UDACITY

‹ Return to Classroom

Identify Customer Segments

REVIEW

HISTORY

Meets Specifications

Awesome 👏👏

Congratulations on passing this project. 🏆🏆

This project was not easy but you have done it gracefully. This is all because of your hard work and continuous evaluation. But still don't relax, keep exploring and learning from the other references. These projects are just a starting phase of learning these things but there are many more complex things out there. So start exploring and keep learning. 💪💪

Useful References

- Visualizing statistical plots with Seaborn
- Univariate Distribution plot
- sklearn's RobustScaler

You can also share your project on LinkedIn and ask the audience for a necessary feedback or open the project for anyone to collaborate. This way you will find many interesting connections and engagement with others.

I wish you good luck. Looking forward to your success.
For any queries, you can ask on Knowledge Portal as well.

Stay ! Stay Safe

DON'T FORGET TO RATE MY WORK AS PROJECT REVIEWER! YOUR FEEDBACK IS VERY HELPFUL AND APPRECIATED.

Preprocessing

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

✅ Missing value codes given in feat_info's last column have been used to convert all codes to NaNs.

✅ You have taken care of the 'X' and 'XX' strings in their code

Comments:

- Good job writing a loop to handle the missing value cases. Nice attention to the special encoding of the 'X' and 'XX' cases compared to the others.
- It is necessary to have all the missing values altered in the format that can be comprehended by the panda's tool so that you can later choose to deal with them easily. For e.g. replacing them or dropping rows with missing values etc.
- It's important to take special care when the given dataset has undeclared missing values that may still hinder the data analysis, such as 'X' and 'XX' strings.

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

✅ Columns with a large number of missing values have been removed from the analysis

✅ Patterns in missing values have been identified between other columns

Suggestions:

If a significant number of values are missing from the column, it is worth removing them, but there are still some pros and cons.

Pros:

- Complete removal of data with missing values results in robust and highly accurate model
- Deleting a particular row or a column with no specific information is better since it does not have a high weightage

Cons:

- Loss of information and data
- Works poorly if the percentage of missing values is high (say 30%) compared to the whole dataset

🔗 5 Ways To Handle Missing Values In Machine Learning Datasets

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.

✅ The data has been split into two parts based on how much information is missing from each row.

Comments:

- Very good job in setting the threshold as 20 for splitting the data based on the missing values.
- It is essential to look at the distribution over specific columns for rows with more than a certain number of missing values, with the rows more petite than a certain number of missing values.

✅ The subsets have been compared to see if they are qualitatively different from one another.

Comments:

- Good visuals and comparisons of the features provided in the notebook.
- These bar charts show us a summary of all the values in a single picture.

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

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

Comments:

- Nice job balancing the preprocessing requirements without the issue of dimensionality. Remember that it is essential to avoid diluting the variability captured by components in the PCA step.
- All categorical variables need to have numerical values for further analysis since we would be fitting a Kmeans model to solve the problem, and this particular model only ingests numeric values while training.

Suggestions:

- Always ensure you do not fall into the **curse of dimensionality**.

As the number of features grows, the amount of data we need to accurately distinguish between these features (to give us a prediction) and generalize our model (learned function) grows.

- In most cases, high cardinality makes it difficult for the model to identify such patterns; hence, the model doesn't generalize well to examples outside the training set.

🔗 Dealing with features that have high cardinality

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

✅ Two mixed-type features, PRAEGENDE_JUGENDJAHRE and CAMEO_INTL_2015, engineered into two new features.

Comments:

Nice work generating the new features.

- **Decade = information about the decade the youth belongs to.**
- **Movement = information about the prevalent movement**
- **Wealth = information about the wealth of a family**
- **Life Stage = stage of the family development.**

Why are we doing this operation?

Values in `PRAEGENDE_JUGENDJAHRE` and `CAMEO_INTL_2015` won't be helpful for the algorithm (uninterpretable by the algorithm). That is why we derive these numerical features from using the information in our model effectively.

🔗 Complete Guide to Feature Engineering: Zero to Hero

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

✅ Dataset includes all original features with appropriate data types and re-engineered features

✅ Features that are not formatted for further analysis have been excluded.

Comments:

- Excellent job at retaining original features with appropriate data types
- You have engineered the features reflecting Wealth, Life Stage, Decade and Movement, which will help increase the variability in the data.
- Features like PRAEGENDE_JUGENDJAHRE and CAMEO_INTL_2015 are worth dropping in their original formulations as they are mixed variable and is uninterpretable by most of the modeling techniques.

Suggestion:

You can also explore other mixed variables like LP_LEBENSPHASE_GROB, PLZ8_BAUMAX, etc., to extract meaningful features to increase the variability further.

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

✅ A function applying pre-processing operations has been created

✅ same steps can be applied to the general and customer demographics alike

Comments:

- Great employment of all your preprocessing work into the `clean_data` function.

- This way, you will obey the 🔗 DRY principle and reduce the risk of missing any preprocessing step to be applied to the population and customer datasets.**

Suggestions:

- You might want to add a code cell after the **clean_data** function to test it out or to verify that it works on the general demographics data as expected.
- You can use python's assert method. Here is one sample code:

```
assert azdias_clean_data.shape[0] == azdias_manual_clean.shape[0], "clean_data function is not working properly, rows mismatch"
assert azdias_clean_data.shape[1] == azdias_manual_clean.shape[1], "clean_data function is not working properly, columns mismatch"
print("If this is all you see, you passed the tests")
```

Feature Transformation

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

✅ Feature scaling has been properly applied to the demographics data.

✅ Imputation has been performed to remove the remaining missing values

✅ Valid justifications are provided for handling the missing values and scaling operation, in the report.

What if you don't handle missing values?

- Certain sklearn estimators do not work if the data contains missing values since their mathematical operations demand the data to have all values in numeric format.
- For instance, if you directly try to fit the Standscaler object without handling missing values, then you will face the following error:

```
ValueError: Input contains NaN, infinity or a value too large for dtype('float64')
```

- So it becomes necessary to handle the missing values before any fitting operation.

What if you don't feature scale the data?

- Suppose a feature's variance is orders of magnitude more than the variance of other features. In that case, that feature might dominate other features in the dataset, which is not something we want to happen.
- This skews the PCA towards high magnitude features when fitted on such data.

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.

Good justification has been provided for choosing 125 components.

The principal component cutoff is justified correctly by observing the turning point or drop-off in variance.

Why are we performing this step?

It is pretty challenging to work with a dataset having hundreds of features. This can lead to an ML model suffering from 🔗 overfitting. So to avoid this type of problem, it is necessary to apply 🔗 Feature Extraction techniques. Advantages of such methods are:

- Accuracy improvements.
- Overfitting risk reduction.
- Speed up in training.
- Improved Data Visualization.
- Increase in explainability of our model.

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.

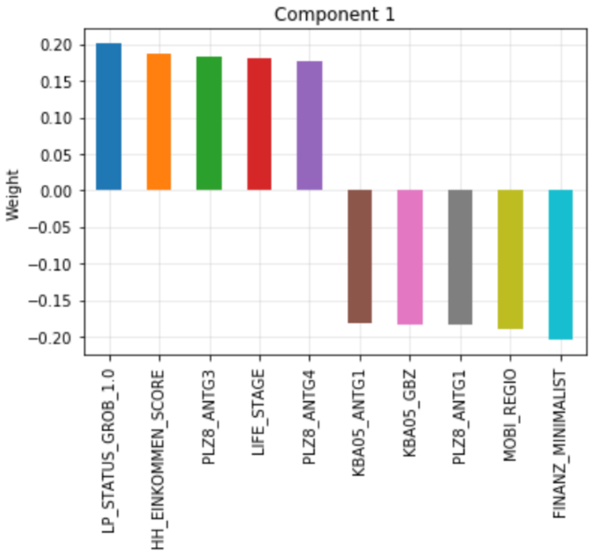
✅ You should map each weight to their corresponding feature name, then sort the features according to weight.

Comments:

- Good job! all the features have been mapped to correct weights.

Suggestions:

- Instead of analyzing the numerical weights, you can visualize the most prominent features as shown below:



- (hint: use pandas plot() method)

✅ You should investigate and interpret feature associations from the first three principal components

Comments:

- The interpretation of the printed-out component coefficients is well done!
- This part is tricky since a lot of the coefficients have ‘better’ meanings associated with smaller values.

🔗 Interpretation of the Principal Components

✅ You should write a function that you can call at any time to print the sorted list of feature weights, for the *i*-th principal component

Comments:

- You have done a great job in defining a function for mapping the features with their weights.
- This will reduce the redundancy in the code and obey the DRY principle.

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.

✅ *Multiple cluster counts have been tested on the general demographics data.*

Comments:

- Excellent work! you have wonderfully tested the scores for a good range of K values.

✅ average point-centroid distances have been reported

✅ A decision on the number of clusters to use is made and justified

Comments:

- 15 clusters seem like a valid number of clusters to keep.
- Excellent work creating the scree plot to identify the popular elbow curve that helps decide the ideal number of centroids to be used for clustering.

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.

✅ Compute the proportion of data points in each cluster and visualize, for the general population and the customer data

✅ Identified overrepresented and underrepresented clusters

✅ Inferred the kind of people being represented by each cluster type.

Comments:

- Good job here; you have correctly:
    - compared the proportion of data.
    - identified the overrepresented and underrepresented clusters.
    - interpret the extracted data to infer the kind of people.
- The extracted customer segments will help the mail-order sales company target the specific customer segments while initiating direct marketing campaigns.
- This will help the company to achieve the highest expected rate of returns.

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

✅ Feature wrangling, selection and engineering using clean_data() function.

Comments:

- Awesome! the clean_data function is applied successfully without any issue.
- You might receive a different set of features after cleaning the customer data. Have a look at 🔗 this post to know the workaround.

✅ Data transformation using already fitted Imputer object

✅ Data transformation using already fitted Scaler object.

✅ Feature extraction using already fitted PCA object

✅ Data segmentation using already fitted KMeans model.

Comments:

- Good job using the previously-fitted sklearn objects to process the customer demographics data.
- It is crucial since we need to transform the customer demographic data in the same way (same mean and same standard deviation) as we did with the general demographic data.

DOWNLOAD PROJECT

RETURN TO PATH