Machine_Learning_Customer_Segments_for_wholesale_distributor

August 20, 2018

1 Machine Learning Engineer Nanodegree

1.1 Unsupervised Learning

1.2 Project: Creating Customer Segments

1.3 Getting Started

In this project, we 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. 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.

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.".format(*da
except:
    print("Dataset could not be loaded. Is the dataset missing?")
```

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

1.4 Data Exploration

In this section, we 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 we will track through the course of this project.

Note that the dataset is composed of six important product categories: **'Fresh'**, **'Milk'**, **'Grocery'**, **'Frozen'**, **'Detergents_Paper'**, and **'Delicatessen'**.

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

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

       Detergents_Paper  Delicatessen
count        440.000000    440.000000
mean        2881.493182   1524.870455
std         4767.854448   2820.105937
min            3.000000      3.000000
25%          256.750000    408.250000
50%          816.500000    965.500000
75%         3922.000000   1820.250000
max        40827.000000  47943.000000
```

1.4.1 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, we add **three** indices of our choice to the `indices` list which will represent the customers to track. It is suggested to try different sets of samples until we 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 = [3,87,200]

        # Create a DataFrame of the chosen samples
        samples = pd.DataFrame(data.loc[indices], columns = data.keys()).reset_index(drop = True
```

1.4.2 Question

Consider the total purchase cost of each product category and the statistical description of the dataset above for our sample customers.

- What kind of establishment (customer) could each of the three samples we've chosen represent?

The mean values are as follows:

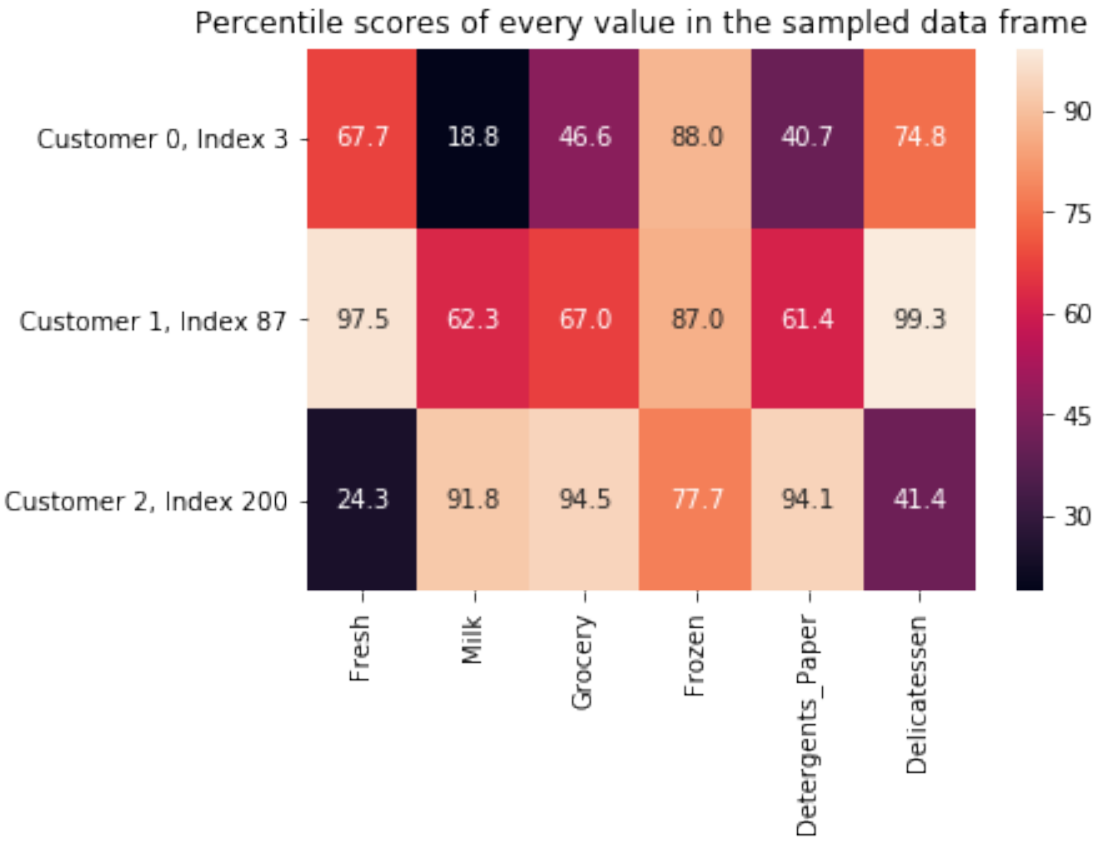
- Fresh: 12000.2977
- Milk: 5796.2
- Grocery: 7951.277273
- Frozen: 3071.9
- Detergents_paper: 2881.4
- Delicatessen: 1524.8

Knowing this, how do our samples compare? Does that help in driving our insight into what kind of establishments they might be?

```
In [6]: import seaborn as sns
        import matplotlib.pyplot as plt

        #Percentile values of the the sampled data
        percentile_values = 100. *data.rank(axis=0, pct=True).iloc[indices].round(decimals=3)
        #heatmap of percentiled value
        sns.heatmap(data=percentile_values,annot=True,fmt='.1f')
        plt.yticks([0.5,1.5,2.5],['Customer 0, Index '+str(indices[0]),'Customer 1, Index '+str(
        plt.title('Percentile scores of every value in the sampled data frame')

        print("Chosen samples of wholesale customers dataset:")
        display(samples)
```

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

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	13265	1196	4221	6404	507	1788
1	43265	5025	8117	6312	1579	14351
2	3067	13240	23127	3941	9959	731



Answer:

- CUSTOMER 0:
  - As we can see from the above heatmap, the maximum spending of this customer is for Fresh products with a spending of 13265 and score of 67.7th percentile, then comes the Frozen products with a spending of 6404 and score of 74.8th percentile and finally delicatessen products comes after this with a spending of 1788 and score of 67.7th percentile. Now even though we have a high percentile value for the delicatessen products, the spending in actual money is quite less, whereas the spending in actual currency for fresh and frozen products is substantially high.
  - One the other hand the spending for other categories like milk,grocery and detergents_papers is below 50th percentile. This points points to the fact that 'CUSTOMER 0' might be a small retail owner who sells fresh and frozen products with a small side shop which provides limited amount of delicatessen products like sandwiches etc.
- CUSTOMER 1:
  - The above heatmap shows that the spending of 'CUSTOMER 1' is more than 60th percentile in all of the categories.

  - Infact 'CUSTOMER 1' spends the maximum amount of money in fresh products with spending and percentile score of 43265 and 97.5 respectively. The second highest spending is in Delicatessen products with spendinng and percentile score of 14351 and 99.3 respectively. The third is frozen products with 6312 money spent anually and a percentile score of 87.0. Also the other categories like milk, grocery and detergent_paper have a subtantially high percentile score with the scores being 62.3, 67 and 61.4 respectively.
  - Such a trend can bee seen in big resturants who provide both breakfast, lunch and dinner.
    * This is because people tend to have coffe, milk products, egg dishes etc in breakfast which accounts for the above average spending in milk products and grocery.
    * Also since we have predicted it to be a big resturant it will have an above average spending in detergents to wash the dishes and also above average spending in tissue papers which accounts for the affore mentioned percentile score of category detergent_papar.
    * In lunch and dinner people tend to have more things made up of vegetables and meats(which are usually frozen), and this is the time when people usually tend to come in large numbers. People have all sorts of things during lunch and dinner, ranging from all vegan meal, salads, meat dishes, fruit dishes etc. This accounts for the very high spending in fresh and frozen products.
    * The spending in delicatessen is also high(at a 99.3th percentile) which points to the fact that this might be a continental resturant which serves a variety of food.
  - These facts mentioned above makes us conclude that 'CUSTOMER 1' is probably a big famous continental resturant.

- CUSTOMER 2:
  - The heat map and table above indicates that the maximum spending is done in categories groceries, milk and detergent_paper with spending and percentile score being (23127,94.5%), (13240,91.8%), (9959,94.1%) respectively.
  - The spending and percentile score in frozen is above average with values (3941,77.7%) and the spending for fresh and delicatessen products is relatively low.
  - Such a trend can be seen in coffee shops or break fast resturants. But its more likely that it is a coffee shop.
    * The maximum use of milk is done in a cafe. Also we hardly see a cafe that sells coffee exclusively. It always or most of the times comes with a section where they sell food. This two statistics accounts for the high spending in milk and groceries.
    * The other high spending is in detergent and papers. This suggests that 'CUSTOMER 2' owns a coffee house that works the entire day or maybe 24X7, only then can we justify such high spending in detergents and papers. This fact also provides a concrete fact that 'CUSTOMER 2' does not own a break-fast resturant as such resturants a open only in the morning.
    * The relatively low spending in other categories makes our assumption even more concrete that this customer is to a very high degree an owner of a cafe.

1.4.3 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 [7]: # TODO: Make a copy of the DataFrame, using the 'drop' function to drop the given featur
        from sklearn.model_selection import train_test_split
        from sklearn.tree import DecisionTreeRegressor
        target_features = ['Fresh', 'Milk', 'Grocery', 'Frozen', 'Detergents_Paper', 'Delicatess

        for target in target_features:
            y_target = data[target]
            new_data = data.drop([target], axis = 1, inplace = False)

            # TODO: Split the data into training and testing sets using the given feature as the
            X_train, X_test, y_train, y_test = train_test_split(new_data, y_target, test_size=0.

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

            regressor = DecisionTreeRegressor(random_state=0)
            regressor.fit(X_train, y_train)

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

```
score for target feature  Fresh  is  -0.252469807688
score for target feature  Milk  is  0.365725292736
score for target feature  Grocery  is  0.602801978878
score for target feature  Frozen  is  0.253973446697
score for target feature  Detergents_Paper  is  0.728655181254
score for target feature  Delicatessen  is  -11.6636871594
```

1.4.4 Question

- Which feature did we attempt to predict?
- What was the reported prediction score?
- Is this feature necessary for identifying customers' spending habits?

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. If we get a low score for a particular feature, that lends us to beleive that that feature point is hard to predict using the other features, thereby making it an important feature to consider when considering relevance.

Answer: * Since there are just features to consider, we selected all the features as target variables one by one and their respective scores are as mentioned above. * The reported prediction scores are:

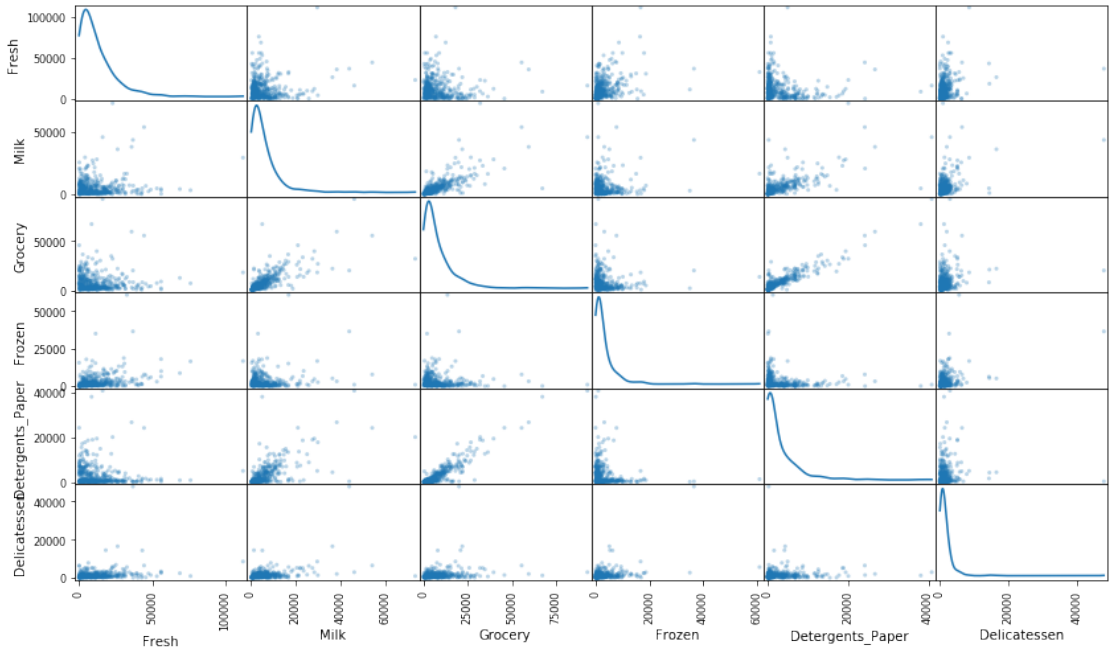
target feature	score
Fresh	-0.2525
Milk	0.3657
Grocery	0.6028
Frozen	0.2539
Detergent_paper	0.7287
Delicatessen	-11.6637

- The feature necessary fro identifying customer segments are selected as follows:
  - As shown in the above table there are two target variables 'Fresh' and 'Delicatessen' that have an `R^2` score which is less than zero. This means that the model fails to fit the data with the when 'Fresh' and 'Delicatessen' are taken as target variables.
  - On the other hand the two features 'Grocery' and 'Detergent_paper' have high `R^2` score which means that there is a very high correlation between 'Grocery' and 'Detergent_paper' and other features. Hence 'Grocery' can be predicted using other features. Thus these features do not provide good insight for identifying customers' spending segment.
  - However there are two features 'Milk' and 'Frozen' which when used as target variable have very little `R^2` which points to the fact that these feature do not have much correlation with other features and as a result of this, these features can be used for identifying customers' spending habits.

1.4.5 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.

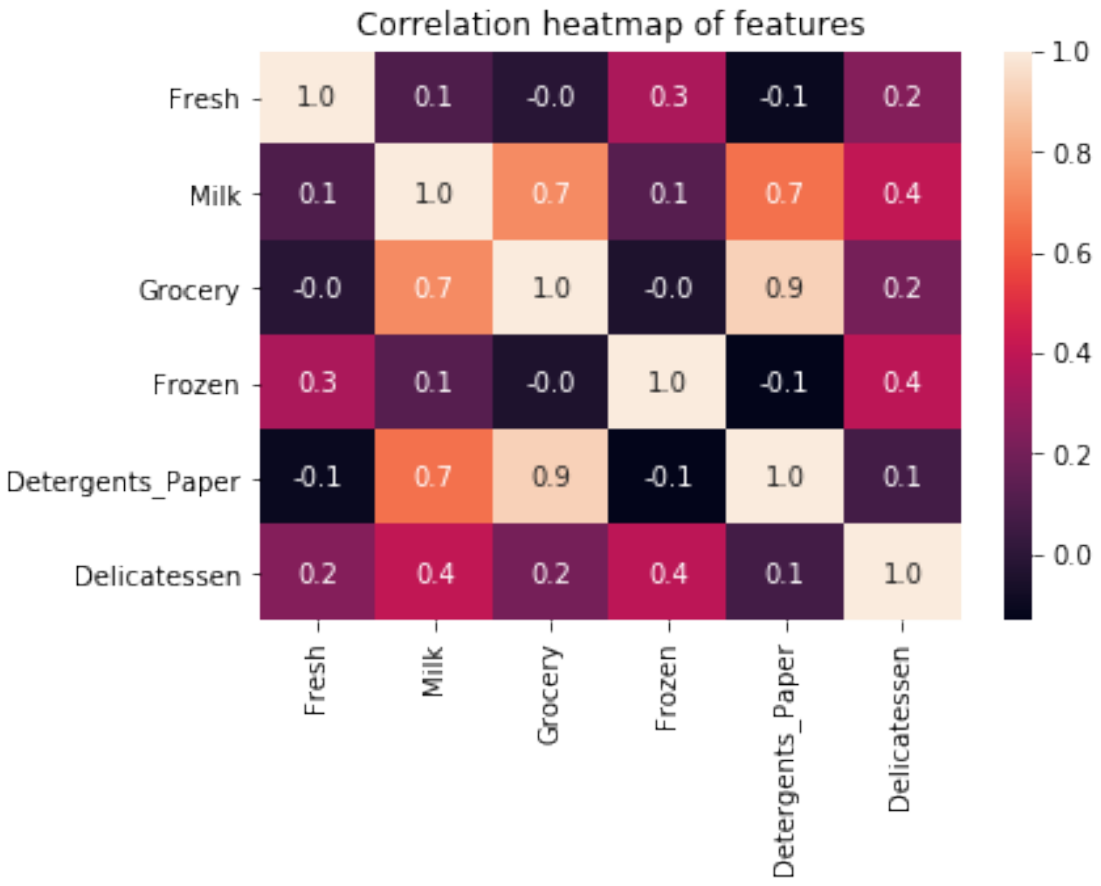
```
In [8]: # Produce a scatter matrix for each pair of features in the data
        pd.plotting.scatter_matrix(data, alpha = 0.3, figsize = (14,8), diagonal = 'kde');
```



1.4.6 Question

- Using the scatter matrix as a reference, discuss the distribution of the dataset, specifically talk about the normality, outliers, large number of data points near 0 among others. If you need to sepearate out some of the plots individually to further accentuate your point, you may do so as well.
- 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?

```
In [9]: sns.heatmap(data=data.corr(),annot=True,fmt='.1f')
        plt.title('Correlation heatmap of features')
        plt.show()
```



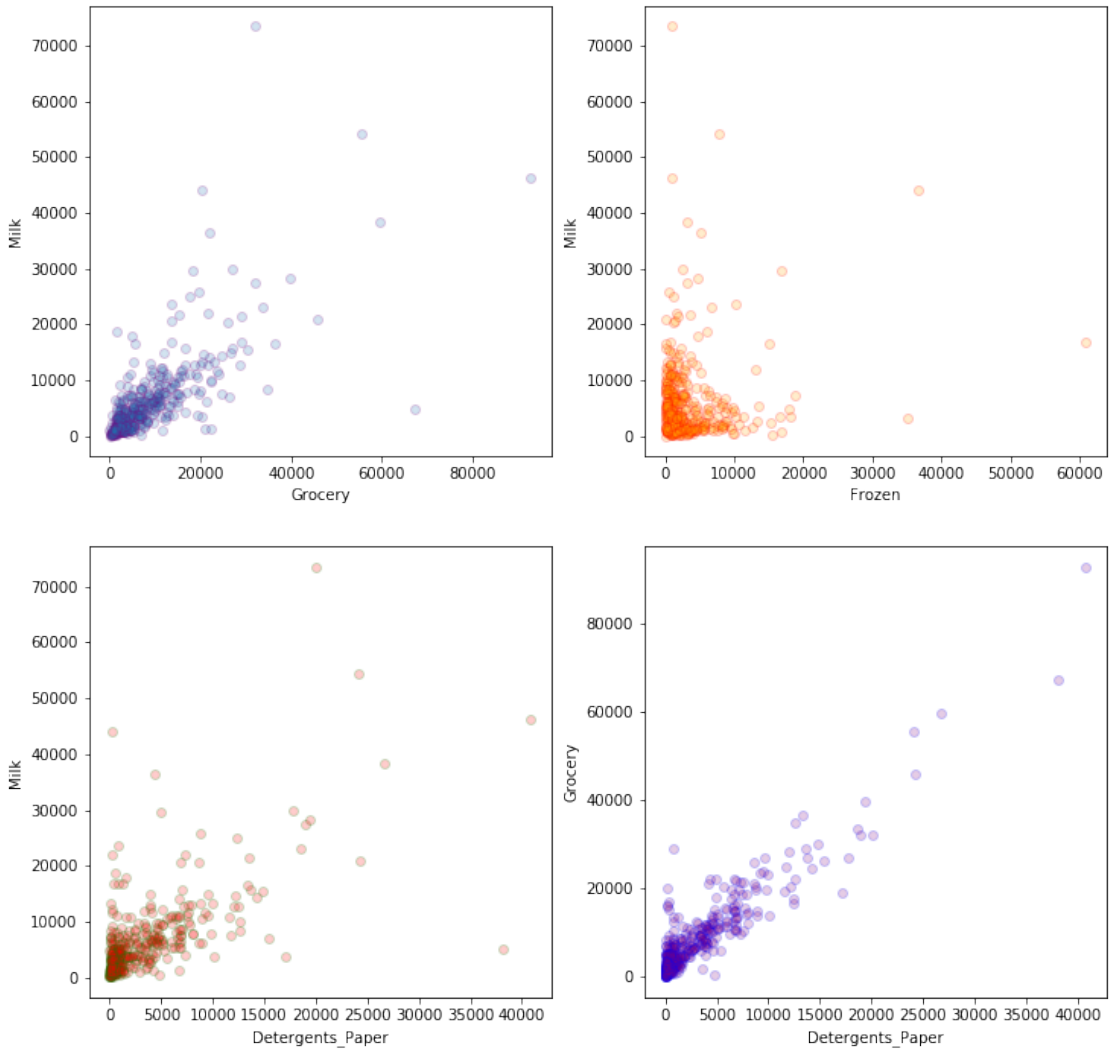
Answer: * From the graphs above we can say that most of the data points are concenterated near zero i.e below the mean. And as a result of this almost all the distributions are right skewed. Also since the distributions are right skewed there are outliers and they will always be toward the right side i.e the customers who spend extrems amounts of money will be very less. * We are going to neglect the features 'Fresh' and 'Delicatessen' as the R^2 score was negative which points to the fact that those two features do not fit the model and are useless. - Thus we concenterate on the other 4 features('Milk','Frozen','Grocery' and 'Detergents_Paper'). - Take a look at the heat-map shown above in whcih each cell represents the correlation of the feature corresponding to the particular row and column. (We won't be considering the auto-correlation of features when we look at the heat map above, we will only be considering the cross-correlations.).

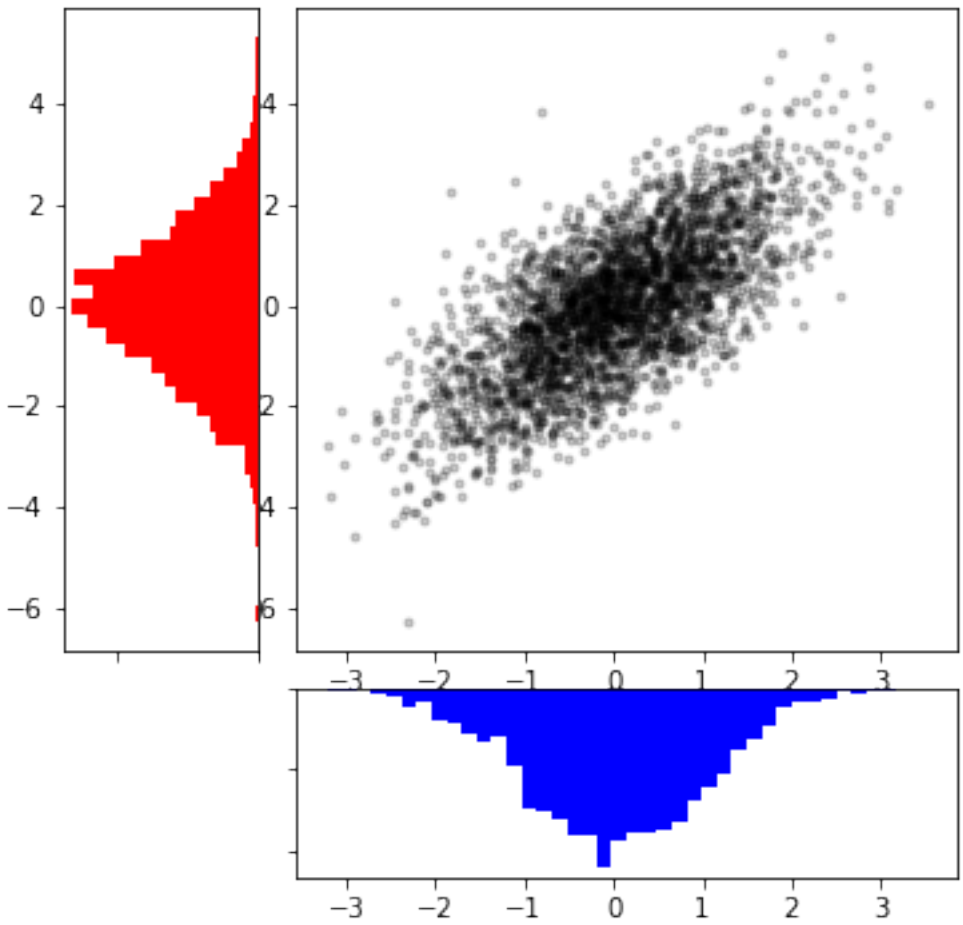
```
- As we can see, the feature 'Milk' has highest correlation with features 'Grocery' and 'Deterg
- The feature 'Grocery' has the maximum correlation with feature 'detergents_Paper' and minimum
- 'Frozen' has minimum correlation with 'Milk'.

- Thus we included all the other four features we are interested in. Now we will take a look at
```

```
In [10]: f, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2,figsize=(12,12))
         f.suptitle('The magnified subplots of useful variables')
         ax1.scatter(np.array(data['Grocery']),np.array(data['Milk']),alpha=0.2,edgecolors='purp
```

```
ax1.set(xlabel='Grocery', ylabel='Milk')
ax2.scatter(np.array(data['Frozen']),np.array(data['Milk']),c='orange',alpha=0.2,edgeco
ax2.set(xlabel='Frozen', ylabel='Milk')
ax3.scatter(np.array(data['Detergents_Paper']),np.array(data['Milk']),c='red',alpha=0.2
ax3.set(xlabel='Detergents_Paper', ylabel='Milk')
ax4.scatter(np.array(data['Detergents_Paper']),np.array(data['Grocery']),c='purple',alp
ax4.set(xlabel='Detergents_Paper', ylabel='Grocery')
plt.show()

# Create some normally distributed data
mean = [0, 0]
cov = [[1, 1], [1, 2]]
x, y = np.random.multivariate_normal(mean, cov, 3000).T

# Set up the axes with gridspec
fig = plt.figure(figsize=(6, 6))
grid = plt.GridSpec(4, 4, hspace=0.2, wspace=0.2)
main_ax = fig.add_subplot(grid[:-1, 1:])
y_hist = fig.add_subplot(grid[:-1, 0], xticklabels=[], sharey=main_ax)
x_hist = fig.add_subplot(grid[-1, 1:], yticklabels=[], sharex=main_ax)

# scatter points on the main axes
main_ax.plot(x, y, 'ok', markersize=3, alpha=0.2)

# histogram on the attached axes
x_hist.hist(x, 40, histtype='stepfilled',
            orientation='vertical', color='blue')
x_hist.invert_yaxis()

y_hist.hist(y, 40, histtype='stepfilled',
            orientation='horizontal', color='red')
y_hist.invert_xaxis()
```

The magnified subplots of useful variables





- The first plot tells us that 'Grocery' and milk are correlated to some extent as with the increase in value of 'Grocery' the value of 'Milk' tends to increase. The correlation value is 0.7 which is pretty high. Thus if we know value of one of the features we can predict the value of the other one.
  - Also since `R^2` score of 'Milk' is less as compared to 'Grocery' we can neglect the 'Grocery' feature and use the feature 'Milk' to create customer segments.
- The second plot is the plot of 'Milk' and 'Frozen' and apparently the values are spread throughout the space. This is verified by the correlation coffecient between them being 0.1. Such small correlation coffecient and a small `R^2` score suggest that Frozen is indeed an important variable to create customer segments.
- The third plot suggest that there is a good correlation between 'Milk' and 'Detergents_Paper' and hence as the reasons provided in the points above we can neglect 'Detergents_Paper'.
- The fourth graph is the most important graph. We might argue that since the correlation between 'Frozen' and 'Grocery', and 'Frozen' and 'Detergents_Paper' is zero it would be

wise to include 'Grocery' and 'Detergents_Paper' as a feature to create customer segments. However we can provide two counter arguments to this which are as follows:

- The R^2 score that we calculated before was high for 'Grocery' as well as 'Detergents_Papers' which indicated that the above two features can be predicted using the other features. Thus neglecting these two features is a good choice.
- The other reason is that if we look at the fourth graph between 'Grocery' and 'Detergents_Paper' it seems that both the values come from a normal distribution with almost the same mean and standard deviation(for better visualization look at the last graph) and this means that knowing the value of one helps us preict the value of other. Also in the first point we concluded that since there is a correlation between 'Milk' and 'Grocery' we can drop 'Grocery' as knowing the value of 'Milk' will help us predict the value of 'Grocery' and owing to this fact we can also drop 'Detergents_Paper' as it comes from the same distribution as that of 'Grocery'.

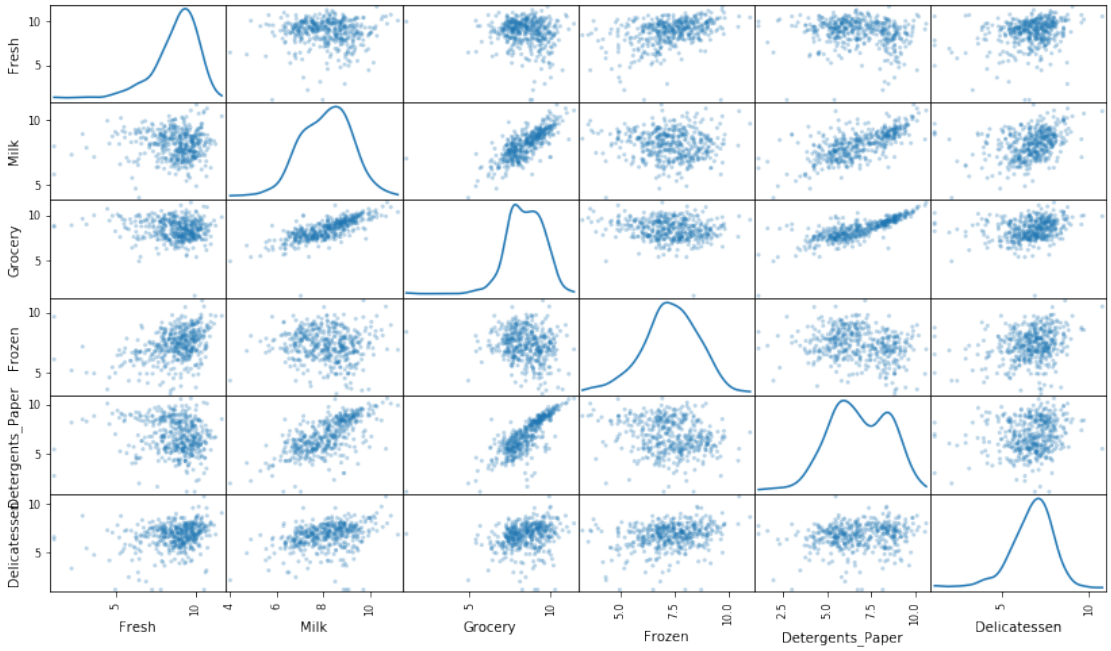
1.5 Data Preprocessing

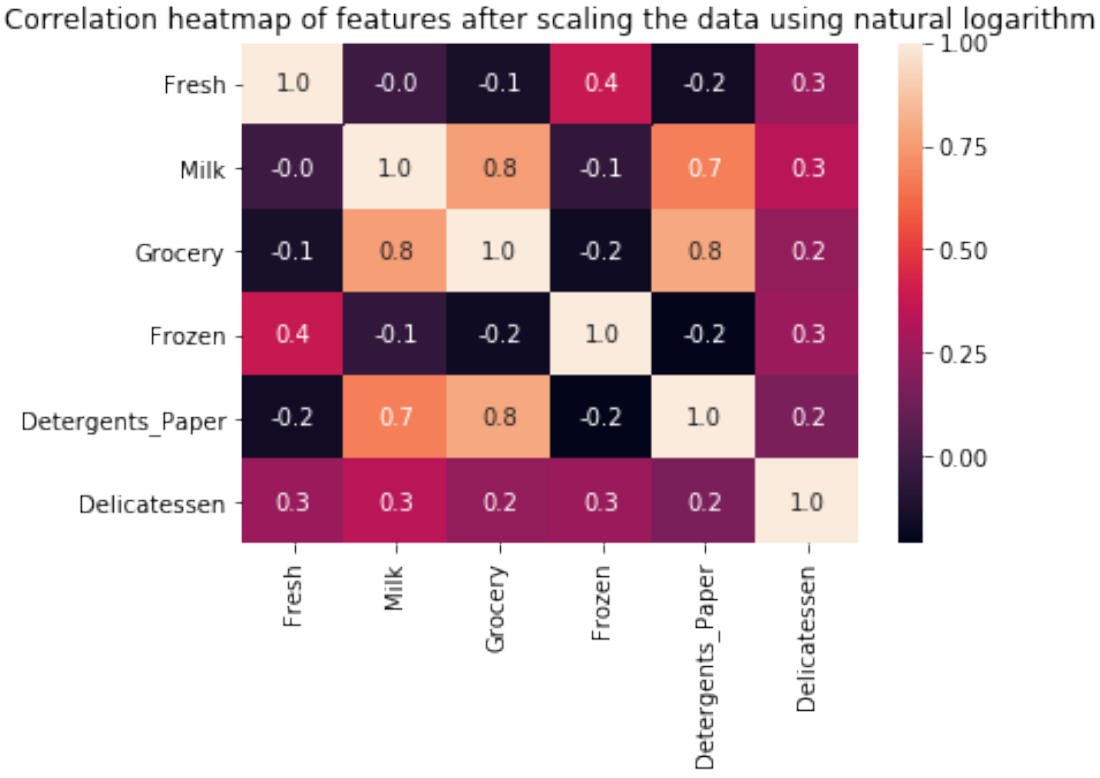
In this section, we will preprocess the data to create a better representation of customers by performing a scaling on the data and detecting (and optionally removing) outliers. Preprocessing data is often times a critical step in assuring that results we obtain from your analysis are significant and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

1.5.1 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 to apply a non-linear scaling — particularly for financial data. One way to achieve this scaling is by using a Box-Cox test, 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 [11]:
```
# 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 = 'kde');
plt.show()

sns.heatmap(data=log_data.corr(),annot=True,fmt='.1f')
plt.title('Correlation heatmap of features after scaling the data using natural logarit
plt.show()
```


Fresh
Milk
Grocery
Frozen
Detergents_Paper
Delicatessen


Correlation heatmap of features after scaling the data using natural logarithm
Fresh 1.0 -0.0 -0.1 0.4 -0.2 0.3
Milk -0.0 1.0 0.8 -0.1 0.7 0.3
Grocery -0.1 0.8 1.0 -0.2 0.8 0.2
Frozen 0.4 -0.1 -0.2 1.0 -0.2 0.3
Detergents_Paper -0.2 0.7 0.8 -0.2 1.0 0.2
Delicatessen 0.3 0.3 0.2 0.3 0.2 1.0
Fresh
Milk
Grocery
Frozen
Detergents_Paper
Delicatessen
1.00
0.75
0.50
0.25
0.00

1.5.2 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 we 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).

In [12]:
```
# Display the log-transformed sample data
display(log_samples)
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	9.492884	7.086738	8.347827	8.764678	6.228511	7.488853
1	10.675099	8.522181	9.001716	8.750208	7.364547	9.571575
2	8.028455	9.490998	10.048756	8.279190	9.206232	6.594413

1.5.3 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: 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 [13]:
```
# For each feature find the data points with extreme high or low values
feature_out_all = {}
all_outliers = set()
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 int
    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] <= Q3 + st
    current_feat_outliers = list((log_data.index[~((log_data[feature] >= Q1 - step) & (
    all_outliers = all_outliers.union(set(current_feat_outliers))
    for index in current_feat_outliers:
        if(feature_out_all.get(index) == None):
            feature_out_all[index] = [feature]
        else:
            feature_out_all[index].append(feature)
```

```
        outliers = []

        for key in feature_out_all:
            if(len(feature_out_all[key]) > 1):
                outliers.append(key)

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

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

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

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

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

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

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

Data points considered outliers for the feature 'Frozen':

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

Data points considered outliers for the feature 'Detergents_Paper':

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

Data points considered outliers for the feature 'Delicatessen':

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

     Delicatessen
```

```
66       3.295837
109      1.098612
128      1.098612
137      3.583519
142      1.098612
154      2.079442
183     10.777768
184      2.397895
187      1.098612
203      2.890372
233      1.945910
285      2.890372
289      3.091042
343      3.610918
```

```
In [14]: for key in feature_out_all:
             if(len(feature_out_all[key]) > 1):
                 print("The point with index ",key," occurs as outlier in features ",feature_out
```

```
The point with index  65  occurs as outlier in features  ['Fresh', 'Frozen']
The point with index  66  occurs as outlier in features  ['Fresh', 'Delicatessen']
The point with index  128  occurs as outlier in features  ['Fresh', 'Delicatessen']
The point with index  154  occurs as outlier in features  ['Milk', 'Grocery', 'Delicatessen']
The point with index  75  occurs as outlier in features  ['Grocery', 'Detergents_Paper']
```

1.5.4 Question

- 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?

Answer: * Yes there are data points considered outliers for more than one feature and they are displayed in the above code. * Yes these data points should be removed from the dataset because they posses characteristics which is different from bulk of the data and this unusual characteristic can affect our prediction to a great extent. * Data points are added to the outliers list because these data points posses certain characteristics which the bulk doesn't posses. - For instance consider a class of 10 students and we are supposed to find the average intelligence of this class. We aim at doing this by conducting a test. Consider the following table which shows the exam scores, scored out of 100 for 10 students in the previously mentioned test.We will make the assumption that all the students are sincere, taught by the same teacher and have a high IQ which was measured upon their admission owing to the highly selective nature of this imaginary class.

```
- As shown in the table and because of the assumptions made, since maximum students have scored
```

Student_id	exam_score
1	97
2	99
3	94
4	5
5	98
6	2
7	96
8	100
9	100
10	99
Class_avg without removing outliers	79
Class_avg after removing outliers	97.875

- Thus all in all outliers are removed so that they do not have adverse effects on our prediction and these outliers are on the two left and right edges of the data which is `1.5 * IQR` units away from the middle 50% data. This convention of chosing a data which is `1.5 * IQR` times away helps to ensure the fact that no important data is lost and only the data which is the oddest is removed.

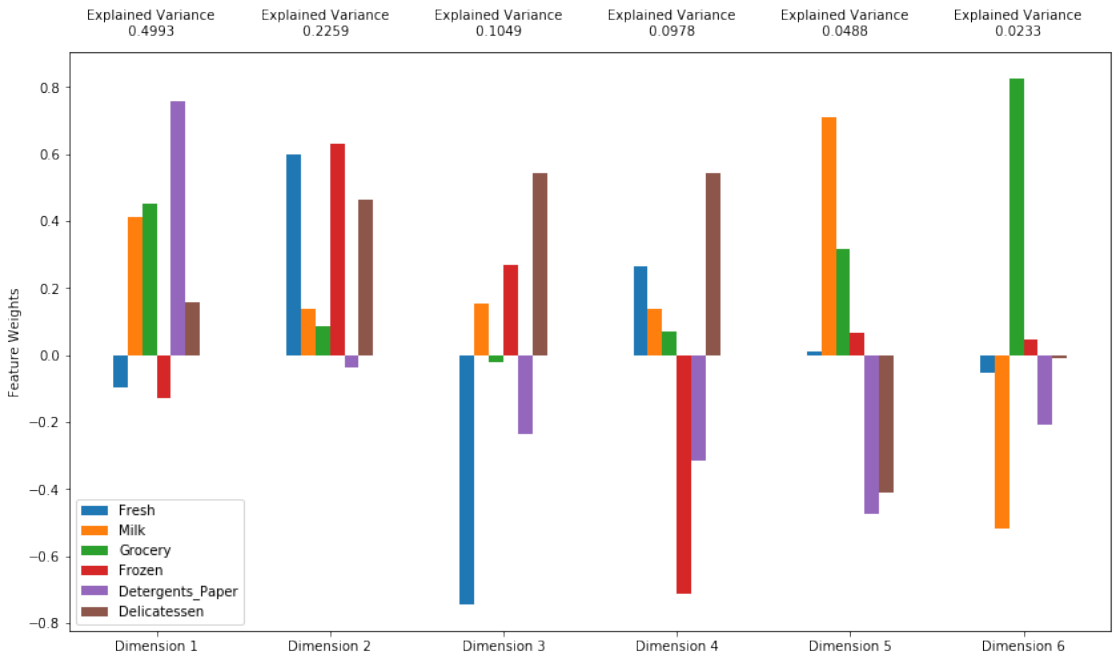
1.6 Feature Transformation

In this section we 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.

1.6.1 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 [15]: from sklearn.decomposition import PCA
         # TODO: Apply PCA by fitting the good data with the same number of dimensions as featur
         pca = PCA(n_components=6)
         pca.fit(good_data)

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

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



1.6.2 Question

- How much variance in the data is explained* **in total** *by the first and second principal component?
- How much variance in the data is explained by the first four principal components?
- Using the visualization provided above,we will talk about each dimension and the cumulative variance explained by each, stressing upon which features are well represented by each dimension(both in terms of positive and negative variance explained).We will discuss what the first four dimensions best represent in terms of customer spending.

Note: 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 in the data explained by first and second principal component is:
  - 0.4993+0.2259 = 0.7252 (72.52%)
- The total variance in the data explained by first four principal component is:
  - 0.4993+0.2259+0.1049+0.0978 = 0.9279 (92.79%)
- If we look closely at the graph above we can see that there is an explained variance for each dimension generated by the PCA. This explained variance tells us how much information is retained in a particular dimension when we change the basis of our original space. The

feature weights for a particular dimension gives us insight about how much a customer belonging to that particular dimension will spend on a particular product. Due to all these reasons we can conclude that explained variance and feature weights together are describing the spending patterns of customer rather than representing the customers.

  - Looking at **dimension 1** which accounts for **49.93%** of the entire variance or **49.93%** of the spending pattern, we can see that `Milk`, `Grocery`, `Detergents_Paper` and `Delicatessen` have feature weights that are positive while all the other features have weights that are negative.
    * The positive feature weights of `Milk`, `Grocery`, `Detergents_Paper` and `Delicatessen` mean that customer that are concenterated in dimension 1 will be spending more on these category of products. Maximum spending will be done for `Detergents_Paper` as the feature weight corresponding to it is the maximum and in the same way the second highest spending for customers clustered near dimension 1 would be for feature `Grocery` as the feature weight corresponding to it is second highest , thierd would be `Milk` and fourth would be `Delicatessen`.
    * The negative feature weights of all the other features indicate that the customers in dimension 1 will avoid buying products from `Fresh` and `Frozen`.

    * *Thus we can generate a heurestic that whatever the dimension is the pattern of customer spending will be such that spending on a particular category of product is directly proportional to the feature weights corresponding to that feature. Thus lower the feature weight of a category lower is the spending on that category and if the feature weights are negative, the customer avoids spending on such product category alltogether and if there is any its very less.*

- **Dimension 2** ranks second in accounting for the explained variance and thus the spending pattern with a value of **22.59%**.

  - In this dimension all the feature weights are positive except `Detergents_Paper`. Thus spending will in descending order as follows: `Frozen`,`Fresh`,`Delicatessen`,`Milk` and the lest spending would on `Grocery`
  - The feature with negative feature weight `Detergents_Paper` would be neglected alltogether by the customers clustered near this dimension.

- **Dimension 3** comes next with an explained ratio of **10.49%**.

  - As we can see from the figure above `Delicatessen`, `Frozen` and `Milk` have positive feature weights with their values descending in the above mentioned order. Thus customers concenterated near this dimension will be spending the maximum in the three features mentioned above. Eventhough the feature weight of milk is positive, its value is very less and hence the spending on milk wont be very high, but ther will still be some spending for `Milk`.
  - All the other categories have negative feature weights and hence the customers in this dimension avoid buying these categories with negative feature weights.

- The fourth dimension **Dimension 4** accounts for **9.78%** of the spending pattern.

  - The customers in this dimension spend the highest in `Delicatessen`, `Fresh`, `Milk` and `Grocery` prodcuts with their values descending in above mentioned order. `Milk` and `Grocery` with positive but low feature weights will have a low spending from the customers in this dimension.
  - Finally the categories with negative feature weights which are `Frozen` and `Detergents_Paper` will be avoided by the customers all together.

- It is to be noted that the spending patterns are unique to each dimenion. The spending patterns in one dimension will not repeat itself in other dimension.

1.6.3 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.

```
In [16]: # Display sample log-data after having a PCA transformation applied

         print('PCA transformed dimension of three samples')
         display(pd.DataFrame(np.round(pca_samples, 4), columns = pca_results.index.values))
```

```
PCA transformed dimension of three samples
```

```
   Dimension 1  Dimension 2  Dimension 3  Dimension 4  Dimension 5  \
0      -0.9986       1.3694       0.2854      -0.3997      -0.6781
1       0.9642       3.2453       0.4714       0.9451      -0.8370
2       3.0820       0.1314       0.3994      -1.4197       0.4747

   Dimension 6
0       0.6194
1       0.0961
2       0.2263
```

1.6.4 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 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 [17]: # 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'])
```

1.6.5 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 [18]: # Display sample log-data after applying PCA transformation in two dimensions
         display(pd.DataFrame(np.round(pca_samples, 4), columns = ['Dimension 1', 'Dimension 2']
```

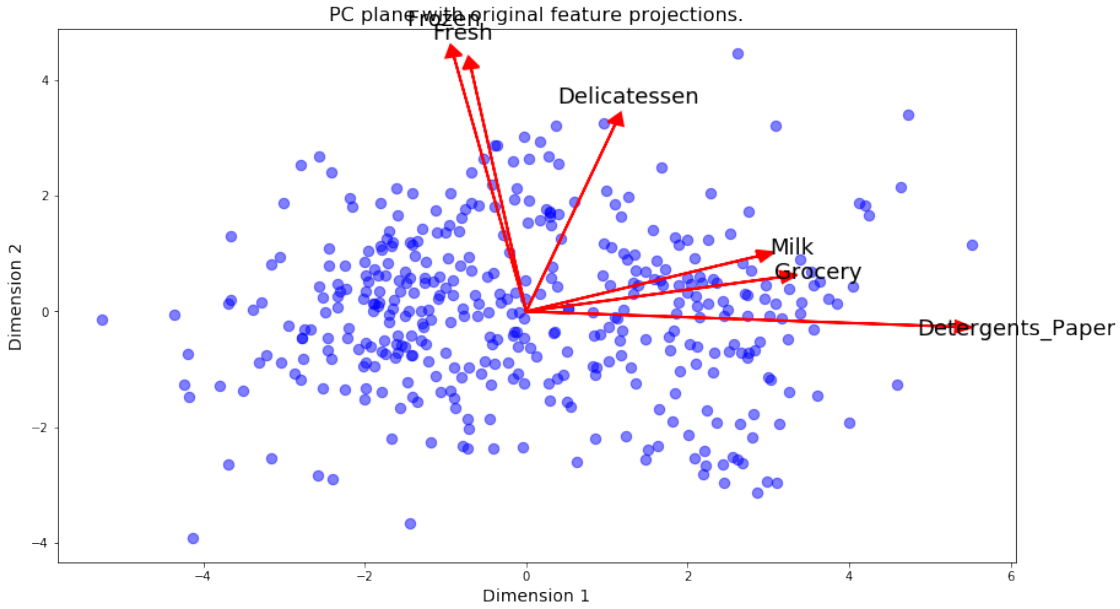
	Dimension 1	Dimension 2
0	-0.9986	1.3694
1	0.9642	3.2453
2	3.0820	0.1314

1.7 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 [19]: # Create a biplot
         vs.biplot(good_data, reduced_data, pca)

Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc1c0a21358>
```



1.7.1 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?

- Since we have plotte the original feature projections in red on the biplot, we can say that a feature is maximally correlated(+1) with a particular dimension when it is parallel to the dimension or is in the direction of the increase of that dimension. This is because with the increase in the value of dimension the feature pointing in that direction will also increase therby pointing to a positive correlation.
- In the same way if the feature is orthogonal to a particular dimension then the correlation is zero because no matter the direction of increase or decrease of that dimension the feature value corresponding to it will remain constsnt.
- Finally if the feature is pointing in the direction opposite(180 degrees) to the increase of a particular dimension then we can say that the particular feature is negatively correlated with the dimension in consideration and has minimum correlation(-1).

We can see from the above biplot and the original feature projections in red that there are three features in maximum correlation with dimension 1 with the following order of decreasing correlation: - Detergents_Paper(max_correlation with dimension 1) - Grocery (second best correlation with dimension 1) - Milk (Third best correlation with dimension 1)

In the same way the three features with maximum correlation with dimension 2 with their are as follows: - Frozen(max_correlation with dimension 2) - Fresh (second best correlation with dimension 2) - delicatessen (Third best correlation with dimension 2)

- The above mentioned points above correlation of features with `dimension 1` and `dimension 2` is proved by the PCA plot above where the feature weights of the respective features in `dimension 1` and `dimension 2` correspond to the correlations mentioned in points above.

1.8 Clustering

In this section, we 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. We will then recover specific data points from the clusters to understand their significance by transforming them back into their original dimension and scale.

- HARD AND SOFT CLUSTERING:
  - Hard Clustering: Hard clustering is a method where every point in the data is assigned to a particular cluster center. Morever a data point can belong to one and only one cluster center.
  - Soft Clustering: Unlike hard clustering where all data points belong to unique centers, the soft clustering method assign clusters on the basis of probability. This means that a data point will belong to all the cluster centers with some probability. After this depending on the probability of a data point being in a particular cluster we tag it with that cluster in which the probability of the data point being in that cluster is maximum. However this does not mean that the point into consideration does not belong to other cluster centers, it always has some probability associated with it however small it is.
- K-Means: This method of clustering is a hard clustering method. This means that every data point when clustered using a K-means will be assigned a unique cluster center. However the assignment depends highly on the initial selection of cluster centers. This is because K-means selects its initial centers randomly and after that it does the following steps:
  - It tries to reduce the squared distance between the cluster centers and the surrounding points and assigns new coordinates to the centers(OPTIMIZATION step).
  - After this the data points are assigned to these new clusters(ASSIGNMENT step).

  These two steps optimization and assignment are carried out again and again till there is no change in the coordinates of the cluster center. This is when we are done with clustering with K-means and we finally have our two clusters. However due to the initial random selection of cluster centers, if the data is spread out i.e there is no clear boundaries between the two clusters, K-means clustering will give us a different result everytime. These results may be what we want or they may not be what we expected. Emperical results suggests that K-means tends to work when we have our data set which is separated in such a way that, not only the clusters have a defining separating boundary, but also when the clusters are in the shape of circular blobs. Even when we have such conditions for our data it is not guranteed that K-means will cluster properly. The clustering quality depends very much on the initial selection of the cluster centers. Thus since our data is highly spread K-means clustering will not be a good choice because of the following reasons:

  - When we look at our dataset we can see that K-means will not make sense as there is no separating boundary. And as mentioned earlier K-means will not work properly when there isn't a proper separating boundary.
  - K-means is a hard clustering method and it will try to assign every point to a unique cluster. And since our data points are too spread out we dont want to be too harsh on assigning the points to a particluar center. Instead we might try to assign every point in the data to every cluster center with some probability which is nothing but soft clustering. The reason for going with soft clustering is that since the data is too spread out, we are uncertain about a particular point belonging to a specific category. Under such uncertain circumstances, what better way to define a datas' category with probability and likelihood.

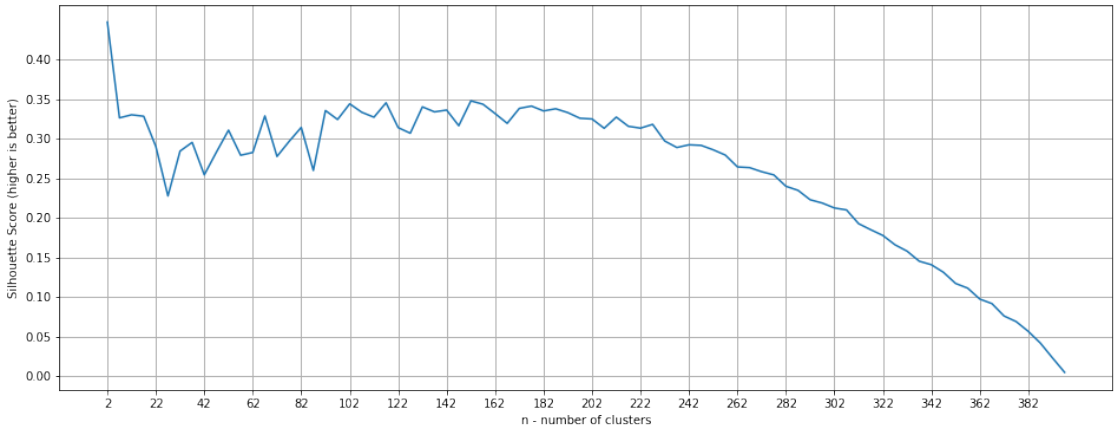
- Gaussian-Mixture Model: As the name suggests the Gaussian Mixture model uses multivariate normal probability distribution to cluster data. Depending on the number of clusters chosen initially this model assigns a specific probability to all the data points and this probability suggests how much a data point belongs to a specific cluster. A familiarity with the normal model suggests that the probability distribution never touches 0 instead it reduces exponentially, reaches very close to zero and touches the zero probability point only at infinity. Due to this fact there is never a zero probability of a data point being in any one of the cluster. Instead there is only a relative probability that suggests the data point being more or less in a particular cluster. This points to the fact that Gaussian-Mixture Model clustering is a soft clustering method.
  - Thus since our data is highly spread out we can think of using a gaussina clustering model and check its validity by using the silhouette score for different numbers of clusters.
- WE FINALLY CHOOSE THE GUSAASIAN MIXTURE MODEL CLUSTERING BECAUSE OF THE ABOVE DISCUSSION.

1.8.1 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 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 [20]:
```
# TODO: Apply your clustering algorithm of choice to the reduced data
from sklearn.mixture import GaussianMixture
from sklearn.metrics import silhouette_score

# Choose the range of k values to test.
possible_n_values = range(2, len(reduced_data)+1, 5)

#Calculating the silhouette_score for every possible value of clusters possible
errors_per_n = []
for n in possible_n_values:
```

```
        clusterer = GaussianMixture(n_components=n)
        clusterer.fit(reduced_data)
        preds = clusterer.predict(reduced_data)
        score = silhouette_score(reduced_data,preds)
        errors_per_n.append(score)

    # Plot the each value of n vs. the silhouette score at that value
    fig, ax = plt.subplots(figsize=(16, 6))
    ax.set_xlabel('n - number of clusters')
    ax.set_ylabel('Silhouette Score (higher is better)')
    ax.plot(possible_n_values, errors_per_n)

    # Ticks and grid
    xticks = np.arange(min(possible_n_values), max(possible_n_values)+1, 20.0)
    ax.set_xticks(xticks, minor=False)
    ax.set_xticks(xticks, minor=True)
    ax.xaxis.grid(True, which='both')
    yticks = np.arange(round(min(errors_per_n), 2), max(errors_per_n), .05)
    ax.set_yticks(yticks, minor=False)
    ax.set_yticks(yticks, minor=True)
    ax.yaxis.grid(True, which='both')
```



In [21]:
```
clusterer = GaussianMixture(n_components=2)
clusterer.fit(reduced_data)
preds = clusterer.predict(reduced_data)
score = silhouette_score(reduced_data,preds)
centers = clusterer.means_
sample_preds = clusterer.predict(pca_samples)
print('Highest silhouette score is with 2 cluster centers and its value is: ',score)
```

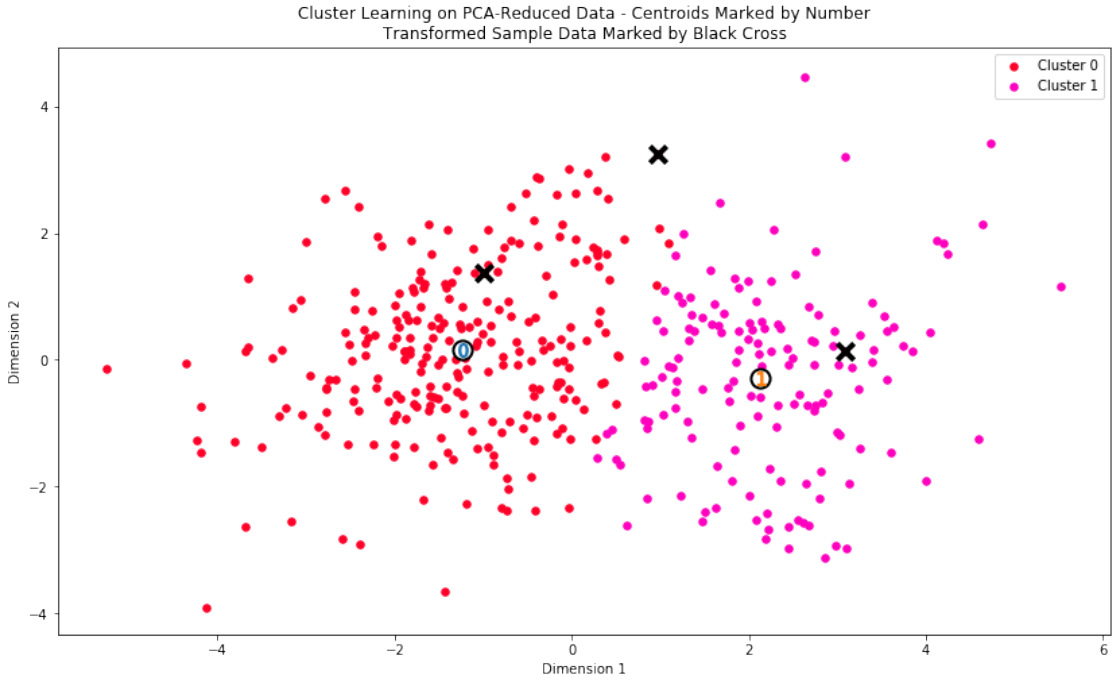
```
Highest silhouette score is with 2 cluster centers and its value is:  0.447411995571
```

- The silhouette scores of all the possible clusters numbers are shown in the above graph.
- As we can see from the graph above, the best silhouette score is for number of clusters being 2 and its value is 0.447

1.8.2 Cluster Visualization

Once we've chosen the optimal number of clusters for our clustering algorithm using the scoring metric above, we can now visualize the results by executing the code block below.

```
In [22]: # Display the results of the clustering from implementation
         vs.cluster_results(reduced_data, preds, centers, pca_samples)
```



1.8.3 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 [23]: # 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_new = pd.DataFrame(np.round(true_centers), columns = data.keys())
        true_centers.index = segments
        display(true_centers)
        true_good_data = pd.DataFrame(np.exp(good_data))
        new_true_good_data = true_good_data.append(true_centers_new,ignore_index=True)

        #Printing the heat map of the percentile values of the averages from Segment 0 and Segm
        indices_new = [398,399]
        #Percentile values of the the sampled data
        percentile_values = 100. *new_true_good_data.rank(axis=0, pct=True).iloc[indices_new].r
        #heatmap of percentiled value
        sns.heatmap(data=percentile_values,annot=True,fmt='.1f')
        plt.yticks([0.5,1.5,2.5],['Segment 0','Segment 1'],rotation='horizontal')
        plt.title('Percentile scores of customers from Segment 0 and 1')
        plt.show()

        display(true_good_data.describe())
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
Segment 0	9468.0	2067.0	2624.0	2196.0	343.0	799.0
Segment 1	5174.0	7776.0	11581.0	1068.0	4536.0	1101.0

	Fresh	Milk	Grocery	Frozen	\
count	398.000000	398.000000	398.000000	398.000000	
mean	12430.630653	5486.314070	7504.907035	3028.809045	
std	12552.698266	6410.878177	9263.803670	3712.563636	
min	255.000000	201.000000	223.000000	91.000000	
25%	4043.500000	1597.250000	2125.000000	830.000000	
50%	9108.000000	3611.500000	4573.000000	1729.500000	
75%	16969.000000	6802.500000	9762.250000	3745.000000	
max	112151.000000	54259.000000	92780.000000	35009.000000	

	Detergents_Paper	Delicatessen
count	398.000000	398.00000
mean	2725.376884	1454.71608
std	4644.023066	1746.45365
min	5.000000	46.00000
25%	263.250000	448.25000
50%	788.000000	997.50000
75%	3660.500000	1830.00000
max	40827.000000	16523.00000

1.8.4 Question

- Considering the total purchase cost of each product category for the representative data points above, and referencing the statistical description of the dataset at the beginning of this project(specifically looking at the mean values for the various feature points). We will see what set of establishments could each of the customer segments represent?

Note: A customer who is assigned to `'Cluster X'` should best identify with the establishments represented by the feature set of `'Segment X'`. Think about what each segment represents in terms their values for the feature points chosen. Reference these values with the mean values to get some perspective into what kind of establishment they represent.

Answer:

- The customers from `Segment 0` have maximum spending in(above 75th percentile) Detergents_paper, Grocery and Milk. It also has above 50th percentile spending in Delicatessen products. This segment of customers likely represents a big cafe where we get everything, ranging from milk products like coffee, fast food items and also most probably desserts.

- As we can see from the above data the customers in `Segment 1` have above average spending in Fresh and Frozen products only. In all the other categories, these customers spend below average. It is to be noted that these customers spend a bit higher in Delicatessen products which is almost 42.5th percentile. Thus we can say that such customers are small shop owners whose main business is to serve fruit-vegetables and items made from frozen things like meat etc.

1.8.5 Predicting our sample data points

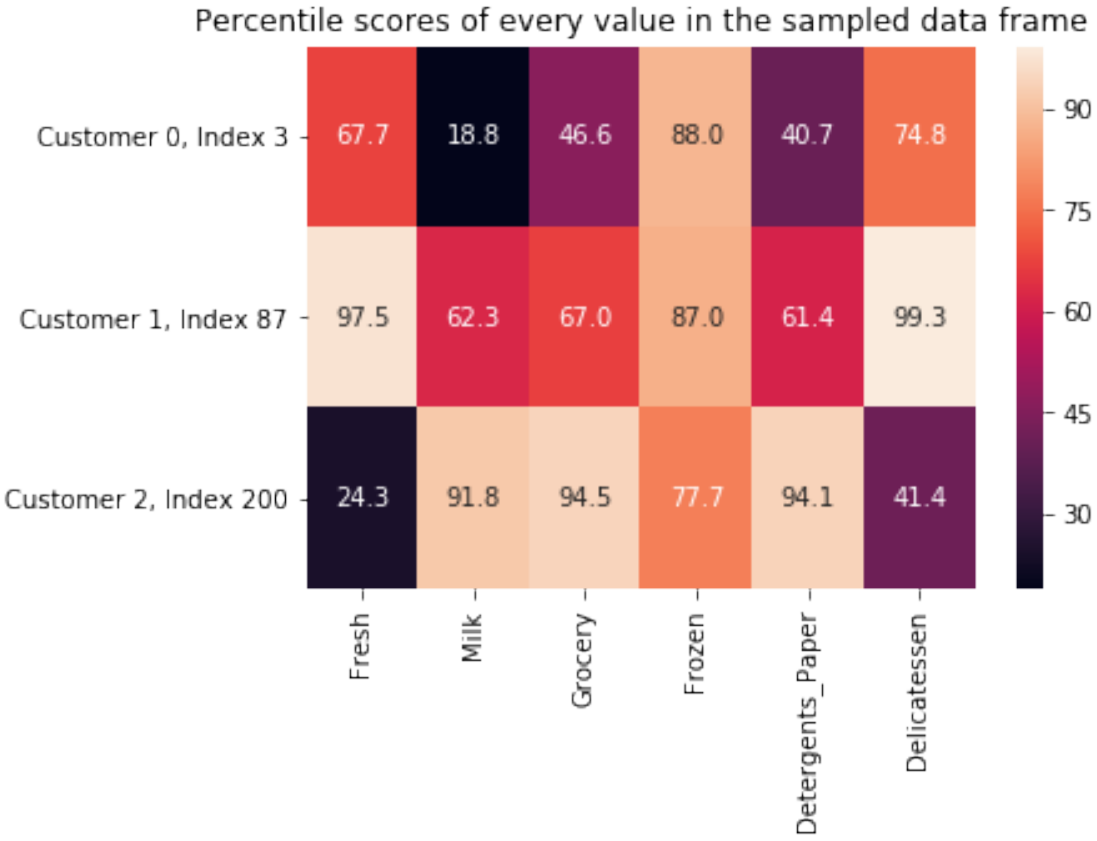
- For each sample point, which customer segment from* **previous question** *best represents it?
- Are the predictions for each sample point consistent with this?*

In [24]:
```
# Display the predictions
for i, pred in enumerate(sample_preds):
    print("Sample point", i, "predicted to be in Cluster", pred)

#Percentile values of the the sampled data
percentile_values = 100. *data.rank(axis=0, pct=True).iloc[indices].round(decimals=3)
#heatmap of percentiled value
sns.heatmap(data=percentile_values,annot=True,fmt='.1f')
plt.yticks([0.5,1.5,2.5],['Customer 0, Index '+str(indices[0]),'Customer 1, Index '+str
plt.title('Percentile scores of every value in the sampled data frame')

print("Chosen samples of wholesale customers dataset:")
display(samples)
```

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

```
Sample point 2 predicted to be in Cluster 1
Chosen samples of wholesale customers dataset:
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicatessen
0	13265	1196	4221	6404	507	1788
1	43265	5025	8117	6312	1579	14351
2	3067	13240	23127	3941	9959	731



Answer:

Customer(index)	Segment
Customer 0(3)	1
Customer 1(87)	1
Customer 2(200)	0

- The segments of the above samples are consistent with out predictions:
  - `Customer 0` and `Customer 1` are clustered in `Cluster 1`. This is consistent with our perdicted segment because as you can see from the above heatmap, Customer 0 and

1 have higher spending in Fresh, Frozen and Delicatessen. All other categories have lower spending as compared to the three categories mentioned before. This is in sync with our predicted `Segment 0` customer where an average customer from this segment tends to follow this trend in spending.
- Customer 2 is clustered as a `Cluster 0` customer. This is because `Customer 2` tends to spend the maximum in Milk, Grocery and Detergents_paper which is again in sync with trend of spending of what an average customer of `Segment 1` would do.

1.9 Conclusion

In this final section, we will investigate ways that we can make use of the clustered data. First, we will consider how the different groups of customers, the ***customer segments***, may be affected differently by a specific delivery scheme. Next, we 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, we will compare the ***customer segments*** to a hidden variable present in the data, to see whether the clustering identified certain relationships.

1.9.1 Question

Companies will often run A/B tests 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?*

Answer:

- The wholesale distributor is considering changing its delivery service from currently 5 days a week(i.e) weekdays to 3 days a week. If we think about it this service is likely going to affect those customers who need dairy(i.e Milk) products. This is because these customers are the ones who will want fresh products daily as these are the products that tend to spoil earlier then thier counterparts. Thus changing the 5 day delivery to 3 day delivery for these customers will likely affect these customers negatively and hence finally result in the wholesale distributor losing its customers.
- On the basis of the discussion above our best guess is that the Customers from `Segment 0` who tend to spend more on Milk products will be unhappy with this new service while the customers from `Segment 1` who spend less on Milk products and more on Frozen products will be indifferent to this new service.
- However to get statistically accurate readings we can apply the A/B split testing in which the distributor will sample randomly from both the segments, assign customers from one of the segment to the new delivery schedule while keeping the customers from another segment same and after getting the feedback from these customers, the distributor can conduct a hypothesis testing to check if its assumption was right and carry the same process again but this time changing the assignment of delivery schedule.

1.9.2 Question

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?

Answer:

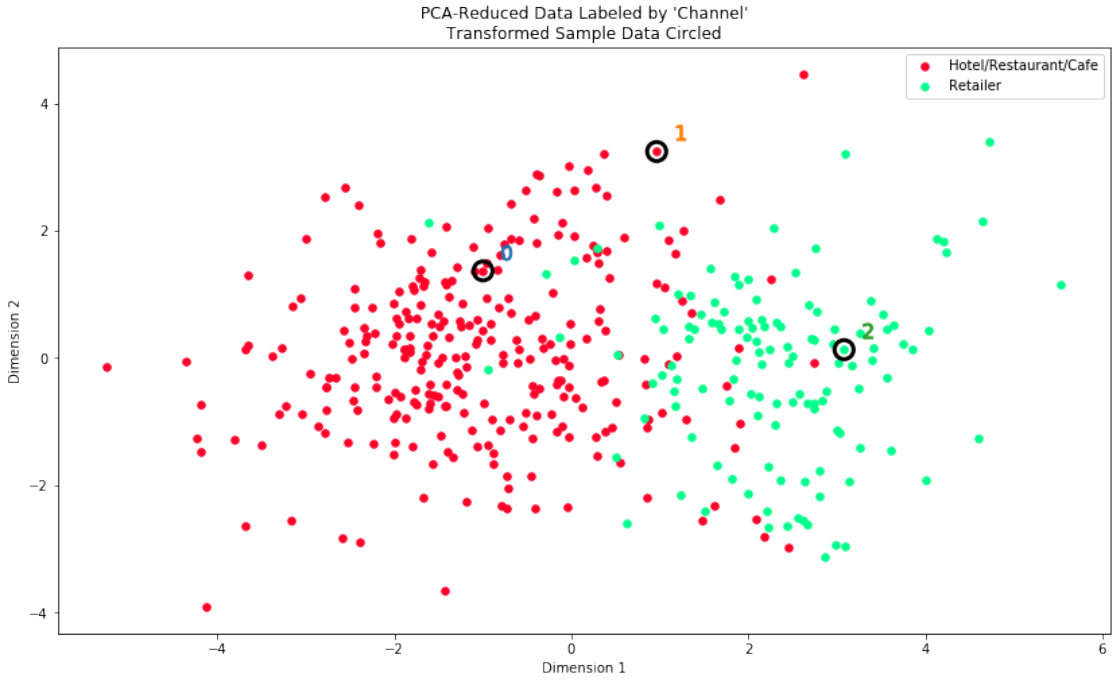
- Since now that we have divided the customers in two segments and found out which customers are likely going to be affected by the delivery schedule, we can label our original customers as customers from `Segment 0` and `Segment 1`.
- In this case the traget variable for our original customers will become these Segments.
- Thus, as we have converted our unlabelled data into a labelled data by performing unsupervised learning, we can now train a supervised model on this new labeled data.
- It is to be noted that now since we have labelled data we can train a supervised model on the full data without feature reduction, or we can train a model on the reduced data that we obtained after PCA.
- If we try to train a supervised learning model on the reduced data, we can simply go with a perceptron classifier, as there is a clear separating boundary between the two clusters formed from the GMM clustering technique.
- After training this model we can simply predict the Segments of the new customers and assign an appropriate delivery schedule to these new customers. It is to be noted that the delivery schedule comes from the A/B testing.

1.9.3 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, we will find the sample points are circled in the plot, which will identify their labeling.

```
In [26]: # Display the clustering results based on 'Channel' data

         vs.channel_results(reduced_data, all_outliers, pca_samples)
```



1.9.4 Question

- How well does the clustering algorithm and number of clusters we'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 we consider these classifications as consistent with our previous definition of the customer segments?

Answer:

- As we can see the clustering algorithm and the number of clusters that we chose are almost indentical to this true underlying distribution. Thus our model selection is perfect.
- I think there are two answers to this. There are customers that can be classified as belonging to true `Retail` category whereas the same is not true for `Hotels/Restaurants/Cafes`. This is visible from the graph above, which shows us that there are a lot of data points belonging to `Hotels/Restaurants/Cafes` category which are present in the side where the `Retail` cluster is, however not a lot of `Retail` category data points are present in the cluster of `Hotels/Restaurants/Cafes`. Now since we have a GMM clustering this means that the probability of `Hotels/Restaurants/Cafes` selling retail products is higher as compared to the probability of a `Retail` customer opening a hotel/resturant/cafe. This is visible in our day to day life too, there are cafes smaller or bigger that tend to sell products to their customers however we do not see a lot of vendors selling coffee or similar items.
- I would consider these classifications as consistent to our previous definition of customer segments. This is because the predictions of `Customer 0` and `Customer 1` is `Segment 0` which

as we specified earlier may belong to category of cafe and that of `Customer 2` is `Segment 1` which we specified earlier to be some vendor. These assumptions of ours are consistent with this true underlying distribution which shows us that `Customer 0` and `Customer 1` belong to `Hotel/Restaurant/Cafe` category and `Customer 2` belongs tp `Retail` category.