Meets Specifications

Perfect submission! 

Exceptional coding work, and analysis demonstrates a pretty fine understanding of clustering in general 

Note that I have been a bit lenient at a few places, so please do go through the remarks and the reading material provided to further improve your understanding.

Good luck for the next project! 

**Data Exploration**

**Three separate samples of the data are chosen and their establishment representations are proposed based on the statistical description of the dataset.**

Awesome work plotting the heatmap vs median for the sample points. For a more fine-grained comparison, you can use the following code to plot the percentile heatmap for sample points:

import seaborn as sns

percentiles\_data = 100\*data.rank(pct=True)

percentiles\_samples = percentiles\_data.iloc[indices]

sns.heatmap(percentiles\_samples, annot=True)

**A prediction score for the removed feature is accurately reported. Justification is made for whether the removed feature is relevant.**

Your interpretation of the relevance of a feature based on its prediction score is absolutely correct!

The low/negative prediction score for a feature means that the values of that feature cannot be predicted well by the other features in the dataset and therefore, the feature is not redundant and may contain useful information not contained in other features.

On the other hand, a feature that can be predicted from other features would not really give us much additional information and thus, would be a fit candidate for removal, if we ever need it to make the dataset more manageable.

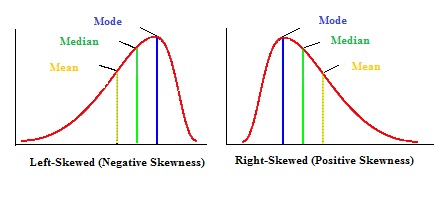
**Suggestion:**

Your choice of random states can have a huge influence on the R^2-score obtained, which could, in turn, have an influence on your interpretation of the relevance of a feature. To mitigate this, you can average the prediction scores over many iterations, say 100, without setting any of the random states.

In particular, such an averaging would lead to a much lower score for Milk (~0.1), which would, in turn, have an influence on your interpretation of its relevance.

**Student identifies features that are correlated and compares these features to the predicted feature. Student further discusses the data distribution for those features.**

**Remarks:**

* The most significant correlation is definitely between Grocery and Detergents\_Paper. Milk is also correlated with both these features, but the correlation is relatively mild.
* This, indeed, complements your interpretation from the previous question. We do get additional information if we keep both of Grocery and Detergents\_Paper in the dataset, but we can drop one just in case we severely need to reduce the dimensionality of our feature space. Later, we will see a better way of reducing the dimensionality of our dataset - PCA.
* Well done remarking that the features' distribution is not normal, but skewed! To be technically precise, the distribution is skewed to the right, as illustrated in the following sketch:
* [](https://udacity-reviews-uploads.s3.amazonaws.com/_attachments/29780/1494633566/pearson-mode-skewness.jpg)
* Clustering algorithms discussed in this project work under the assumption that the data features are (roughly) normally distributed. Significant deviation from zero skewness indicates that we must apply some kind of normalisation to make the features normally distributed.

**Data Preprocessing**

**Feature scaling for both the data and the sample data has been properly implemented in code.**

**Student identifies extreme outliers and discusses whether the outliers should be removed. Justification is made for any data points removed.**

**Remarks:**

* Good work, correctly identifying the Tukey outliers for more than one features!
* The most important factor to consider while removing outliers is the impact they might have on clustering algorithms because of the distance averaging involved. In our context, cluster\_centers turn out to be relatively insensitive to the choice of outliers, unless the outliers end up forming a different cluster by themselves, which could indeed happen if they are not removed at all.
* It is also important to achieve a compromise between removing outliers to get better clustering results, and not removing too much useful information. Removing all the Tukey outliers, even those for only one feature, effectively removes 10% of samples from our dataset and is generally not recommended without a strong justification. Therefore, one might choose to remove only the "extreme" outliers, where "extreme" is reasonably defined, for example, the outliers for more than one features, and/or outliers obtained by increasing the step size in Tukey's method.

**Code tip:**

You can also use the concept of [counter](https://docs.python.org/2/library/collections.html#collections.Counter) to identify the multiple-feature Tukey outliers:

from collections import Counter

# after creating a list of all the Tukey outliers

outliers\_mult = [item for item, count in Counter(outliers).iteritems() if count > 1]

**Feature Transformation**

**The total variance explained for two and four dimensions of the data from PCA is accurately reported. The first four dimensions are interpreted as a representation of customer spending with justification.**

**Remarks:**

* Good work estimating the cumulative explained variance for the first two and four dimensions. You could use the following code to more precisely compute these values:print pca\_results['Explained Variance'].cumsum()
* Nice work elaborating on each dimension, but it would be better if after remarking the relative weights given to the different features in each dimension, you could surmise what kind of customers might be well-separated along this dimension. For example, a dimension giving relatively high (positive or negative) weights to Fresh, Milk, Frozen and Delicatessen would likely separate out the restaurants from the other types of customers.
* Another important point to note is that the sign of a PCA dimension itself is not important, only the relative signs of features forming the PCA dimension are important. In fact, if you run the PCA code again, you might get the PCA dimensions with the signs inversed. For an intuition about this, think about a vector and its negative in 3-D space - both are essentially representing the same direction in space. You might find this [exchange](http://stats.stackexchange.com/questions/88880/does-the-sign-of-scores-or-of-loadings-in-pca-or-fa-have-a-meaning-may-i-revers) informative in this context.

The following links might be of interest in the context of this question:

<https://onlinecourses.science.psu.edu/stat505/node/54>

<http://setosa.io/ev/principal-component-analysis/>

**PCA has been properly implemented and applied to both the scaled data and scaled sample data for the two-dimensional case in code.**

**Clustering**

**The Gaussian Mixture Model and K-Means algorithms have been compared in detail. Student’s choice of algorithm is justified based on the characteristics of the algorithm and data.**

Good job comparing GMM and KMeans!

From a practical standpoint, the main criteria for deciding between these two algorithms are the speed v/s second order information (confidence levels) desired and the underlying structure of our data.

**Regarding your choice of algorithm:**

Both the algorithms will do fine here, although considering the fact that there are no visually separable clusters in the biplot, one might, indeed, prefer the soft-clustering approach of GMM, particularly since the dataset is quite small and scalability is not an issue.

For large datasets, an alternative strategy could be to go with the faster KMeans for preliminary analysis, and if you later think that the results could be significantly improved, use GMM in the next step while using the cluster assignments and centres obtained from KMeans as the initialisation for GMM. In fact, many implementations of GMM automatically perform this preliminary step for initialisation.

**Several silhouette scores are accurately reported, and the optimal number of clusters is chosen based on the best reported score. The cluster visualization provided produces the optimal number of clusters based on the clustering algorithm chosen.**

Awesome coding work! Indeed, number of clusters = 2 gives the best silhouette score among the many considered!

**Important remark regarding the choice of outliers:**

This is one place where your choice of outliers plays a huge role. For example, repeat the analysis without removing any outlier. What is the optimal number of clusters that you get?

**Miscellaneous remarks:**

* If you want to get more support for your results obtained using silhouette analysis, one way is to check how **balanced** are the clusters obtained from different values of number of clusters, using the code given at this [link](http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html#sphx-glr-auto-examples-cluster-plot-kmeans-silhouette-analysis-py).
* Remark that in certain cases, you can even choose a value for number of clusters which gives a sub-optimal score. For example, in the link provided, 2 is not considered optimal, despite having a better Silhouette score, because it doesn't result in **balanced** clusters, while 4 does.
* From [sklearn documentation](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_samples.html), the Silhouette Coefficient is calculated using the mean intra-cluster distance and the mean nearest-cluster distance for each sample. Therefore, it makes sense to use the same distance metric here as the one used in the clustering algorithm. This is Euclidean for KMeans (default metric for Silhouette score) and Mahalanobis for general GMM.
* For GMM, [BIC](http://scikit-learn.org/stable/auto_examples/mixture/plot_gmm_selection.html) could sometimes be a better criterion for deciding on the optimal number of clusters, since it takes into account the probability information provided by GMM. I leave you to experiment with this.

**The establishments represented by each customer segment are proposed based on the statistical description of the dataset. The inverse transformation and inverse scaling has been properly implemented and applied to the cluster centers in code.**

Ideally, you should make an explicit reference to the statistical description of the dataset when proposing the establishments represented by the two clusters, as you did in Q1.

**Sample points are correctly identified by customer segment, and the predicted cluster for each sample point is discussed.**

Nice discussion, but ideally, you should explicitly compare the features of the sample points to those of the cluster centers to conclude whether the predictions from the algorithm agree with your intuition or not.

One interesting point to note from the cluster\_visualization plot is that the two clusters are essentially separated by a value on the first PCA dimension, which we saw earlier is predominantly a combination of Detergents\_Paper, Grocery and Milk. The rest of the features, which figure prominently only in the second PCA dimension, don't really matter!

**Conclusion**

**Student correctly identifies how an A/B test can be performed on customers after a change in the wholesale distributor’s service.**

Excellent! You have correctly identified the key point here which is to test each segment independently, although to be technically correct, you must also form a control group for each segment, corresponding to the test group. For an A/B test to be effective, the experiment group (A) has to be highly similar to the control group (B), before the treatment is applied to the experiment group. If they are dissimilar to each other, then the result of the A/B test might be due to some variable other than the variable being tested.

I give below a few links which might help remove misconceptions on this topic, if any:

<https://www.quora.com/When-should-A-B-testing-not-be-trusted-to-make-decisions/answer/Edwin-Chen-1>

<http://multithreaded.stitchfix.com/blog/2015/05/26/significant-sample/>

<http://techblog.netflix.com/2016/04/its-all-about-testing-netflix.html>

<https://vwo.com/ab-testing/>

<http://stats.stackexchange.com/questions/192752/clustering-and-a-b-testing>

**Student discusses with justification how the clustering data can be used in a supervised learner for new predictions.**

**Comparison is made between customer segments and customer ‘Channel’ data. Discussion of customer segments being identified by ‘Channel’ data is provided, including whether this representation is consistent with previous results.**

Remark that the channel\_visualization validates, to some extent, the choice of using GMM, as the clusters do have a fair amount of overlap in reality. Although a perfect classification is not possible to achieve, soft clustering gives us confidence levels in our predictions, which would understandably be low at the boundary between two clusters.

**Code tip:**

You can calculate the accuracy score for clustering using the following code:

channel\_labels = pd.read\_csv("customers.csv")["Channel"]

channel\_labels = channel\_labels.drop(channel\_labels.index[outliers]).reset\_index(drop = True) - 1

# channel\_labels = abs(channel\_labels -1)

from sklearn.metrics import accuracy\_score

accuracy = accuracy\_score(channel\_labels,preds)

Note that I've subtracted 1 from channel\_labels, because the given channel\_labels are 1 and 2, while our cluster-labels are 0 and 1.

Also, note that the assignment of labels - 0 and 1 - in the clustering algorithm is completely arbitrary. Therefore, you might have to keep or remove channel\_labels = abs(channel\_labels -1) in the above code, to ensure that the cluster and channel labels are "compatible".