UDACITY

‹ Return to Classroom

Identify Customer Segments

审阅

代码审阅

HISTORY

Meets Specifications

Dear student

Great job on your project! You've clearly understood the material from the tutorials and you've done a fantastic job applying it to a real-world dataset. Congratulations on passing quickly and good luck with the next section of the course.

Cheers!

Suggested reading:

- In case you haven't seen it before, the pandas library has a great lightweight pipeline that can be very convenient for this type of data processing: https://www.kdnuggets.com/2019/12/build-pipelines-pandas-pdpipe.html

Preprocessing

Columns with a large amount of missing values have been removed from the analysis. Patterns in missing values have been identified between other columns.

> From the plot it is clear to find 6 columns have missing rate larger than 30%. These columns were removed. For the rest columns, most of them have missing rate lower than 20%, it is impratical to remove all these columns. Hence they are saved.

Nice work with your visualizations! You've done a great job identifying the columns that are outliers in terms of high missing values. The columns that you've removed are definitely outliers in terms of missing values.

All missing values have been re-encoded in a consistent way as NaNs.

Great job replacing the missing value codes here! This is a very Pythonic way to do this.

Mixed-type features have been explored, resulting in re-engineered features.

```
#Defines 2 map functions.
def map_digit1(x):
    try:
        if pd.isnull(x):
            return np.nan
        else:
            return int(x)//10
    except ValueError:
        return np.nan

def map_digit2(x):
    try:
        if pd.isnull(x):
            return np.nan
        else:
            return int(x)%10
    except ValueError:
        return np.nan
```

Nice work generating the new features. Very creative work in flipping around the values of the `CAMEO_INTL_2015` feature for easier interpretability.

Categorical features have been explored and handled based on if they are binary or multi-level.

> In the dataset there are 18 category variables. I checked the unique values of them and decided to convert all of them to dummy variables. Notice for some category variables there are NANs.

Great job! I think you've done a good job balancing the preprocessing requirements with the issue of dimensionality. It's certainly important to avoid diluting the variability captured by components in the PCA step.

The data has been split into two parts based on how much data is missing from each row. The subsets have been compared to see if they are qualitatively different from one another.

Nice job! I think you've done a good job choosing a reasonable threshold. There are two clear clusters in the rows with high missing value counts: one at < 9 missing values per row and another at >30 missing values per row. Either of these (or anything in between) is a good cutoff point for this analysis.

A function applying pre-processing operations has been created, so that the same steps can be applied to the general and customer demographics alike.

Very nice! Great employment of all your work into the clean_data() function. I'd also recommend testing the function out in order to verify that it works on the general demographics data.

Dataset includes all original features with appropriate data types and re-engineered features. Features that are not formatted for further analysis have been excluded.

```
df_new['PRAEGENDE_JUGENDJAHRE_DECADE']=df_new['PRAEGENDE_JUGENDJAHRE'].apply(lambda
 x: map_decade(x))
df_new['PRAEGENDE_JUGENDJAHRE_MOVE']=df_new['PRAEGENDE_JUGENDJAHRE'].apply(lambda x:
map_move(x))
df_new.drop(columns='PRAEGENDE_JUGENDJAHRE',axis=1,inplace=True)
df_new['CAMEO_INTL_2015_1']=df_new['CAMEO_INTL_2015'].apply(lambda x: map_digit1(x))
df_new['CAMEO_INTL_2015_2']=df_new['CAMEO_INTL_2015'].apply(lambda x: map_digit2(x))
df_new.drop(columns='CAMEO_INTL_2015',axis=1,inplace=True)
df_new.fillna(0,inplace=True)
```

Looks good!

Feature Transformation

Feature scaling has been properly applied to the demographics data. Imputation has been performed to

Feature scaling has been properly applied to the demographics data. Imputation has been performed to remove remaining missing values.

> I used fillna to convert all the NANs to 0. I hope it will not influence the clustering. StandardScaler was imported from sklearn to convert the data to same scale.

Great job describing your preprocessing methodology! I'd also suggest trying to impute the missing values using the "mean" method in columns where the values are continuous. The median/most frequent method will definitely work best for the categorical features. Taking a more nuanced approach to handling missing values might show some interesting patterns when you analyze the clusters in your downstream analysis.

Weights on at least three principal components are used to make inferences on correlations between original features of the data. General meanings are ascribed to principal components where applicable.

Excellent analysis of each principal component!

Principal component analysis has been applied to the data to create transformed features. A variability analysis has been performed to justify a decision on the number of features to retain.

> Firstly I used all the features to run PCA. Then after the plot I found after 150 components the sum of variance almost stops to increase. So I decided to use 158(80% of the features in the data set) components.

Great choice! I think ~125 PC's is right in the sweet spot. You could definitely have dropped or kept a few more components, but retaining most of the variance in the dataset while reducing the dimensionality of the dataset by 1/4 is pretty optimal.

Clustering

Multiple cluster counts have been tested on the general demographics data, and the average point-centroid distances have been reported. A decision on the number of clusters to use is made and justified.

> In this section, I tested different number of clusters and plot the distance with the numbers. Unfortunately, I cannot find a obvious turning point. So I choose 20 as the cluster number since at this point the distance is relatively small and increasing more clusters does not give a huge decrease of the distance.

Looks good! There definitely wasn't a distinct elbow in either of the curves so this is a bit of a judgement call. Anywhere from 15-20 clusters looks pretty good

Anywhere from 15-20 clusters looks pretty good.

A comparison is made between the general population and customers to identify segments of the population that are central to the sales company's base as well as those that are not.

Cleaning, feature transformation, dimensionality reduction, and clustering models are applied properly to the customer demographics data.

```
#normalization using StandardScaler
customers_trans=scaler.transform(customers_new)
#transform the customers data using pca object
X_customers=pca_new.transform(customers_trans)
#predict clustering using the kmeans object
customers_pred=model_final.predict(X_customers)
```

Nice work! By re-using the same (previously fitted) methods, you can extrapolate how your pipeline will perform when you apply it to new data.

下载项目

返回 PATH

给这次审阅打分

开始