Guided Projects Artificial Intelligence & Machine Learning

Guided Projects: Unsupervised Learning

PLSA: Text Document Clustering

Name	Lakshmi Thirunavukkarasu
Course	AI and ML (Batch 5)
Problem Statement	Perform topic modelling using the 20 Newsgroup dataset.

Software requirements prerequisites

Anaconda

Python 3.8

Python Packages

- NLTK
- NumPy
- Pandas
- Scikit
- Matplotlib

Steps

1. **Text Preprocessing**

```
 # Convert into lower case character
df_train[0] = df_train[0].map(lambda x: x.lower())

# remove from, subject, nntp and organization details from the text
df_train[0] = df_train[0].apply(lambda x: remove_sentence(x))

#replace web addresses with null value
df_train[0] = df_train[0].map(lambda x: re.sub(r'[\w\.-]+@[\w\.-]+','',x))

#remove all the special characters
df_train[0] = df_train[0].map(lambda x: re.sub('[,\.!?_=()"|><*]', '', x))

#Remove numeric values
df_train[0] = df_train[0].map(lambda x: re.sub(r'[0-9]+', '', x))

#Apply lemmization
df_train[0] = df_train[0].apply(lambda x: lemmatizer(x))

#Remove stopwords
df_train[0] = df_train[0].apply(lambda x: stop_words(x))
```

2. Convert the text into TDIDF vector

```
# Convert to TF-IDF format
#vectorizer_train= TfidfVectorizer(max_df=0.50, min_df=0.07, stop_words='english', use_idf=True)
vectorizer_train = TfidfVectorizer(max_df=0.50, min_df=0.01, stop_words=None,use_idf=True,
                                   max_features = 1000,lowercase=False, ngram_range=(1,3))
X_train = vectorizer_train.fit_transform(df_train[0])
tfidf_feature_names = vectorizer_train.get_feature_names_out()
print(X_train.shape)
print(tfidf_feature_names)
```

3. Create LDA model

```
from sklearn.decomposition import LatentDirichletAllocation as LDA
lda = LDA(n_components=4, random_state=0)
clf = lda.fit(X_train.toarray(), labels_train)
clf_l = clf.transform(X_train.toarray())

```

4. Create PLSA model

```
#Number of topcis
from sklearn.decomposition import NMF
no_of_topics = 4
model = NMF(n_components = no_of_topics, init='nndsvd',regularization='components', random_state = 111 )
W1 = model.fit_transform(X_train)
H1 = model.components_
```

5. Display top 20 words for each topic identified by LDA and PLSA models

```python
def plot_top_words(model, feature_names, n_top_words, title):
    fig, axes = plt.subplots(1, 4, figsize=(30, 15), sharex=True)
    axes = axes.flatten()
    for topic_idx, topic in enumerate(model.components_):
        top_features_ind = topic.argsort()[: -n_top_words - 1 : -1]
        top_features = [feature_names[i] for i in top_features_ind]
        weights = topic[top_features_ind]

        ax = axes[topic_idx]
        ax.barh(top_features, weights, height=0.7)
        ax.set_title(f"Topic {topic_idx +1}", fontdict={"fontsize": 30})
        ax.invert_yaxis()
        ax.tick_params(axis="both", which="major", labelsize=20)
        for i in "top right left".split():
            ax.spines[i].set_visible(False)
        fig.suptitle(title, fontsize=40)

    plt.subplots_adjust(top=0.90, bottom=0.05, wspace=0.90, hspace=0.3)
    plt.show()
```