Assignment 1

CS5304 - Machine Learning

Deadline: February 5, 2018; Points: 15

For this assignment, you will be required to classify documents from the RCV1 dataset, which is an archive of over 800k manually categorized newswire stories made available by Reuters, Ltd. for research purposes [1]. The first 23,149 samples will be used for training, 10k are randomly chosen from the remaining for validation, and 50k will be randomly chosen for testing. The ids used for validation are provided in `validation.txt` and the ids for testing are hidden. Use scikit-learn [2] to load the data.

[1]: `http://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf`
[2]: `http://scikit-learn.org/stable/datasets/rcv1.html`

1. *k-Nearest-Neighbors and Naive Bayes*

1a. (3 Points) *Classification with kNN.* Use the KNeighborsClassifier from scikit-learn to classify documents in the validation set. Only consider the labels from `labels.txt`. Find a good value for K by exploring a few options. Wrap your classifier in a Python class called **CS5304KNNClassifier** and it should have a **train** and **predict** method. We will be calling these methods during grading. You'll also need to submit a file called `ks.txt` that includes the value for K you've chosen for each label. For this exercise, only use the first 1000 training examples.

1b. (3 Points) *Classification with NB.* Use the BernoulliNB from scikit-learn to classify documents in the validation set. Only consider the labels from `labels.txt`. Wrap your classifier in a Python class called **CS5304NBClassifier** and it should have a **train** and **predict** method. Use all the training examples.

1c. (1 Points) *Data Analysis.* Which label is the most difficult to classify? Why do you think this is the case?

2. *k-Means*

2a. (6 Points) *Classification with k-Means.* Use the KMeans from scikit-learn to classify documents in the validation set. Only consider the first 10 labels. Wrap this kMeans classifier in **CS5304KMeansClassifier** with a **train** and **predict** method, which will be called during grading. Use all the training examples.

Hint: Find the centroids for the training data. Pass these to scikit-learn's KMeans as the value for init.

2b. (1 Point) *Visualizing Clusters.* Visualize the predictions from KMeans for label 33 with a 2-dimensional plot. You'll need to choose a dimensionality reduction technique (such as TruncatedSVD). For plotting, use matplotlib.

2c. (1 Point) *Data Analysis.* Train a new KMeans model for label 33. This time, initialize the centroids randomly (if you use the scikit-learn method `fit`, then don't provide a value for `y`). Visualize the clusters using the same dimensions as from 2b. Do the clusters look similar as when you provided the labels? What do the clusters look like if you use 3 or 4 clusters instead of 2? Explain why.

FAQ

1. Q: How do I load the training data for RCV1?

   A: To load the training data...

   ```
   from sklearn.datasets import fetch_rcv1

   train_data = fetch_rcv1(subset='train')
   ```

   The first time this command is run, it will download the whole dataset (roughly 0.6 GB). This will take some time.

2. Q: How do I load the validation data for RCV1?

   A: The validation data is a subset of the test data.

   ```
   import numpy as np
   import pandas pd

   test_data = fetch_rcv1(subset='test')
   ids = pd.read_csv('validation.txt')
   mask = np.isin(test_data.sample_ids, ids)
   validation_data = test_data.data[mask]
   validation_target = test_data.target[mask]
   ```

3. Q: Can I train using the validation set or test set?

   A: No! This will result in a model whose prediction performance will not reflect what you might observe in the real world. You'll lose points for the entire question if you do this.

4. Q: Scikit-learn throws an error when I run `fetch_rcv1`. Help!

   A: Make sure you are using the latest version of scikit-learn. This was fixed in August 2017.