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 [1]:
```
# 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 DataFrames

# 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.".for
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 [2]:
```
# Display a description of the dataset
display(data.describe())
```

	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	1524.870455
std	12647.328865	7380.377175	9503.162829	4854.673333	4767.854448	2820.105937
min	3.000000	55.000000	3.000000	25.000000	3.000000	3.000000
25%	3127.750000	1533.000000	2153.000000	742.250000	256.750000	408.250000
50%	8504.000000	3627.000000	4755.500000	1526.000000	816.500000	965.500000
75%	16933.750000	7190.250000	10655.750000	3554.250000	3922.000000	1820.250000
max	112151.000000	73498.000000	92780.000000	60869.000000	40827.000000	47943.000000

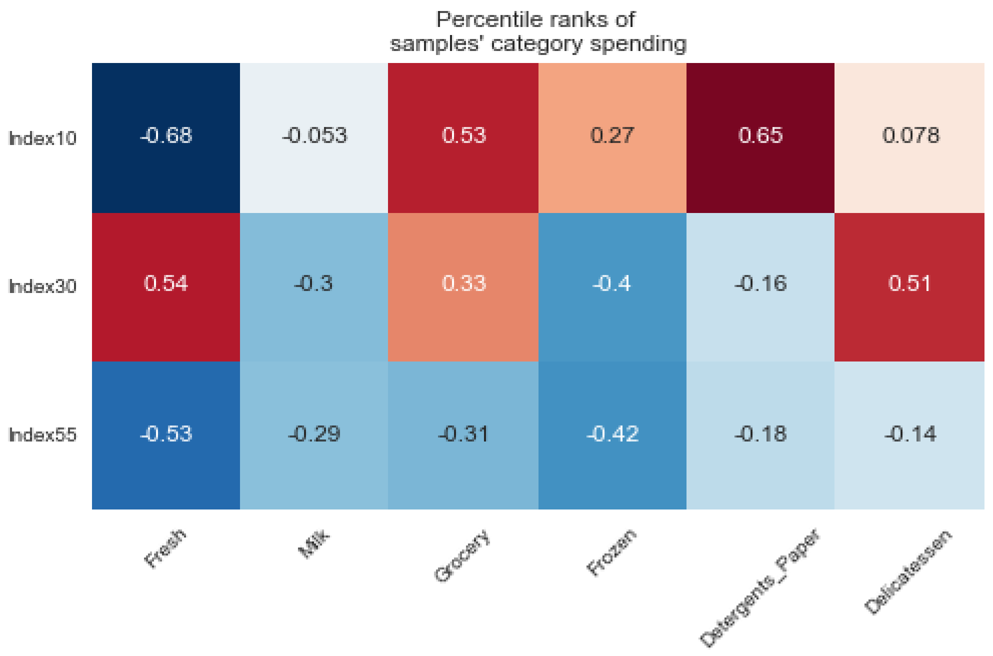
Implementation: Selecting Samples

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 [3]:
```
# TODO: Select three indices of your choice you wish to sample from the dataset

import seaborn as sns
import matplotlib.pyplot as plt

indices = [10, 30, 55]


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

sns.heatmap((samples-data.mean())/data.std(ddof=0), annot= True, cbar =False, squ

plt.yticks([2.5, 1.5, .5], ['Index'+str(x) for x in indices], rotation ='horizont
plt.xticks(rotation=45, ha ='center')
plt.title('Percentile ranks of \nsamples\' category spending');
```

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

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	3366	5403	12974	4400	5977	1744
1	18815	3610	11107	1148	2134	2963
2	5264	3683	5005	1057	2024	1130



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.

In [ ]:
```
** Answer:

Segment 0:

    Could be a retailer as the grocery expenditure for the segment is $12,974.00 
quartile range ($10,655) and also the mean($7,951). Detergents_Paper and Delicate
the Mean of the Dataset and Detergents_paper 75 per cent quartile range is also h
quartile and mean for Fresh in the Dataset were both higher than the expenditure 
Milk 75 per cent quartile was $7,190 for the Dataset while it was $5,403 for segm
both more than the Mean and the 75 per cent quartile of the Dataset. The expendit
seems to confirm Segment 0 is most likely a retailer.

Segment 1:

    Segment 1 is most likely a Market; the Fresh order and the Grocery order are 
    segment 0 was $18,815 while the 75 per cent Quartile range expenditure for th
    $12,000. The Grocery order expenditure is more than 30 per cent than that of 
    cent Quartile range for the Dataset is $10,655.75 while expenditure for segme
    Delicatessen all the other feature's expenditure are below the Mean expenditu


Segment 2:

    Segment 2 to appears to be a cafe given fresh and grocery along with milk are
    expenditures are all below the Mean and the 75 per cent Quartile expenditure 
    deduction this could be a cafe; they purchase a bit of everything just in a l
    the dataset.

**
```

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 [5]:
```
def feature_in_Dataset(feature):

    # TODO: Make a copy of the DataFrame, using the 'drop' function to drop the g


    feature_data = data.drop([feature], axis=1)

    #TODO: SPLit the data into training and testing sets using the given feature 

    from sklearn.cross_validation import train_test_split

    X_train, X_test, y_train, y_test = train_test_split(feature_data, data[featur

    # TODO: Create a decision tree regressor and fit it to the training set

    from sklearn.tree import DecisionTreeRegressor

    regressor = DecisionTreeRegressor(random_state=31)

    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 ("The score for feature {:16} is {:+.8f}".format(feature, score))

for feature in data.columns.values:
    feature_in_Dataset(feature)
```

```
The score for feature Fresh            is -0.66649735
The score for feature Milk             is +0.22635211
The score for feature Grocery          is +0.78376321
The score for feature Frozen           is -4.47028601
The score for feature Detergents_Paper is +0.71374182
The score for feature Delicatessen     is -2.91704723
```

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 attempt to predict Detergents_Paper. The reported prediction score is 0.713741818774.Given that higher coeeficient of Determination means better fit a score of approximately 0.714 means this feature (Detergents_Paper) might not be necessary for identifying customers' spending habits. I then decided to include all six features and the results showed that Grocery and as mentioned earlier Detergents_Paper are highly correlated and are thus not good predictors.

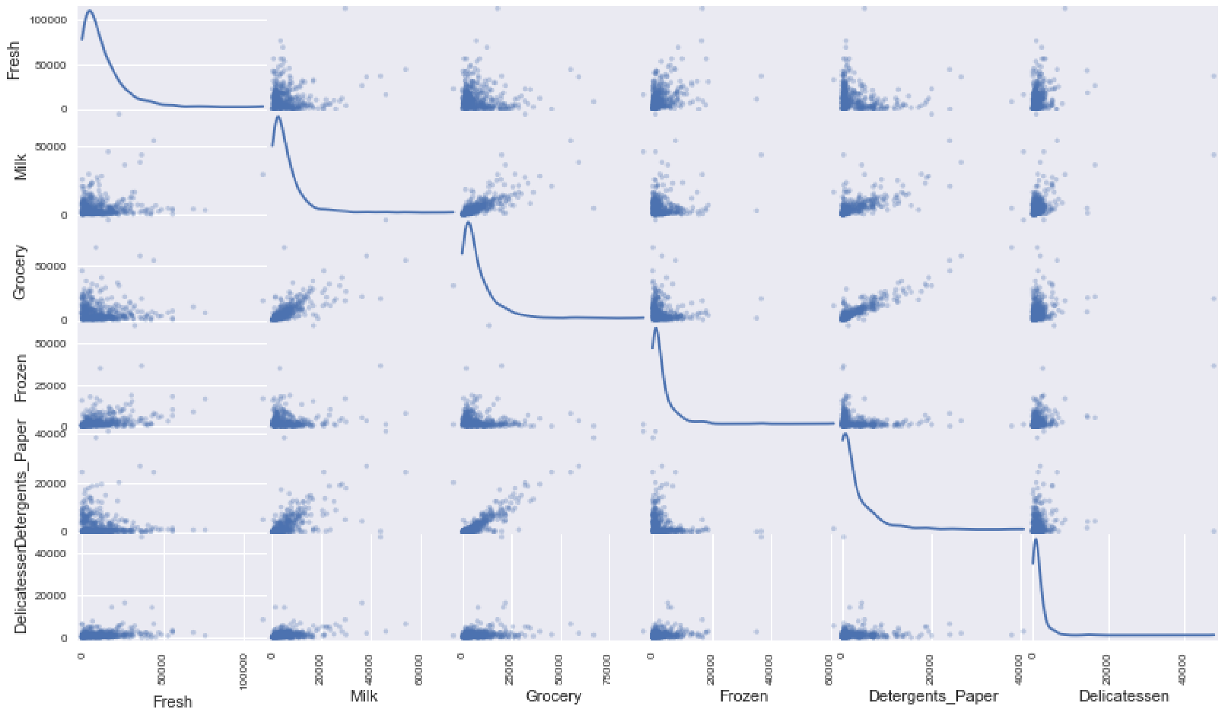
Milk displays low correlation with the other categories and as such could be a good qualifier for determining customer spendings.

**

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 [6]:
```
# Produce a scatter matrix for each pair of features in the data
pd.plotting.scatter_matrix(data, alpha = 0.3, 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:
All of the features data have long tails and are skewed right, this means the distributions have fewer observations on the right (toward higher values)[1].Given that a normal distribution has the mean as the center of the distribution and the observations are symmetrical around the mean that is on both the left and right side; none of the observed data distribution fit this description. Most of the data for each feature lies toward the right of the mode. Detergent_Paper and Grocery exhibit some correlation aslo frozen and fresh also show a bit of correlation. It confirm to some extent that detergents_paper feature might not be as relevant to predicting the customer segment.

Reference:

[1] http://stattrek.com/statistics/charts/data-patterns.aspx (http://stattrek.com/statistics/charts/data-patterns.aspx) **

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.

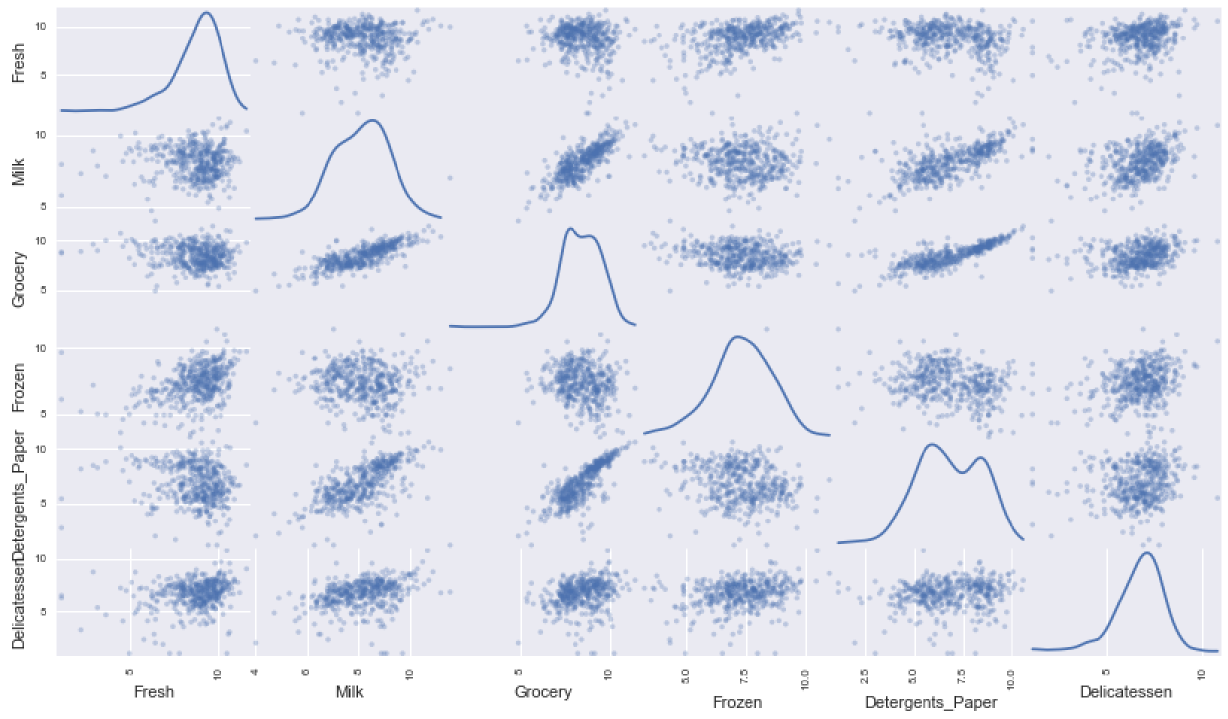
Implementation: Feature Scaling

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 [7]:
```
# 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.plotting.scatter_matrix(log_data, alpha = 0.3, figsize = (14,8), diagonal = 'k
```



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 [8]:
```
# Display the log-transformed sample data
display(log_samples)
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	8.121480	8.594710	9.470703	8.389360	8.695674	7.463937
1	9.842410	8.191463	9.315331	7.045777	7.665753	7.993958
2	8.568646	8.211483	8.518193	6.963190	7.612831	7.029973

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 [9]:
```
# For each feature find the data points with extreme high or low values

count ={}


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 times t
    step = (Q3-Q1)* 1.5

    # Display the outliers
    print "Data points considered outliers for the feature '{}':".format(feature)
    display(log_data[~((log_data[feature] >= Q1 - step) & (log_data[feature] <= Q
    for i in log_data[~((log_data[feature] >=Q1- step) & (log_data[feature] <= Q3
        count[i] = count.get(i, 0) +1


# OPTIONAL: Select the indices for data points you wish to remove

outliers =list(count.keys())

print [i for i in count.items() if i[1] >1]




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

#Original and new Data

print 'Original shape of data:\n', data.shape

print 'New shape of the data:\n', good_data.shape
```

```
Data points considered outliers for the feature 'Fresh':
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

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:

There are data points that are considered outliers, namely 65(2 features), 66(2 features), 75 (2 features), 128(2 features) and 154(3 features). Given that clustering the data will be better with a more normal distribution it will make sense to remove the data points that will skew the data. After applying the interquartile range step, using data points that are considered out of the IQR(Inter Quartile Range) maybe counter-intuitive.

Later we will be using principal compomnent analyis(PCA), the principle components will often do a good job of differentiating the outliers, but at the expense of differentiating more interesting groups[1]. Hence why removing the ouliers by this method it will make the data better optimized for applying the PCA.

The effect of droppping these Outliers may affect both results and assumptions[2].Main reason for this is: The results are actual values of expenditure by customers. So though the are able to skew the data when taken together they still represents valid customer expenditures.

Reference

[1] https://www.quora.com/What-effect-does-removing-outliers-have-on-quantitative-analysis-of-genetics-data (https://www.quora.com/What-effect-does-removing-outliers-have-on-quantitative-analysis-of-genetics-data)

[2] http://www.theanalysisfactor.com/outliers-to-drop-or-not-to-drop/ (http://www.theanalysisfactor.com/outliers-to-drop-or-not-to-drop/)

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

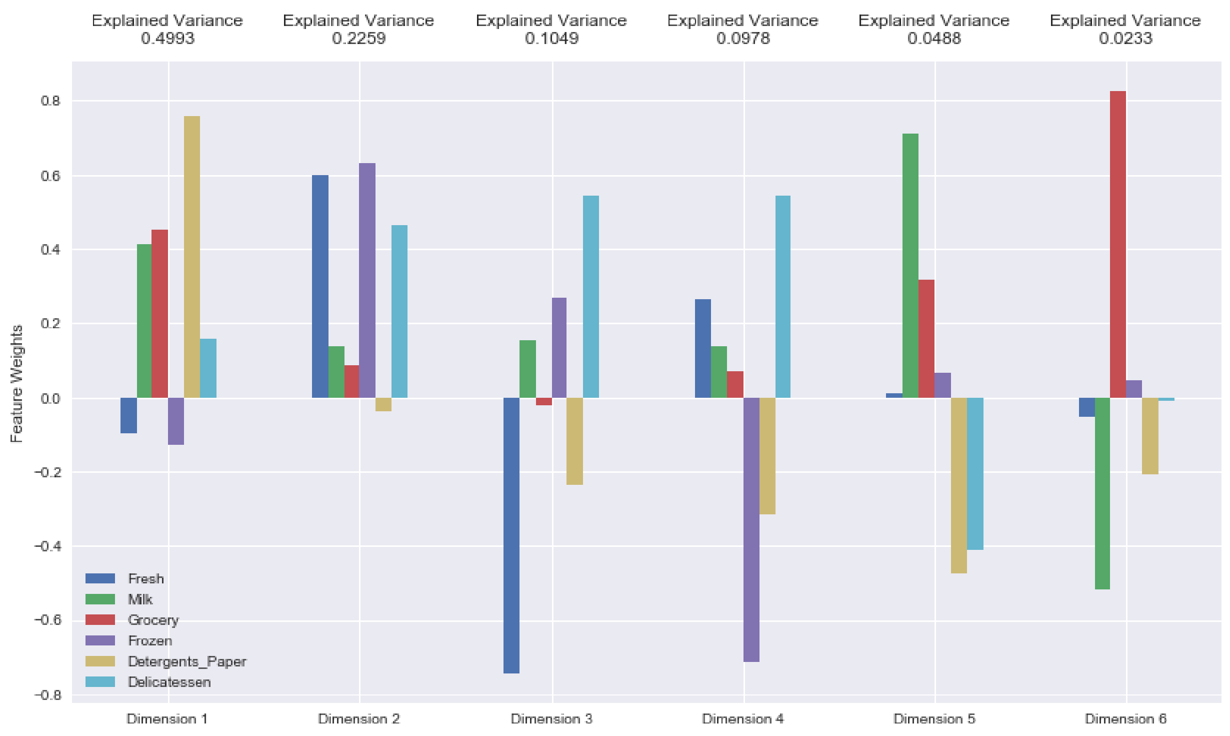
Now that the data has been scaled to a more normal distribution and has had any necessary outliers removed, we can now apply PCA to the `good_data` to discover which dimensions about the data best maximize the variance of features involved. In addition to finding these dimensions, PCA will also report 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 [10]:
```
# TODO: Apply PCA by fitting the good data with the same number of dimensions as
from sklearn.decomposition import PCA

pca = PCA(n_components=6)
pca.fit(good_data)


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

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

#Cumulative sums

print "1 &2 combined: ", pca_results['Explained Variance'].cumsum()[1]
print "1,2,3&4 combined: ",pca_results['Explained Variance'].cumsum()[3]
```

```
1 &2 combined:  0.7252
1,2,3&4 combined:  0.9279
```



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 total variance of the first and second principal components is 71.9%. The first four components have a combined principal components of 93.14%.

Dimension 1 had a total variance of 44.24%. Its total comprises of a large contribution from Milk, Grocery and Detergents_paper. In earlier observation it was seen grocery and detergent_paper had the greatest correlation. The contributions from frozen, fresh and delicatessen are very similar and account for less than the three features previously discussed.This dimension seems to represent purchases by customers of supplies used daily and on a regular basis.

Dimension 2 Customer purchases in this dimension are made up predominantly of fresh, frozen and delicatessen. These items looks similar to what customers may purchase from a cafe.

Dimension 3: This is comprised of large contribution from Delicatessen, frozen and fresh. This dimension showed customers purchasing mostly high value foods and not food that needs further preparation.

Dimension 4: Frozen has a high positive feature weight, while delicatessen and fresh round out the top contributor to this dimension. Customers have a high demand for frozen in this dimension.

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 [11]:
```
# Display sample log-data after having a PCA transformation applied
display(pd.DataFrame(np.round(pca_samples, 4), columns = pca_results.index.values
```

	Dimension 1	Dimension 2	Dimension 3	Dimension 4	Dimension 5	Dimension 6
0	2.1793	0.5069	0.8243	-1.0031	-0.4516	0.3107
1	1.2491	0.9103	-0.3467	0.9615	-0.5883	0.4436
2	0.8402	-0.4169	0.0896	0.1186	-0.4245	-0.1404

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 [12]:
```
# TODO: Apply PCA by fitting the good data with only two dimensions
pca = PCA(n_components =2)
pca.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 [13]:
```
# Display sample log-data after applying PCA transformation in two dimensions
display(pd.DataFrame(np.round(pca_samples, 4), columns = ['Dimension 1', 'Dimensi
```

	Dimension 1	Dimension 2
0	2.1793	0.5069
1	1.2491	0.9103
2	0.8402	-0.4169

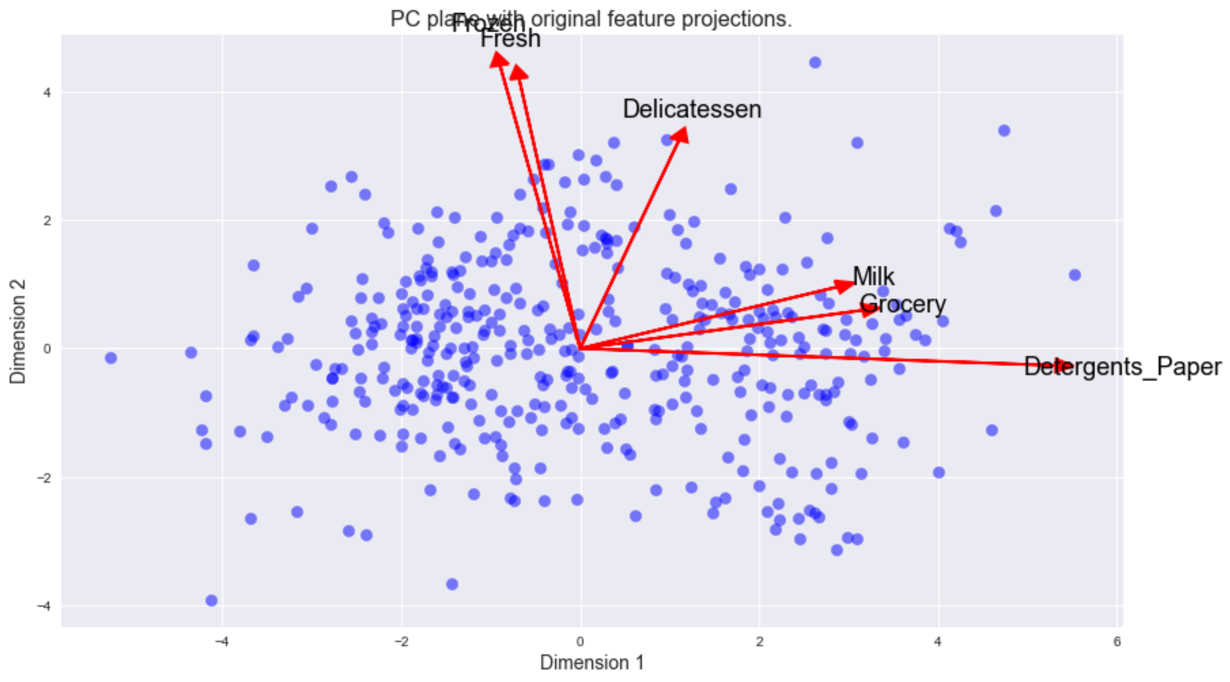
Visualizing a Biplot

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

```
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0xedf51d0>
```



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:

```
            Advantages of using K-Means clustering algorithm
```

Easy to implement.[1]

The K-means procedure is easily programmed and is computationally economical, so that it is feasible to process very large samples on a digital computer. K-means algortithm is one of first which a data analyst will use to investigate a new data set because it is algorithmically simple, relatively robust and gives “good enough” answers over a wide variety of data sets.[2]

References:

[1].(Harrington, Peter. Machine Learning in Action,p208. New York: Manning Publications Co. 2012)

[2] Wesan, Barbakh And Colin Fyfe. “Local vs global interactions in clustering algorithms: Advances over K-means.” International Journal of knowledge-based and Intelilligent Engineering Systems 12 (2008).83 – 99

```
            Advantages to using a Gaussian Mixture Model clustering algo
rithm
```

It is the fastest algorithm for learning mixture models[1]

The popularity arises not only from its highly desirable computational properties, but also from its ability to approximate many naturally occurring real-world data, thanks to the law of large numbers. [2]

References:

[1] http://scikit-learn.org/stable/modules/mixture.html (http://scikit-learn.org/stable/modules/mixture.html)

[2] (Yu, D; Deng, L. Automatic Speech Recognition, A deep learning approach, Chapter 2. Springer, 2015.)

Even though K-means clustering algorithm is faster it tends to work better with more uniform clustering. Even though we applied the PCA (Principal Component Analysis) , I still feel given the distribution noted by the features in the dataset using Gaussian Mixture model may be more appropriae.

**

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 [15]:
```
# TODO: Apply your clustering algorithm of choice to the reduced data

from sklearn.mixture import GaussianMixture
from sklearn.metrics import silhouette_samples, silhouette_score


clusterer = GaussianMixture(n_components=3, random_state=42)
clusterer.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 chos
score = silhouette_score(reduced_data, preds)

print "GaussianMixture:" , score

#Silhouette score for 9 different number of cluusters for GaussianMixture

# 2  ==  0.40988504347
# 3  ==  0.41057253381
# 4  ==  0.302923653936
# 5  ==  0.234385436856
# 6  ==  0.297454420693
# 7  ==  0.35301486966
# 8  ==  0.31424715791
# 9  ==  0.303039972618
# 11 ==  0.320853889284
```

```
GaussianMixture: 0.361193625039
```

Question 7

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

**Answer:

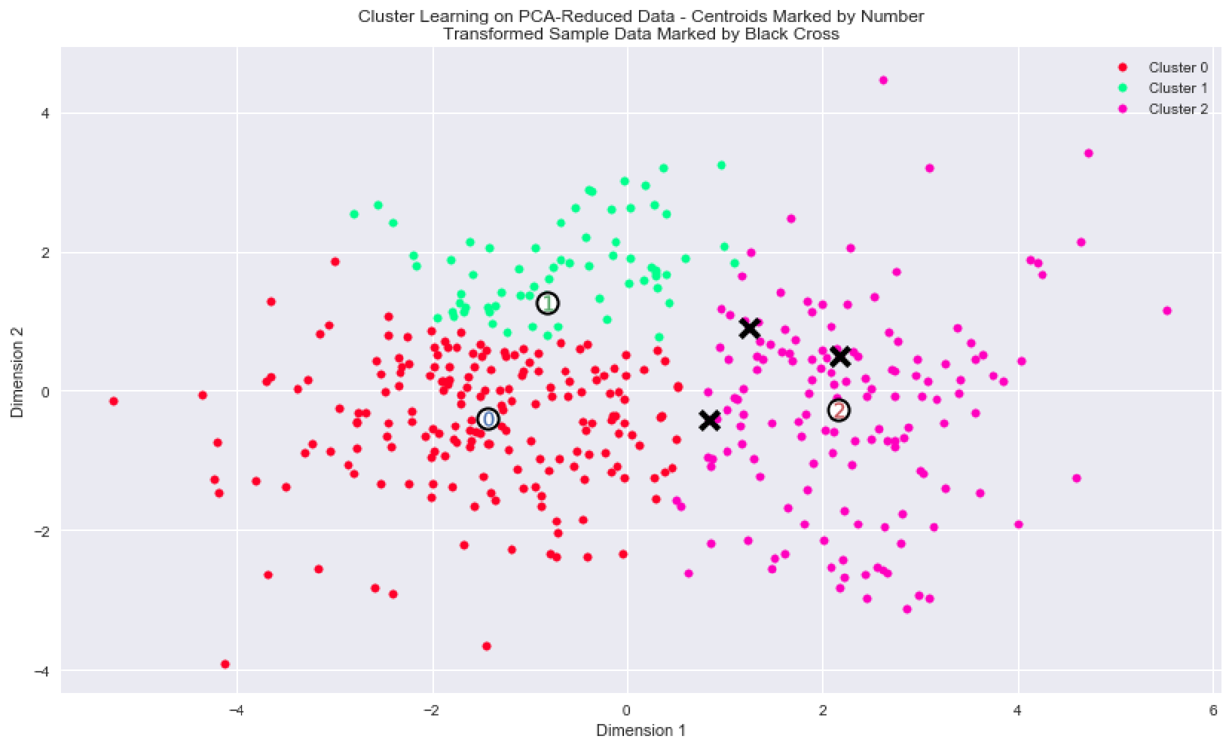
The number of clusters with the best silhouette score is 3.

**

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 [16]:
```
# 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 [17]:
```
# TODO: Inverse transform the centers
log_centers = pca.inverse_transform(centers)

# TODO: Exponentiate the centers
true_centers = np.exp(log_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)

#A closer look at 'good dataset'

print "\nStatistics with Outliers removed."
display(np.exp(good_data).describe())
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
Segment 0	6904.0	1770.0	2295.0	1584.0	303.0	599.0
Segment 1	17686.0	2854.0	3478.0	4183.0	451.0	1425.0
Segment 2	5200.0	7904.0	11779.0	1073.0	4656.0	1115.0

```
Statistics with Outliers removed.
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
count	398.000000	398.000000	398.000000	398.000000	398.000000	398.00000
mean	12430.630653	5486.314070	7504.907035	3028.809045	2725.376884	1454.71608
std	12552.698266	6410.878177	9263.803670	3712.563636	4644.023066	1746.45365
min	255.000000	201.000000	223.000000	91.000000	5.000000	46.00000
25%	4043.500000	1597.250000	2125.000000	830.000000	263.250000	448.25000
50%	9108.000000	3611.500000	4573.000000	1729.500000	788.000000	997.50000
75%	16969.000000	6802.500000	9762.250000	3745.000000	3660.500000	1830.00000
max	112151.000000	54259.000000	92780.000000	35009.000000	40827.000000	16523.00000

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 after doing the data cleaning it looks more like a market as th
e largest customer expenditure is Fresh. The total expenditure of $6,904
 is however below the mean and the 75% Quartile of the good dataset ($16,
969 and $12,430 respectively). All the other purchases are below the 50%
 Quartile of the 'Good dataset' this suggest a customer with lower spendi
ng power than the average customer in the Dataset.

Segment 1 now loks like a market also allbeit a  large one as its Fresh o
rder is the largest of the six features; coming in at $17,686 larger than
 the 75% Quartile and the Mean of the 'good Dataset'. Its next largest ex
penditure is on the grocery $3,478 this is lower than the 75% Quartile an
d the Mean of the Dataset. The other purchases are much lower than that o
f expenditure on Fresh and are also lower than the 75% Quartile of the 'g
ood Dataset'.

Segment 2 looks to be a Supermarket . The Grocery expenditure is $11,779.
00, another interesting observation is the Frozen's expenditure being bei
ng $1073 much lower than the 75% Quartile of the 'good Dataset'. The cust
omer seem to be buying non-perishable in bulk.
```

Overall my postulations in question 1 were quite different than the results of these segments. The adjusting of the data obviously had an impact on the new sample results. Given the removal of the Outliers in the new dataset the results may better reflect the more likely customer expenditure for any random segment chosen from the Data.

**

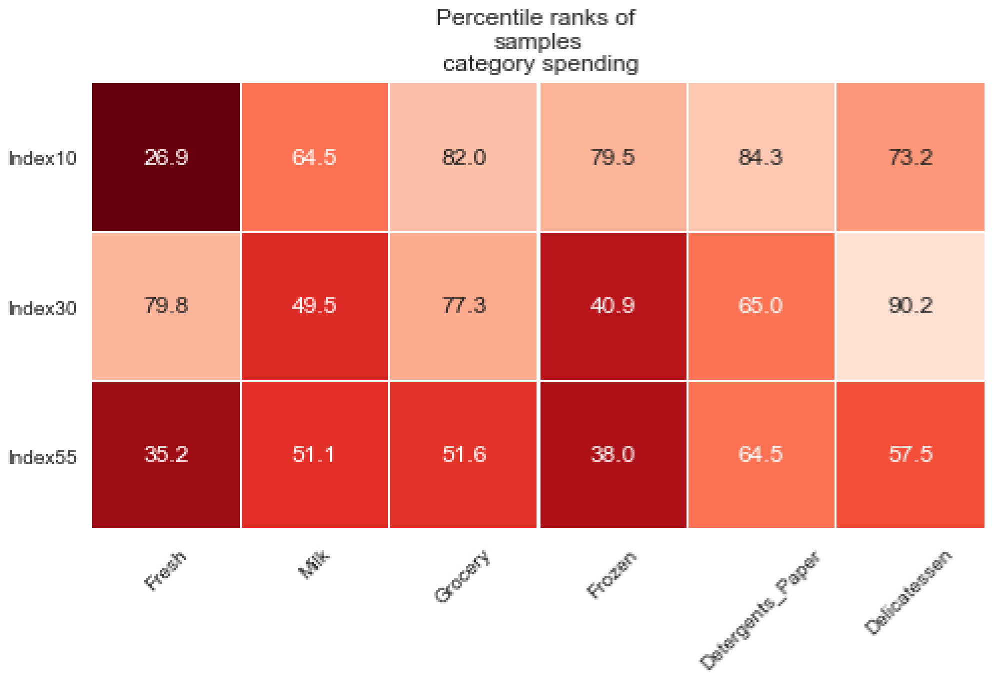
Question 9

*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 [18]:
```
# Display the predictions


for i, pred in enumerate(sample_preds):
    print "Sample point", i, "predicted to be in Cluster", pred
#percentile ranks

import matplotlib.pyplot as plt
import seaborn as sns

pcts =100. *data.rank(axis=0, pct=True).iloc[indices].round(decimals=3)
print "\n", pcts

#Visualizations with heatmap

sns.heatmap(pcts, annot=True, linewidth =0.1, vmax =99, fmt ='.1f', cmap='Reds_r'

plt.yticks([2.5, 1.5, .5], ['Index'+str(x) for x in indices], rotation ='horizont
plt.xticks(rotation=45, ha ='center')
plt.title('Percentile ranks of \nsamples\n category spending');
```

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

    Fresh  Milk  Grocery  Frozen  Detergents_Paper  Delicatessen
10   26.9  64.5     82.0    79.5              84.3          73.2
30   79.8  49.5     77.3    40.9              65.0          90.2
55   35.2  51.1     51.6    38.0              64.5          57.5
```



**Answer:

```
All three sample falls to cluster 2.  In my assessment of the samples probable
classification given the expenditure totals on items that could be
representative of supermarkets, markets and cafe I never envisioned all of them
belonging to the same category. I would say given the work done by the
clustering algorithm after removing the outliers and applying PCA the results
could be taken as correct classificaion. Segment 1 predictions does not look
consistent, its expenditure on grocery is $2,295 much lower than the 'good data'
75% Quartile value of $ 9,762. Its expenditure in Fresh would imply it retails a
larger portion of perishable, however its expenditure of $6,904 is still much
lower than the 75% Quartile of Fresh in the 'good data'.

**
```

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:

Despite the samples having confirmed they were all from supermarkets, I am leaning to the idea that it was more a general description of retailers.

For Segment 0 we could change the delivery for a short period from 5 days to 3 then conduct a customer poll to ascertain how they have been affected. A simple 2 question questionnaire at checkout could accomplish this. The retailer could also look at the sales receipts to see where they are loosing and gaining and this information could be passed on to the distrubutor to help with their decision.

For segment 1 the A/B testing could be implemented. The purchase cost seem somewhat uniform in the major product categories. It could make sense to see the inner working of the A/B testing to be utilized in this instance:

Given we have done the Clustering of the customer segements; let us see how the clusters were affected by the A/B test. If people in group A let us say customers who spent more on grocery convert 10% given the change in delivery days from 5 to 3 days while customers in group B increase by similar 10% we could say there is significance statistically speaking. However, on the other hand if there is a proportionate decrease in B let's say decrease by same 10% we could say the change is not statistically significant. We could then run A/B testing on each feature in each segment, and from there we could deduce if there are statistical differences that are significant enough to render a change in number of delivery days economically feasible.

For segment 2, Fresh product cost is nearly three times the cost of any other product category. Intuition might say changing the number of days from 5 to 3 may impact the quality of the offerings at the reailers. However A/B testing maybe implemented to assess the impact the change is having on the customers.

**

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:

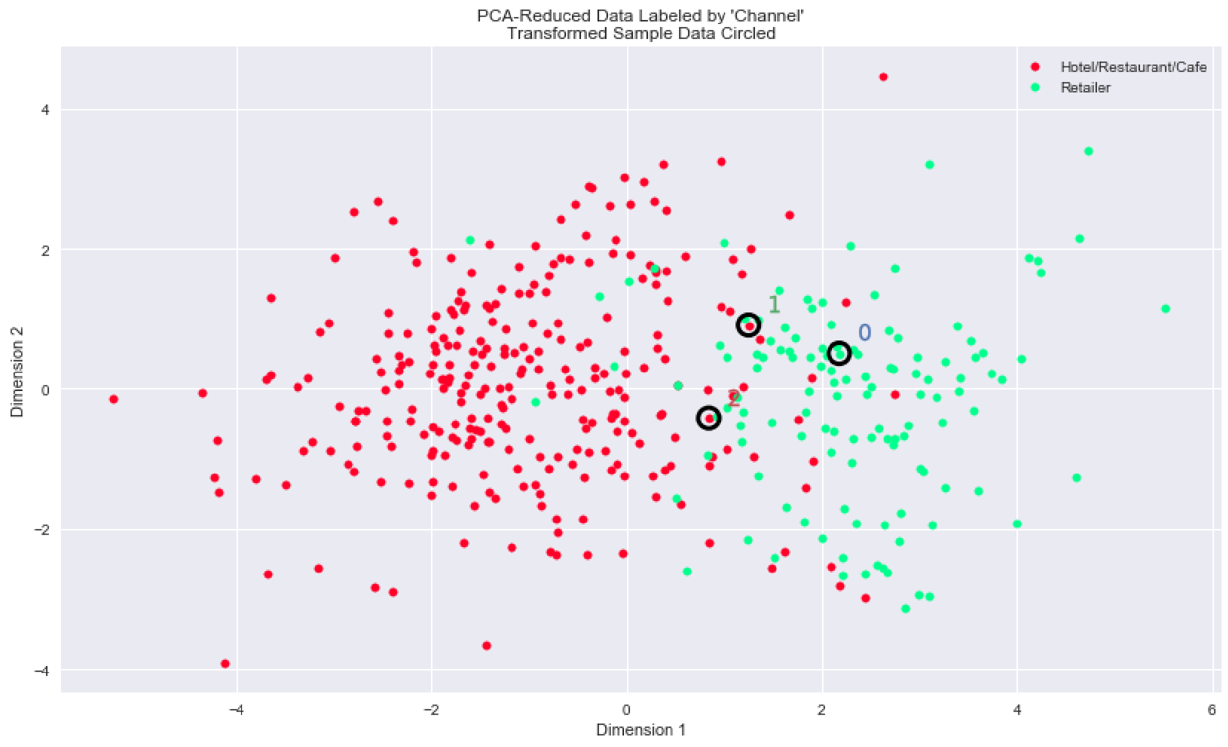
After doing the clustering with an unspervised algorithm such as GaussianMixture in this case we can use some of this data as target variable. We could use the trained GMM to predict the customer's segment as underlying for the training labels on the supervised learner. Feature engineering is the name given to using this method; we will use the output of an unsupervised learning analysis as input to a new supervised learning analyis. The new supervised learner upon being trained on the existing customer dataset using new labels can then predict new customers based on their estimated product spending.

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 [19]:
```
# 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:

Using GaussianMixture the optimal number of clusters were 3, looking at the channel data my classification would be off. The clustering from the channel data classified the customer groups into only two categories. Looking at the clustering I would say some customer segments would be classified purely as 'Retailers' or 'Hotels/Restaurants/Cafes'. To some degree I would conised the classification as consistent, however i did not bunch hotels with cafes or restaurants.

**

> **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.