More on ML with Python

Exercise 1 : Visualizing Word2Vec Word Embeddings using t-SNE

Word embeddding is a type of word representation, by means of a high-dimensional numerical vector (around hundred of components). Popular word embeddings as `Word2Vec`, `BERT` are based on neural networks.

To understand how we can visualize word clusters, we shall combine `Word2Vec` representation and `t-SNE`.
To do so we shall use `gensim` Python library and begin to import the embeddings

```
import numpy
import gensim
model_gn = gensim.models.KeyedVectors.
load_word2vec_format(’/home/marianne/GoogleNews-vectors-negative300.bin.gz’,
binary=True)
```

The embeddings can be downloaded with the link

`https://drive.google.com/file/d/1Jqn0svMINDhi2zSPkAT6T21H1XQFf-WB/view?usp=sharing`

1. We now create synthetic data, naturally associated to clusters

   ```
   keys = [’Paris’, ’Python’, ’Sunday’, ’Tolstoy’, ’Twitter’, ’bachelor’,
   ’delivery’, ’election’, ’expensive’, ’experience’, ’financial’, ’food’,
   ’iOS’, ’peace’, ’release’, ’war’]

   embedding_clusters = []
   word_clusters = []
   for word in keys:
         embeddings = []
         words = []
         for similar_word, _ in model_gn.most_similar(word, topn=30):
              words.append(similar_word)
              embeddings.append(model_gn[similar_word])
         embedding_clusters.append(embeddings)
         word_clusters.append(words)

   embedding_clusters = np.array(embedding_clusters)
   n, m, k = embedding_clusters.shape
   ```

2. Perform PCA on this synthetic dataset and visualize the different clusters related to each `key`

3. Perform t-SNE and and visualize the different clusters related to each `key` in the t-SNE space

4. Compare both

More on ML with Python

For the two following exercises, we consider the results of a survey given to visitors of hostels listed on Booking.com and TripAdvisor.com. Our features here are the average ratings for different categories

- "f1": "Staff"
- "f2": "Hostel booking"
- "f3": "Check-in and check-out"
- "f4": "Room condition"
- "f5": "Shared kitchen condition"
- "f6": "Shared space condition"
- "f7": "Extra services"
- "f8": "General conditions conveniences"
- "f9": "Value for money"
- "f10": "Customer Co-creation"

Our target variable is the hostel's overall `rating` on the website. The dataset is `hostel_factors.csv` and can be downloaded on the github repository

Exercise 2 : Feature importance and Random Forests

1. Download the data `hostel_factors.csv`, store it in a dataframe and add the names of the columns to the dataframe
2. Fit a Random Forest Regressor. Evaluate the model. Comment
3. Calculate the Random Forest Built-in Feature Importance of each feature using the function `feature_importances_`
4. Calculate the Permutation Based Feature Importance using the function `permutation_importance` of the library `sklearn`
5. Use the function `shap.TreeExplainer` of the library `shap` to calculate the shapley values of the model. Compare

Exercise 3 : Quantile regression with ensemble methods

1. Fit a Quantile Regression Forest.
2. Store the quantiles corresponding to the 97.5th and 2.5th percentile
3. Plot the confidence intervals for the regression