Machine Learning Engineer Nanodegree

Unsupervised Learning

Project 3: 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
import renders as rs
from IPython.display import display # Allows the use of display() for DataFrames

# Show matplotlib plots inline (nicely formatted in the notebook)
%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.".fo
rmat(*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 [2]:

```
# Display a description of the dataset
display(data.describe())
```

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

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
indices = [85,338,36]
# Create a DataFrame of the chosen samples
samples = pd.DataFrame(data.loc[indices], columns = data.keys()).reset_index(dro
p = True)
print "Chosen samples of wholesale customers dataset:"
display(samples)
```

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

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	16117	46197	92780	1026	40827	2944
1	3	333	7021	15601	15	550
2	29955	4362	5428	1729	862	4626

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: first customer, I've picked the most total spending from the table and my intuition is that it probably be the restaurants because the quantity of cleaning items, grocery, and fresh are quite high while frozen is very low when compare to the rest. so it's supposed to be restaurant who cook the fresh food to customer.

Second customers, I've decided to pick this one because it's interesting to see that Fresh is extremly low when compare to Frozen and others so my intuition is this grocery store are focusing on Food and mostly frozen food.

Third customers, my intuition think it's a market because the high quantity of fresh food and not much on the detergents and paper which it's mean it is a place where you sell fresh food and not interesting in cleaning that much so I think it's a 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 [40]:

```
# TODO: Make a copy of the DataFrame, using the 'drop' function to drop the give
n feature
new_data = data
temp_data = data
# Get the forzen column in target
target_col = temp_data.columns[0]
print "target to drop " , target_col

y_all = temp_data[target_col]
new_data = data.drop(['Fresh'], axis = 1)
feature_cols = list(new_data.columns[:6])
print "the rest feature " , feature_cols

X_all = new_data[feature_cols]

# TODO: Split the data into training and testing sets using the given feature as
 the target
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_all, y_all , test_size=0.2
5, random_state=42)
# TODO: Create a decision tree regressor and fit it to the training set
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor(random_state=38)
regressor.fit(X_train,y_train)
score = regressor.score(X_test,y_test)

##### Score 2 ######
target_col = temp_data.columns[1]
y_all = temp_data[target_col]
print "target to drop " , target_col
new_data = data.drop(['Milk'], axis = 1)
feature_cols = list(new_data.columns[:6])
print "the rest feature " , feature_cols
X_all = new_data[feature_cols]
X_train, X_test, y_train, y_test = train_test_split(X_all, y_all , test_size=0.2
5, random_state=42)
regressor.fit(X_train,y_train)
score2 = regressor.score(X_test,y_test)


### Score 3 ####
target_col = temp_data.columns[2]
y_all = temp_data[target_col]
print "target to drop " , target_col
new_data = data.drop(['Grocery'], axis = 1)
feature_cols = list(new_data.columns[:6])
print "the rest feature " , feature_cols

X_all = new_data[feature_cols]
X_train, X_test, y_train, y_test = train_test_split(X_all, y_all , test_size=0.2
5, random_state=42)
regressor.fit(X_train,y_train)
score3 = regressor.score(X_test,y_test)

### Score 4 ####
target_col = temp_data.columns[3]
y_all = temp_data[target_col]
print "target to drop " , target_col
new_data = data.drop(['Frozen'], axis = 1)
feature_cols = list(new_data.columns[:6])
print "the rest feature " , feature_cols
```

```
X_all = new_data[feature_cols]
X_train, X_test, y_train, y_test = train_test_split(X_all, y_all , test_size=0.2
5, random_state=42)
regressor.fit(X_train,y_train)
score4 = regressor.score(X_test,y_test)

### Score 5 ####
target_col = temp_data.columns[4]
y_all = temp_data[target_col]
print "target to drop " , target_col

new_data = data.drop(['Detergents_Paper'], axis = 1)
feature_cols = list(new_data.columns[:6])
print "the rest feature " , feature_cols

X_all = new_data[feature_cols]
X_train, X_test, y_train, y_test = train_test_split(X_all, y_all , test_size=0.2
5, random_state=42)
regressor.fit(X_train,y_train)
score5 = regressor.score(X_test,y_test)

### Score 6 ####
target_col = temp_data.columns[5]
y_all = temp_data[target_col]
print "target to drop " , target_col

new_data = data.drop(['Delicatessen'], axis = 1)
feature_cols = list(new_data.columns[:6])
print "the rest feature " , feature_cols

X_all = new_data[feature_cols]
X_train, X_test, y_train, y_test = train_test_split(X_all, y_all , test_size=0.2
5, random_state=42)
regressor.fit(X_train,y_train)
score6 = regressor.score(X_test,y_test)

print ""
print "Score for each feature"
print "Fresh " ,score
print "Milk " ,score2
print "Grocery " ,score3
print "Frozen " ,score4
print "Detergents_Paper " ,score5
print "Delicatessen " ,score6
```

```
target to drop  Fresh
the rest feature  ['Milk', 'Grocery', 'Frozen', 'Detergents_Paper',
 'Delicatessen']
target to drop  Milk
the rest feature  ['Fresh', 'Grocery', 'Frozen', 'Detergents_Paper',
 'Delicatessen']
target to drop  Grocery
the rest feature  ['Fresh', 'Milk', 'Frozen', 'Detergents_Paper', 'D
elicatessen']
target to drop  Frozen
the rest feature  ['Fresh', 'Milk', 'Grocery', 'Detergents_Paper',
 'Delicatessen']
target to drop  Detergents_Paper
the rest feature  ['Fresh', 'Milk', 'Grocery', 'Frozen', 'Delicatess
en']
target to drop  Delicatessen
the rest feature  ['Fresh', 'Milk', 'Grocery', 'Frozen', 'Detergents
_Paper']

Score for each feature
Fresh  -0.383151155093
Milk  0.106643975351
Grocery  0.679453054149
Frozen  -0.255399635128
Detergents_Paper  0.490544025213
Delicatessen  -11.4463191367
```

Question 2

Which feature did you attempt to predict? What was the reported prediction score? Is this feature is 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.

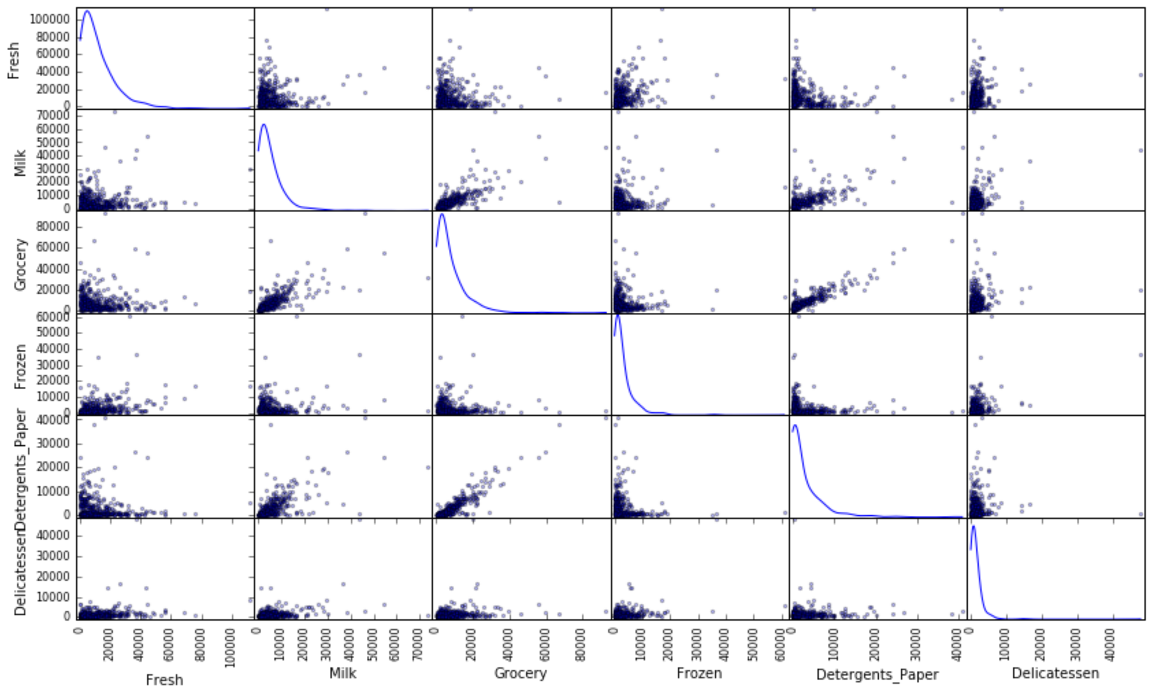
Answer: attemp to predict "Frozen" which got -0.255 prediction score so it's mean that Frozen is failed to fit the data. So I am curious that why I got negative value so I've decided to check all value from other field if it's drop what's going to happen, and here's the result. From above the data, I could summarize that Grocery is the most perfect fit of all featuers

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 [41]:

```
# Produce a scatter matrix for each pair of features in the data
pd.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 the features is very skew to the far left so from my point of view that it's so hard to understand the graph from this skew data because data is not normal distributed where most of the data lie on the left and it's not look like the bell curve at all

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:

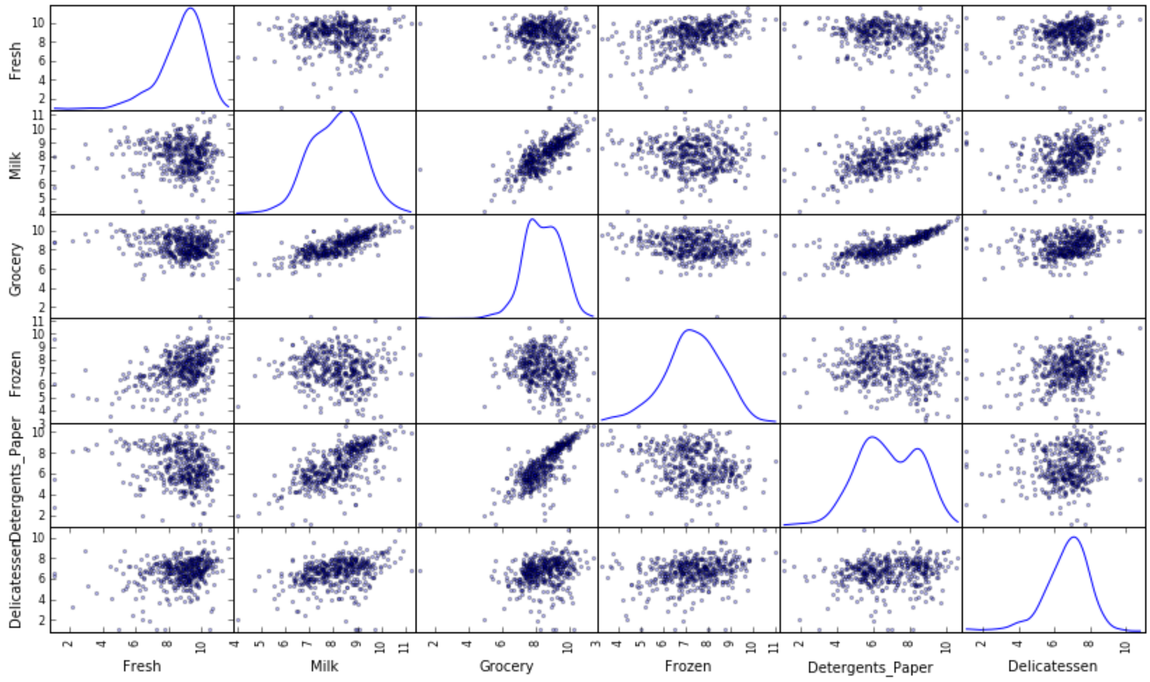
- Assign a copy of the data to `log_data` after applying a logarithm scaling. Use the `np.log` function for this.
- Assign a copy of the sample data to `log_samples` after applying a logrithm scaling. Again, use `np.log`.

In [42]:

```
# 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 [44]:

```
# Display the log-transformed sample data
display(log_samples)
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	9.687630	10.740670	11.437986	6.933423	10.617099	7.987524
1	1.098612	5.808142	8.856661	9.655090	2.708050	6.309918
2	10.307452	8.380686	8.599326	7.455298	6.759255	8.439448

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 [81]:

```
# 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 times
 the interquartile range)
    step = 1.5*(Q3-Q1)

    # Display the outliers
    print "Data points considered outliers for the feature
'{}':".format(feature)
    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  = [0,5]

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

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? 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: I have consider to outliers "Fresh" and "Delicatessen" because a lot of data from this features are lower than Q1-step and greater than Q3+step which is consider this data might be too critical(Too low, or Too high when compare to the median)

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:

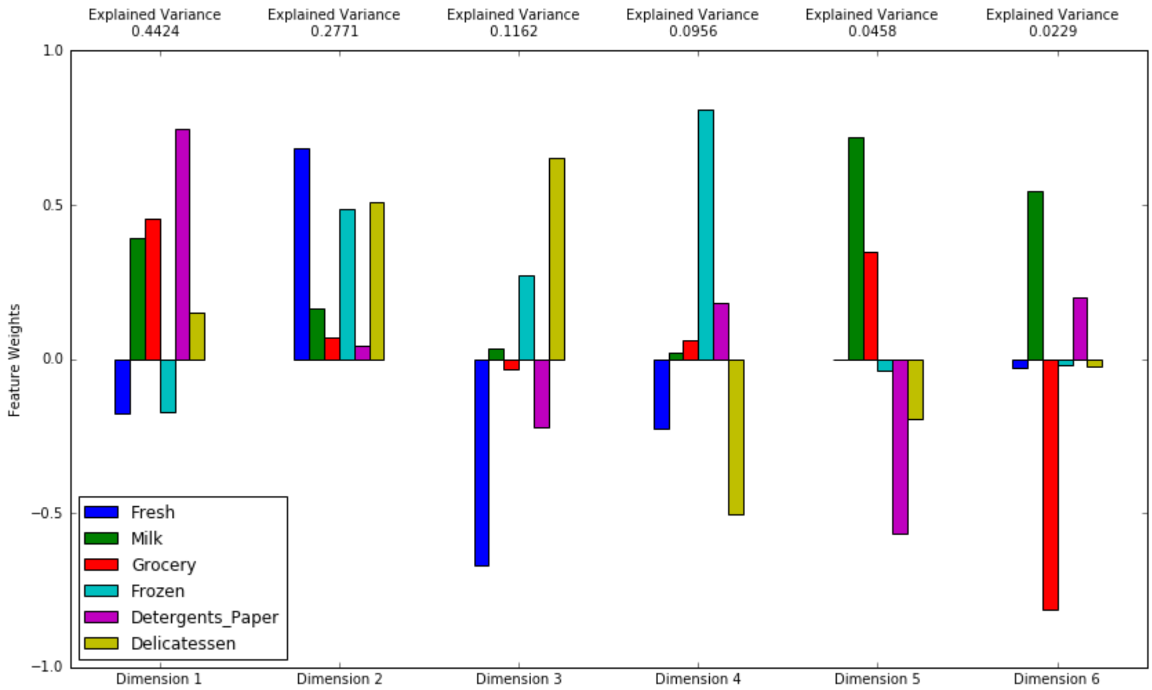
- Import `sklearn.decomposition.PCA` and assign the results of fitting PCA in six dimensions with `good_data` to `pca`.
- Apply a PCA transformation of the sample log-data `log_samples` using `pca.transform`, and assign the results to `pca_samples`.

In [76]:

```
from sklearn.decomposition import PCA
# TODO: Apply PCA to the good data with the same number of dimensions as feature
sn = min(data.shape)
featuresn = min(data.shape)
pca = PCA(n_components=featuresn)
pca.fit(good_data)

# TODO: Apply a PCA transformation to the sample log-data
pca_samples = pca.transform(log_samples)

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



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 indivdual feature weights.

Answer: In first and second principal component variance are positive-weighted while the third has very high negative on Fresh data and the fourth has very high negative on Delicatessen data also.

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 [77]:

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

	Dimension 1	Dimension 2	Dimension 3	Dimension 4	Dimension 5	Dimension 6
0	5.3495	1.9487	-0.7479	-0.2439	0.5348	-0.2842
1	-2.8707	-4.7774	6.3275	3.0379	0.7533	-2.2454
2	0.1216	2.1124	0.1518	-1.1204	-0.0915	-0.0764

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 reuslts to `reduced_data`.
- Apply a PCA transformation of the sample log-data `log_samples` using `pca.transform`, and assign the results to `pca_samples`.

In [79]:

```
# TODO: Fit PCA to the good data using only two dimensions
pca = PCA(n_components=2)
pca.fit(good_data)

# TODO: Apply a PCA transformation the good data
reduced_data = pca.transform(good_data)

# TODO: Apply a PCA transformation to the sample log-data
pca_samples = pca.transform(log_data)

# 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 [80]:

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

	Dimension 1	Dimension 2
0	1.7518	0.0715
1	1.8063	0.8721
2	1.8993	1.6790
3	-1.1195	1.4588
4	0.8014	2.4624
5	1.0871	0.3941
6	1.1364	-0.2003
7	1.5883	0.9704
8	0.8762	-0.5956
9	2.8946	0.7441
10	2.1259	0.7522
11	-0.9855	0.0630
12	2.2319	1.3291
13	1.9106	1.3366
14	2.3336	0.9851
15	-0.4132	-0.8141
16	2.8100	-1.9509
17	-0.2492	0.6047
18	1.4067	1.8673
19	1.0216	-0.4135
20	0.8394	1.1485
21	-1.7159	-0.1470
22	0.1615	2.8315
23	2.8249	3.7186
24	2.0862	2.5556
25	1.2283	-1.5174
26	-1.8755	0.4287
27	-2.3933	-0.5384
28	3.5925	1.0758
29	-0.5713	1.1438
...	...	...
410	0.4965	0.4189
411	-0.7119	0.0770
412	2.1983	-5.3327

413	0.1191	1.7309
414	-0.5239	0.0285
415	1.6150	0.8682
416	2.5631	-0.0492
417	1.9366	-0.4680
418	3.3192	-2.3801
419	0.3308	0.1612
420	3.0202	-1.6911
421	1.4675	1.1004
422	-0.4326	0.7858
423	0.4253	0.5146
424	1.0907	-0.0531
425	-2.3305	1.8134
426	2.0958	1.6511
427	-0.4332	2.6790
428	0.1508	-0.6583
429	-1.4227	-1.0911
430	0.1311	-0.3678
431	0.2427	1.7346
432	-0.1251	-0.4951
433	-0.9212	-0.5721
434	1.0154	0.4458
435	-0.5995	2.8932
436	-3.1812	2.0063
437	3.7504	0.9302
438	-1.6543	0.4754
439	-0.7342	-3.6350

440 rows × 2 columns

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: The advantages of K-Means is it's the very fast algorithm but it will fall into local minimum which depends on the random centroid. Mostly it will use for prototype of learning and you have to run it several times to see the differenct result while Gaussian Mixture Model clustering algorithm might not be as fast as K-Means but it won't bias towards zero or bias the cluster size toward specific data. So this reason, I'm choosing Gaussian Mixture Model over K-means because of it's not bias the cluster like K-means

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 [135]:

```
# TODO: Apply your clustering algorithm of choice to the reduced data
from sklearn import mixture
clusterer = mixture.GMM(n_components=2)
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)
from sklearn.metrics import silhouette_score
# TODO: Calculate the mean silhouette coefficient for the number of clusters cho
sen
score = silhouette_score(reduced_data,preds)
print "2 Clusters : " , score

clusterer = mixture.GMM(n_components=3)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
print "3 Clusters : " , score

clusterer = mixture.GMM(n_components=4)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
print "4 Clusters : " , score

clusterer = mixture.GMM(n_components=5)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
print "5 Clusters : " , score

clusterer = mixture.GMM(n_components=6)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
print "6 Clusters : " , score

clusterer = mixture.GMM(n_components=7)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
print "7 Clusters : " , score

clusterer = mixture.GMM(n_components=8)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
print "8 Clusters : " , score

clusterer = mixture.GMM(n_components=3)
```

```
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
sample_preds = clusterer.predict(pca_samples)
score = silhouette_score(reduced_data,preds)
```

```
2 Clusters :  0.323547497098
3 Clusters :  0.373425672571
4 Clusters :  0.361485520252
5 Clusters :  0.257009024138
6 Clusters :  0.264295620681
7 Clusters :  0.312474736887
8 Clusters :  0.31496603254
```

Question 7

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

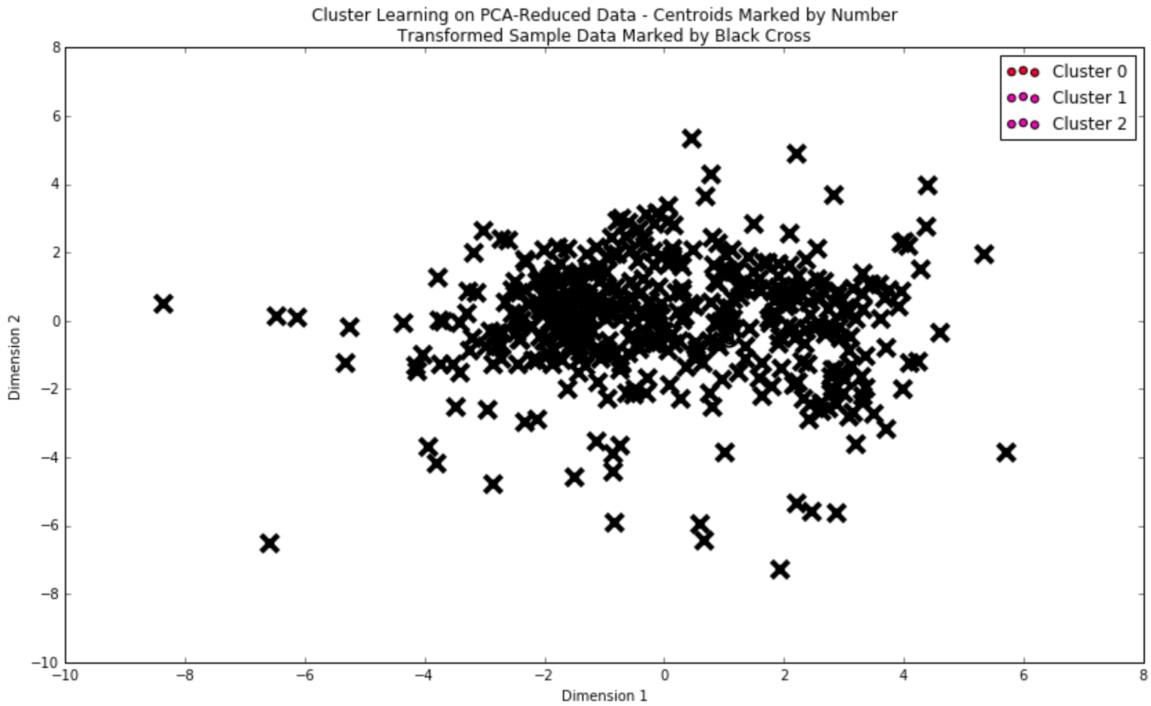
Answer: I've tried to change to different cluster but 3 cluster have the best silhouette score

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 [121]:

```
# Display the results of the clustering from implementation
rs.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 [123]:

```
# 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)
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
Segment 0	3791.0	4785.0	7353.0	1003.0	1944.0	737.0
Segment 1	8788.0	2587.0	3309.0	1988.0	497.0	817.0

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:

From the score, there's recommend 3 clusters in this data while there are only 2 segments from my point of view Segment 0 can represented cluster 1 and 2 where Grocery is the highest while Segment 1 represented cluster 3 where Fresh is the highest

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 [125]:

```
# 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 2
Sample point 1 predicted to be in Cluster 2
Sample point 2 predicted to be in Cluster 2
Sample point 3 predicted to be in Cluster 1
Sample point 4 predicted to be in Cluster 2
Sample point 5 predicted to be in Cluster 2
Sample point 6 predicted to be in Cluster 2
Sample point 7 predicted to be in Cluster 2
Sample point 8 predicted to be in Cluster 2
Sample point 9 predicted to be in Cluster 2
Sample point 10 predicted to be in Cluster 2
Sample point 11 predicted to be in Cluster 1
Sample point 12 predicted to be in Cluster 2
Sample point 13 predicted to be in Cluster 2
Sample point 14 predicted to be in Cluster 2
Sample point 15 predicted to be in Cluster 1
Sample point 16 predicted to be in Cluster 2
Sample point 17 predicted to be in Cluster 1
Sample point 18 predicted to be in Cluster 2
Sample point 19 predicted to be in Cluster 2
Sample point 20 predicted to be in Cluster 2
Sample point 21 predicted to be in Cluster 1
Sample point 22 predicted to be in Cluster 2
Sample point 23 predicted to be in Cluster 2
Sample point 24 predicted to be in Cluster 2
Sample point 25 predicted to be in Cluster 2
Sample point 26 predicted to be in Cluster 1
Sample point 27 predicted to be in Cluster 1
Sample point 28 predicted to be in Cluster 2
Sample point 29 predicted to be in Cluster 1
Sample point 30 predicted to be in Cluster 2
Sample point 31 predicted to be in Cluster 2
Sample point 32 predicted to be in Cluster 1
Sample point 33 predicted to be in Cluster 1
Sample point 34 predicted to be in Cluster 0
Sample point 35 predicted to be in Cluster 2
Sample point 36 predicted to be in Cluster 2
Sample point 37 predicted to be in Cluster 2
Sample point 38 predicted to be in Cluster 2
Sample point 39 predicted to be in Cluster 1
Sample point 40 predicted to be in Cluster 2
Sample point 41 predicted to be in Cluster 2
Sample point 42 predicted to be in Cluster 2
Sample point 43 predicted to be in Cluster 2
Sample point 44 predicted to be in Cluster 2
Sample point 45 predicted to be in Cluster 2
Sample point 46 predicted to be in Cluster 2
Sample point 47 predicted to be in Cluster 2
Sample point 48 predicted to be in Cluster 2
Sample point 49 predicted to be in Cluster 2
Sample point 50 predicted to be in Cluster 1
Sample point 51 predicted to be in Cluster 2
Sample point 52 predicted to be in Cluster 2
Sample point 53 predicted to be in Cluster 2
Sample point 54 predicted to be in Cluster 1
Sample point 55 predicted to be in Cluster 2
Sample point 56 predicted to be in Cluster 2
Sample point 57 predicted to be in Cluster 2
Sample point 58 predicted to be in Cluster 1
Sample point 59 predicted to be in Cluster 2
Sample point 60 predicted to be in Cluster 2
```

```
Sample point 61 predicted to be in Cluster 2
Sample point 62 predicted to be in Cluster 2
Sample point 63 predicted to be in Cluster 2
Sample point 64 predicted to be in Cluster 1
Sample point 65 predicted to be in Cluster 0
Sample point 66 predicted to be in Cluster 0
Sample point 67 predicted to be in Cluster 2
Sample point 68 predicted to be in Cluster 2
Sample point 69 predicted to be in Cluster 1
Sample point 70 predicted to be in Cluster 1
Sample point 71 predicted to be in Cluster 2
Sample point 72 predicted to be in Cluster 1
Sample point 73 predicted to be in Cluster 2
Sample point 74 predicted to be in Cluster 2
Sample point 75 predicted to be in Cluster 0
Sample point 76 predicted to be in Cluster 1
Sample point 77 predicted to be in Cluster 2
Sample point 78 predicted to be in Cluster 1
Sample point 79 predicted to be in Cluster 2
Sample point 80 predicted to be in Cluster 1
Sample point 81 predicted to be in Cluster 0
Sample point 82 predicted to be in Cluster 2
Sample point 83 predicted to be in Cluster 1
Sample point 84 predicted to be in Cluster 2
Sample point 85 predicted to be in Cluster 2
Sample point 86 predicted to be in Cluster 2
Sample point 87 predicted to be in Cluster 2
Sample point 88 predicted to be in Cluster 1
Sample point 89 predicted to be in Cluster 1
Sample point 90 predicted to be in Cluster 1
Sample point 91 predicted to be in Cluster 1
Sample point 92 predicted to be in Cluster 2
Sample point 93 predicted to be in Cluster 1
Sample point 94 predicted to be in Cluster 2
Sample point 95 predicted to be in Cluster 0
Sample point 96 predicted to be in Cluster 0
Sample point 97 predicted to be in Cluster 0
Sample point 98 predicted to be in Cluster 0
Sample point 99 predicted to be in Cluster 1
Sample point 100 predicted to be in Cluster 2
Sample point 101 predicted to be in Cluster 2
Sample point 102 predicted to be in Cluster 2
Sample point 103 predicted to be in Cluster 2
Sample point 104 predicted to be in Cluster 1
Sample point 105 predicted to be in Cluster 1
Sample point 106 predicted to be in Cluster 2
Sample point 107 predicted to be in Cluster 2
Sample point 108 predicted to be in Cluster 2
Sample point 109 predicted to be in Cluster 0
Sample point 110 predicted to be in Cluster 1
Sample point 111 predicted to be in Cluster 2
Sample point 112 predicted to be in Cluster 1
Sample point 113 predicted to be in Cluster 1
Sample point 114 predicted to be in Cluster 1
Sample point 115 predicted to be in Cluster 1
Sample point 116 predicted to be in Cluster 1
Sample point 117 predicted to be in Cluster 1
Sample point 118 predicted to be in Cluster 1
Sample point 119 predicted to be in Cluster 1
Sample point 120 predicted to be in Cluster 1
Sample point 121 predicted to be in Cluster 1
```

```
Sample point 122 predicted to be in Cluster 0
Sample point 123 predicted to be in Cluster 2
Sample point 124 predicted to be in Cluster 1
Sample point 125 predicted to be in Cluster 2
Sample point 126 predicted to be in Cluster 1
Sample point 127 predicted to be in Cluster 2
Sample point 128 predicted to be in Cluster 0
Sample point 129 predicted to be in Cluster 1
Sample point 130 predicted to be in Cluster 1
Sample point 131 predicted to be in Cluster 0
Sample point 132 predicted to be in Cluster 1
Sample point 133 predicted to be in Cluster 1
Sample point 134 predicted to be in Cluster 1
Sample point 135 predicted to be in Cluster 1
Sample point 136 predicted to be in Cluster 2
Sample point 137 predicted to be in Cluster 2
Sample point 138 predicted to be in Cluster 1
Sample point 139 predicted to be in Cluster 1
Sample point 140 predicted to be in Cluster 2
Sample point 141 predicted to be in Cluster 1
Sample point 142 predicted to be in Cluster 1
Sample point 143 predicted to be in Cluster 1
Sample point 144 predicted to be in Cluster 1
Sample point 145 predicted to be in Cluster 2
Sample point 146 predicted to be in Cluster 1
Sample point 147 predicted to be in Cluster 1
Sample point 148 predicted to be in Cluster 1
Sample point 149 predicted to be in Cluster 1
Sample point 150 predicted to be in Cluster 1
Sample point 151 predicted to be in Cluster 1
Sample point 152 predicted to be in Cluster 1
Sample point 153 predicted to be in Cluster 1
Sample point 154 predicted to be in Cluster 0
Sample point 155 predicted to be in Cluster 2
Sample point 156 predicted to be in Cluster 2
Sample point 157 predicted to be in Cluster 1
Sample point 158 predicted to be in Cluster 2
Sample point 159 predicted to be in Cluster 2
Sample point 160 predicted to be in Cluster 2
Sample point 161 predicted to be in Cluster 0
Sample point 162 predicted to be in Cluster 1
Sample point 163 predicted to be in Cluster 2
Sample point 164 predicted to be in Cluster 2
Sample point 165 predicted to be in Cluster 2
Sample point 166 predicted to be in Cluster 2
Sample point 167 predicted to be in Cluster 2
Sample point 168 predicted to be in Cluster 1
Sample point 169 predicted to be in Cluster 1
Sample point 170 predicted to be in Cluster 2
Sample point 171 predicted to be in Cluster 2
Sample point 172 predicted to be in Cluster 2
Sample point 173 predicted to be in Cluster 2
Sample point 174 predicted to be in Cluster 1
Sample point 175 predicted to be in Cluster 2
Sample point 176 predicted to be in Cluster 2
Sample point 177 predicted to be in Cluster 1
Sample point 178 predicted to be in Cluster 1
Sample point 179 predicted to be in Cluster 1
Sample point 180 predicted to be in Cluster 2
Sample point 181 predicted to be in Cluster 2
Sample point 182 predicted to be in Cluster 2
```

```
Sample point 183 predicted to be in Cluster 2
Sample point 184 predicted to be in Cluster 0
Sample point 185 predicted to be in Cluster 1
Sample point 186 predicted to be in Cluster 1
Sample point 187 predicted to be in Cluster 2
Sample point 188 predicted to be in Cluster 2
Sample point 189 predicted to be in Cluster 2
Sample point 190 predicted to be in Cluster 1
Sample point 191 predicted to be in Cluster 0
Sample point 192 predicted to be in Cluster 1
Sample point 193 predicted to be in Cluster 2
Sample point 194 predicted to be in Cluster 1
Sample point 195 predicted to be in Cluster 1
Sample point 196 predicted to be in Cluster 2
Sample point 197 predicted to be in Cluster 2
Sample point 198 predicted to be in Cluster 1
Sample point 199 predicted to be in Cluster 1
Sample point 200 predicted to be in Cluster 2
Sample point 201 predicted to be in Cluster 2
Sample point 202 predicted to be in Cluster 2
Sample point 203 predicted to be in Cluster 0
Sample point 204 predicted to be in Cluster 0
Sample point 205 predicted to be in Cluster 2
Sample point 206 predicted to be in Cluster 1
Sample point 207 predicted to be in Cluster 2
Sample point 208 predicted to be in Cluster 2
Sample point 209 predicted to be in Cluster 2
Sample point 210 predicted to be in Cluster 1
Sample point 211 predicted to be in Cluster 2
Sample point 212 predicted to be in Cluster 1
Sample point 213 predicted to be in Cluster 2
Sample point 214 predicted to be in Cluster 2
Sample point 215 predicted to be in Cluster 2
Sample point 216 predicted to be in Cluster 2
Sample point 217 predicted to be in Cluster 1
Sample point 218 predicted to be in Cluster 2
Sample point 219 predicted to be in Cluster 0
Sample point 220 predicted to be in Cluster 1
Sample point 221 predicted to be in Cluster 2
Sample point 222 predicted to be in Cluster 1
Sample point 223 predicted to be in Cluster 1
Sample point 224 predicted to be in Cluster 1
Sample point 225 predicted to be in Cluster 1
Sample point 226 predicted to be in Cluster 2
Sample point 227 predicted to be in Cluster 1
Sample point 228 predicted to be in Cluster 2
Sample point 229 predicted to be in Cluster 1
Sample point 230 predicted to be in Cluster 2
Sample point 231 predicted to be in Cluster 2
Sample point 232 predicted to be in Cluster 1
Sample point 233 predicted to be in Cluster 0
Sample point 234 predicted to be in Cluster 1
Sample point 235 predicted to be in Cluster 2
Sample point 236 predicted to be in Cluster 1
Sample point 237 predicted to be in Cluster 1
Sample point 238 predicted to be in Cluster 0
Sample point 239 predicted to be in Cluster 1
Sample point 240 predicted to be in Cluster 1
Sample point 241 predicted to be in Cluster 1
Sample point 242 predicted to be in Cluster 1
Sample point 243 predicted to be in Cluster 2
```

```
Sample point 244 predicted to be in Cluster 2
Sample point 245 predicted to be in Cluster 2
Sample point 246 predicted to be in Cluster 1
Sample point 247 predicted to be in Cluster 0
Sample point 248 predicted to be in Cluster 1
Sample point 249 predicted to be in Cluster 1
Sample point 250 predicted to be in Cluster 1
Sample point 251 predicted to be in Cluster 2
Sample point 252 predicted to be in Cluster 1
Sample point 253 predicted to be in Cluster 2
Sample point 254 predicted to be in Cluster 2
Sample point 255 predicted to be in Cluster 1
Sample point 256 predicted to be in Cluster 1
Sample point 257 predicted to be in Cluster 2
Sample point 258 predicted to be in Cluster 2
Sample point 259 predicted to be in Cluster 2
Sample point 260 predicted to be in Cluster 1
Sample point 261 predicted to be in Cluster 1
Sample point 262 predicted to be in Cluster 1
Sample point 263 predicted to be in Cluster 2
Sample point 264 predicted to be in Cluster 2
Sample point 265 predicted to be in Cluster 2
Sample point 266 predicted to be in Cluster 2
Sample point 267 predicted to be in Cluster 1
Sample point 268 predicted to be in Cluster 2
Sample point 269 predicted to be in Cluster 1
Sample point 270 predicted to be in Cluster 1
Sample point 271 predicted to be in Cluster 1
Sample point 272 predicted to be in Cluster 0
Sample point 273 predicted to be in Cluster 1
Sample point 274 predicted to be in Cluster 2
Sample point 275 predicted to be in Cluster 0
Sample point 276 predicted to be in Cluster 2
Sample point 277 predicted to be in Cluster 1
Sample point 278 predicted to be in Cluster 1
Sample point 279 predicted to be in Cluster 2
Sample point 280 predicted to be in Cluster 1
Sample point 281 predicted to be in Cluster 2
Sample point 282 predicted to be in Cluster 2
Sample point 283 predicted to be in Cluster 1
Sample point 284 predicted to be in Cluster 2
Sample point 285 predicted to be in Cluster 1
Sample point 286 predicted to be in Cluster 1
Sample point 287 predicted to be in Cluster 1
Sample point 288 predicted to be in Cluster 1
Sample point 289 predicted to be in Cluster 0
Sample point 290 predicted to be in Cluster 2
Sample point 291 predicted to be in Cluster 1
Sample point 292 predicted to be in Cluster 1
Sample point 293 predicted to be in Cluster 2
Sample point 294 predicted to be in Cluster 1
Sample point 295 predicted to be in Cluster 2
Sample point 296 predicted to be in Cluster 1
Sample point 297 predicted to be in Cluster 2
Sample point 298 predicted to be in Cluster 2
Sample point 299 predicted to be in Cluster 0
Sample point 300 predicted to be in Cluster 2
Sample point 301 predicted to be in Cluster 2
Sample point 302 predicted to be in Cluster 2
Sample point 303 predicted to be in Cluster 2
Sample point 304 predicted to be in Cluster 2
```

```
Sample point 305 predicted to be in Cluster 2
Sample point 306 predicted to be in Cluster 2
Sample point 307 predicted to be in Cluster 1
Sample point 308 predicted to be in Cluster 1
Sample point 309 predicted to be in Cluster 2
Sample point 310 predicted to be in Cluster 1
Sample point 311 predicted to be in Cluster 1
Sample point 312 predicted to be in Cluster 2
Sample point 313 predicted to be in Cluster 1
Sample point 314 predicted to be in Cluster 1
Sample point 315 predicted to be in Cluster 2
Sample point 316 predicted to be in Cluster 1
Sample point 317 predicted to be in Cluster 2
Sample point 318 predicted to be in Cluster 1
Sample point 319 predicted to be in Cluster 2
Sample point 320 predicted to be in Cluster 1
Sample point 321 predicted to be in Cluster 1
Sample point 322 predicted to be in Cluster 1
Sample point 323 predicted to be in Cluster 2
Sample point 324 predicted to be in Cluster 1
Sample point 325 predicted to be in Cluster 2
Sample point 326 predicted to be in Cluster 1
Sample point 327 predicted to be in Cluster 0
Sample point 328 predicted to be in Cluster 1
Sample point 329 predicted to be in Cluster 1
Sample point 330 predicted to be in Cluster 1
Sample point 331 predicted to be in Cluster 2
Sample point 332 predicted to be in Cluster 1
Sample point 333 predicted to be in Cluster 2
Sample point 334 predicted to be in Cluster 1
Sample point 335 predicted to be in Cluster 2
Sample point 336 predicted to be in Cluster 1
Sample point 337 predicted to be in Cluster 1
Sample point 338 predicted to be in Cluster 0
Sample point 339 predicted to be in Cluster 1
Sample point 340 predicted to be in Cluster 2
Sample point 341 predicted to be in Cluster 2
Sample point 342 predicted to be in Cluster 2
Sample point 343 predicted to be in Cluster 2
Sample point 344 predicted to be in Cluster 1
Sample point 345 predicted to be in Cluster 2
Sample point 346 predicted to be in Cluster 2
Sample point 347 predicted to be in Cluster 2
Sample point 348 predicted to be in Cluster 1
Sample point 349 predicted to be in Cluster 2
Sample point 350 predicted to be in Cluster 1
Sample point 351 predicted to be in Cluster 2
Sample point 352 predicted to be in Cluster 0
Sample point 353 predicted to be in Cluster 0
Sample point 354 predicted to be in Cluster 2
Sample point 355 predicted to be in Cluster 0
Sample point 356 predicted to be in Cluster 0
Sample point 357 predicted to be in Cluster 0
Sample point 358 predicted to be in Cluster 2
Sample point 359 predicted to be in Cluster 0
Sample point 360 predicted to be in Cluster 1
Sample point 361 predicted to be in Cluster 1
Sample point 362 predicted to be in Cluster 1
Sample point 363 predicted to be in Cluster 2
Sample point 364 predicted to be in Cluster 1
Sample point 365 predicted to be in Cluster 2
```

```
Sample point 366 predicted to be in Cluster 1
Sample point 367 predicted to be in Cluster 1
Sample point 368 predicted to be in Cluster 1
Sample point 369 predicted to be in Cluster 0
Sample point 370 predicted to be in Cluster 1
Sample point 371 predicted to be in Cluster 1
Sample point 372 predicted to be in Cluster 1
Sample point 373 predicted to be in Cluster 2
Sample point 374 predicted to be in Cluster 1
Sample point 375 predicted to be in Cluster 1
Sample point 376 predicted to be in Cluster 2
Sample point 377 predicted to be in Cluster 1
Sample point 378 predicted to be in Cluster 1
Sample point 379 predicted to be in Cluster 2
Sample point 380 predicted to be in Cluster 1
Sample point 381 predicted to be in Cluster 1
Sample point 382 predicted to be in Cluster 2
Sample point 383 predicted to be in Cluster 1
Sample point 384 predicted to be in Cluster 2
Sample point 385 predicted to be in Cluster 1
Sample point 386 predicted to be in Cluster 1
Sample point 387 predicted to be in Cluster 1
Sample point 388 predicted to be in Cluster 1
Sample point 389 predicted to be in Cluster 1
Sample point 390 predicted to be in Cluster 1
Sample point 391 predicted to be in Cluster 1
Sample point 392 predicted to be in Cluster 0
Sample point 393 predicted to be in Cluster 1
Sample point 394 predicted to be in Cluster 1
Sample point 395 predicted to be in Cluster 1
Sample point 396 predicted to be in Cluster 2
Sample point 397 predicted to be in Cluster 1
Sample point 398 predicted to be in Cluster 1
Sample point 399 predicted to be in Cluster 1
Sample point 400 predicted to be in Cluster 1
Sample point 401 predicted to be in Cluster 1
Sample point 402 predicted to be in Cluster 1
Sample point 403 predicted to be in Cluster 2
Sample point 404 predicted to be in Cluster 1
Sample point 405 predicted to be in Cluster 1
Sample point 406 predicted to be in Cluster 2
Sample point 407 predicted to be in Cluster 2
Sample point 408 predicted to be in Cluster 2
Sample point 409 predicted to be in Cluster 2
Sample point 410 predicted to be in Cluster 2
Sample point 411 predicted to be in Cluster 1
Sample point 412 predicted to be in Cluster 0
Sample point 413 predicted to be in Cluster 2
Sample point 414 predicted to be in Cluster 1
Sample point 415 predicted to be in Cluster 2
Sample point 416 predicted to be in Cluster 2
Sample point 417 predicted to be in Cluster 2
Sample point 418 predicted to be in Cluster 2
Sample point 419 predicted to be in Cluster 2
Sample point 420 predicted to be in Cluster 2
Sample point 421 predicted to be in Cluster 2
Sample point 422 predicted to be in Cluster 1
Sample point 423 predicted to be in Cluster 2
Sample point 424 predicted to be in Cluster 2
Sample point 425 predicted to be in Cluster 1
Sample point 426 predicted to be in Cluster 2
```

```
Sample point 427 predicted to be in Cluster 2
Sample point 428 predicted to be in Cluster 2
Sample point 429 predicted to be in Cluster 1
Sample point 430 predicted to be in Cluster 2
Sample point 431 predicted to be in Cluster 2
Sample point 432 predicted to be in Cluster 1
Sample point 433 predicted to be in Cluster 1
Sample point 434 predicted to be in Cluster 2
Sample point 435 predicted to be in Cluster 2
Sample point 436 predicted to be in Cluster 1
Sample point 437 predicted to be in Cluster 2
Sample point 438 predicted to be in Cluster 1
Sample point 439 predicted to be in Cluster 0
```

Answer: the prediction is consistency with the cluster from Question 8

Conclusion

Question 10

Companies often run [A/B tests (https://en.wikipedia.org/wiki/A/B_testing)](https://en.wikipedia.org/wiki/A/B_testing) when making small changes to their products or services. If the wholesale distributor wanted to change its delivery service from 5 days a week to 3 days a week, how would you use the structure of the data to help them decide on a group of customers to test?
Hint: Would such a change in the delivery service affect all customers equally? How could the distributor identify who it affects the most?

Answer: the delivery service won't affect all customers equally because of it's depends on each customers segments. If you use Unsupervised to seperate customers into segments I've recommend their to test the segments that have a high buy in quantity of Fresh food and this segments will affects the most because they're interesting in faster delivery.

Question 11

Assume the wholesale distributor wanted to predict a new feature for each customer based on the purchasing information available. How could the wholesale distributor use the structure of the clustering data you've found to assist a supervised learning analysis?
Hint: What other input feature could the supervised learner use besides the six product features to help make a prediction?

Answer: 6 products of features will use for segment unsupervised customers. We need the features that relevant for supervised learning such as

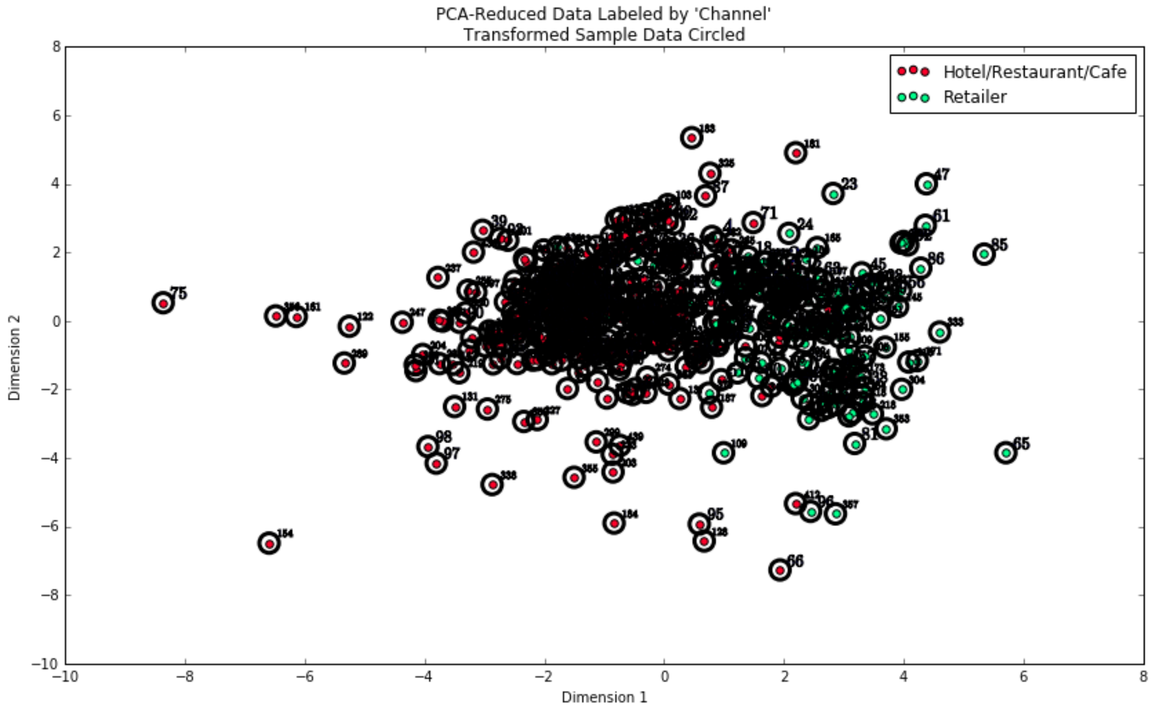
Visualizing Underlying Distributions

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 on 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 [127]:

```
# Display the clustering results based on 'Channel' data
rs.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: a bit different while the number show that we can segment into 3 segments but from this graph we can only divided it into 2 segments. There is not obvious the seperate data into 2 segments while in the middle of the data. I would consider these classification is not consistent that much due to data can't purely divide it.

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.