UDACITY

‹ Return to "Machine Learning Engineer Nanodegree" in the classroom

DISCUSS ON STUDENT HUB

Creating Customer Segments

REVIEW

CODE REVIEW

HISTORY

Requires Changes

1 SPECIFICATION REQUIRES CHANGES

Overall fantastic work! There's just one section that needs a little more depth of analysis, but overall you went above and beyond. It's clear you put a lot of effort into the project, and the one revision required shouldn't take you too long.

Keep at it! You're nearly done!

Data Exploration

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

Good job here!
Your intuitions are backed up with statistical descriptions of the data 👍

TIP

In general, I find it really helpful to visualize sample points when I'm trying to figure out what they represent. You can do this quite simply with the following code 😄

```
import matplotlib.pyplot as plt
import seaborn as sns

samples_for_plot = samples.copy()
samples_for_plot.loc[3] = data.median()

labels = ['Sample 1','Sample 2','Sample 3','Median']
samples_for_plot.plot(kind='bar')
plt.xticks(range(4),labels)
plt.show()
```

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

Great!
You nailed the key point here - if we can reliably reconstruct a feature from other features, it probably doesn't contain a whole lot of unique information. Conversely, if we can't, then the feature must contain some unique information and is necessary to retain. 👍

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

Good job addressing the distributions! Often, when we have right-skewed data it approximates a log-normal distribution. It's for this reason that we normalize it in the next section with a log-transformation. However, there are cases where the log-transform doesn't get us as close to normal as we'd like. Another option in that case is the BoxCox transformation.
We can apply it here like so

```
from scipy.stats import boxcox
boxcox_data = data.apply(lambda x: boxcox(x)[0])
pd.scatter_matrix(boxcox_data, alpha = 0.3, figsize = (14,10), diagonal = 'kd
e');
```

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.

Great job!
You identified the points which are outliers in more than 1 feature 👍

TIPS ON HANDLING OUTLIERS
Deciding what to do with outliers is never a simple choice. Simply removing them can often delete useful structure in our data, but leaving them in can cause problems in many machine learning algorithms (including clustering and PCA!). There's no one-size-fits-all solution, so I definitely recommend taking a look at the links below. They do a great job of covering some common cases and how to handle them 😄
http://www.theanalysisfactor.com/outliers-to-drop-or-not-to-drop/
http://unilytics.com/how-to-handle-outliers-in-your-data/

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.

Great start here, but you'll need to go into a bit more depth about what sort of customers could be separated by each component. Let's take the first one as an example.
Note how there is strong-to-moderate negative weights on detergent, grocery and milk and small positive weights on fresh and frozen. This means that as a customers score in this section goes up, they will buy less detergent, grocery and milk while buying a little more fresh and frozen. Now consider:

- What sort of customer would have a high score in this component?
- What sort of customer would have a high *negative* score in this component?
- Based on your answers, what sort of spending pattern does this component represent?

Make sure this is answered for each of the first four components.

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.

Great work!
In general, K-Means offers better performance if we care about

- Speed
- Scaleability
- Simplicity

Whereas GMM provides more

- Flexibility
- Robustness

The fact that K-Means assumes that all clusters is globular is a pretty enormous assumption, and is always something we have to take into consideration. GMM is far less rigid in this - it allows these spheres to be stretched and compressed.

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.

> NOTE: None of my scores are above .4? Is this okay? I was expecting higher silhouette scores than .39.

Yep! Silhouette scores are not as straightforward to interpret as something like R^2 - they can be very dataset-dependent, and tend to give you better understanding of the relative performance of clustering on that dataset than the absolute performance.

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.

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

Conclusion

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

Awesome job! The key here is that we test each segment individually. That way we avoid generalizing our results to customers where they wouldn't apply.

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

Awesome job! This is indeed how we'd translate the problem to a supervised setting. Why might we want to do this instead of just using our clusterer? A couple possibilities:

1. Maybe we want to take advantage of the explanatory capability of some supervised models. Decision trees, for instance, can give us information about feature importance. Although it's not clear in this exact setting that this is necessary, if our data were more complex (say, if we used all the components instead of just the top it might be useful to understand how those clusters are created.
2. Scaling. Clusterers are good for small to medium-size datasets (some clustering algorithms can't even handle medium-sized ones). K-means is probably the fastest and most efficient one, but if we had a ton of clusters we'd still need to be comparing new data points to all of those means. Using logistic regression, on the other hand, we could get much faster answers and scale up to a much larger dataset if we wanted too.

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.

Nice! You might want to try GMM here - it tends to perform better in general, and definitely on this dataset.

RESUBMIT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH