as a small- to mid-sized business.

Considering the above, large spending in food, frozen items and detergents, I'd lean to think that this customer could be related to businesses with heavy use of food (i.e. restaurants or food related businesses)

Segment 1:

This customer has a very high spending of detergents (16%)along with the majority of its spending around FRESH, MILK and GROCERY (78%). In terms of the statistical description, this business shows:

- 25+% percentile for FRESH
- 75+% percentile for MILK,
- 75+% percentile for GROCERY and
- 25+% percentile for FROZEN
- 75+% percentile for DETERGENTS
- 50+% percentile for DELICATESSEN Having said this, looking at the spending distribution is clear that this type of business does a heavy use of GROCERY, MILK, DETERGENTS, followed by DELICATESSEN, FRESH and FROZEN. It also has a very large Total spending amount, hence, it makes me lean to a sort of Grocery Store or alike.

Question 9

*For each sample point, which customer segment from **Question 8** best represents it? Are the predictions for each sample point consistent with this?*

Run the code block below to find which cluster each sample point is predicted to be.

In [56]:
```
# Display the predictions
for i, pred in enumerate(sample_preds):
    print "Sample point", i, "predicted to be in Cluster", pred
```

```
Sample point 0 predicted to be in Cluster 0
Sample point 1 predicted to be in Cluster 0
Sample point 2 predicted to be in Cluster 0
```

Answer:

The predictions observed do align with the segments results from Q8. In particular, one common characteristic was the lack of large spending in DETERGENTS for any of the samples. This also aligns with my initial classification of the samples on Q1; this is, basically, **food related business with different sizes.**

Conclusion

In this final section, you will investigate ways that you can make use of the clustered data. First, you will consider how the different groups of customers, the ***customer segments***, may be affected differently by a specific delivery scheme. Next, you will consider how giving a label to each customer (which *segment* that customer belongs to) can provide for additional features about the customer data. Finally, you will compare the ***customer segments*** to a hidden variable present in the data, to see whether the clustering identified certain relationships.

Question 10

Companies will often run A/B tests (https://en.wikipedia.org/wiki/A/B_testing) when making small changes to their products or services to determine whether making that change will affect its customers positively or negatively. The wholesale distributor is considering changing its delivery service from currently 5 days a week to 3 days a week. However, the distributor will only make this change in delivery service for customers that react positively. *How can the wholesale distributor use the customer segments to determine which customers, if any, would react positively to the change in delivery service?*
Hint: Can we assume the change affects all customers equally? How can we determine which group of customers it affects the most?

Answer: I used this site to get more information on A/B Testing:

https://vwo.com/ab-testing/ (https://vwo.com/ab-testing/)

https://www.optimizely.com/ab-testing/ (https://www.optimizely.com/ab-testing/)

I'd recommend the wholesaler to:

1) Select a random sample of customers from each customer segment

2) Each customer segment's sample must be distributed so 50% is assigned the new delivery method, and the other 50% maintains the regular 5 days schedule (so it serves as a control group for the customer segment in question)

3) Then evaluate on the selected trial A/B customers (on each customer segment) their impressions over the new delivery scheme.

4) Given the outcome of the A/B results for each customer segment, the wholesaler could make a decision on whether to stay on the regular schedule for one of the groups or even on both groups if none of the groups received the change positively.

Question 11

Additional structure is derived from originally unlabeled data when using clustering techniques. Since each customer has a ***customer segment*** it best identifies with (depending on the clustering algorithm applied), we can consider *'customer segment'* as an **engineered feature** for the data. Assume the wholesale distributor recently acquired ten new customers and each provided estimates for anticipated annual spending of each product category. Knowing these estimates, the wholesale distributor wants to classify each new customer to a ***customer segment*** to determine the most appropriate delivery service.
How can the wholesale distributor label the new customers using only their estimated product spending and the ***customer segment*** *data?*
Hint: A supervised learner could be used to train on the original customers. What would be the target variable?

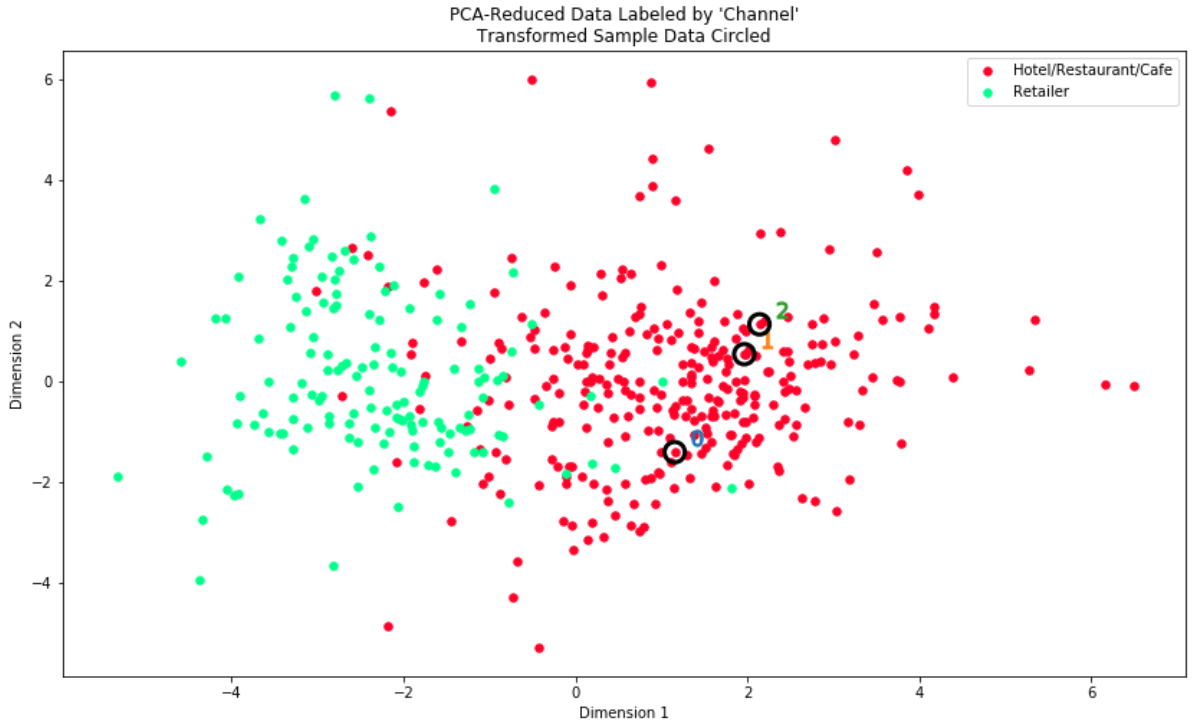
Answer: In this case, we have already trained our clustering model using GMM. We'd need to covert (apply log) to our new customer data, predict the clusters using the "pca" **clusterer**.predict function (applied to the new customers'data) so as to obtain the predicted segments for these businesses. With this info, the wholesaler cold make an informed decision on which delivery scheme shall be used for these new 10 customers (given that A/B tests have been performed, assessed and the wholesaler has defined path for delivery for each customer segment).

Visualizing Underlying Distributions

At the beginning of this project, it was discussed that the `'Channel'` and `'Region'` features would be excluded from the dataset so that the customer product categories were emphasized in the analysis. By reintroducing the `'Channel'` feature to the dataset, an interesting structure emerges when considering the same PCA dimensionality reduction applied earlier to the original dataset.

Run the code block below to see how each data point is labeled either `'HoReCa'` (Hotel/Restaurant/Cafe) or `'Retail'` the reduced space. In addition, you will find the sample points are circled in the plot, which will identify their labeling.

In [57]:
```
# Display the clustering results based on 'Channel' data
vs.channel_results(reduced_data, outliers, pca_samples)
```



Question 12

How well does the clustering algorithm and number of clusters you've chosen compare to this underlying distribution of Hotel/Restaurant/Cafe customers to Retailer customers? Are there customer segments that would be classified as purely 'Retailers' or 'Hotels/Restaurants/Cafes' by this distribution? Would you consider these classifications as consistent with your previous definition of the customer segments?

Answer:

How well does the clustering algorithm and number of clusters you've chosen compare to this underlying distribution of Hotel/Restaurant/Cafe customers to Retailer customers? The number of clusters in the data match with the optimal number of clusters my clustering algorithm found, this is, two groups or clusters.

Are there customer segments that would be classified as purely 'Retailers' or 'Hotels/Restaurants/Cafes' by this distribution? Looking at the previous scatter plot in Q12, there are certainly points that clearly lean towards one or the other cluster (Ho/Res/Ca or Ret) at the extreme ends of Dimension 1, however, it is important to note that the cutoff point (decision line) between these two clusters is not clearly defined. This can be seen, for instance, between values -2 and 1 of the Dimension 1, where most of the points are red, BUT there are still several in green color. Similarly, some red points reach very low Dim. 1 values (~-3) which are in the "green" area. This is to say, while many points could be "clearly" classified, there are still regions where some uncertainty exists.

Would you consider these classifications as consistent with your previous definition of the customer segments? The type of business obtained using the 'Channel' feature does also align with the cluster insights I had gathered in previous questions (Q8), this is, 'Ho/Res/Ca' aligns with my "food-type" businesses and 'Retailer" aligns with my "grocery-type" business.

> **Note**: Once you have completed all of the code implementations and successfully answered each question above, you may finalize your work by exporting the iPython Notebook as an HTML document. You can do this by using the menu above and navigating to **File -> Download as -> HTML (.html)**. Include the finished document along with this notebook as your submission.

In [ ]: