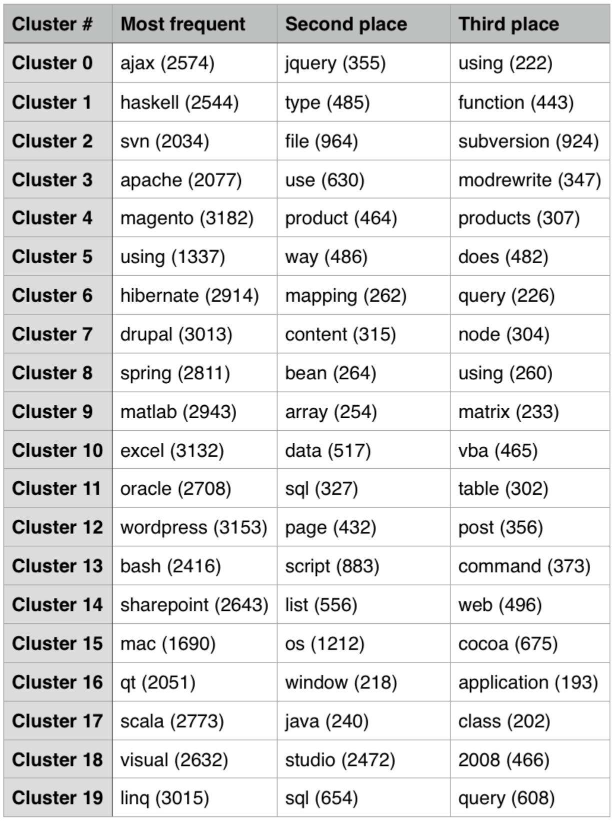
Machine Learning HW4 Report

b03901057 Chin-Cheng Chan

Problem 1

The table below shows the three most common words in each cluster.



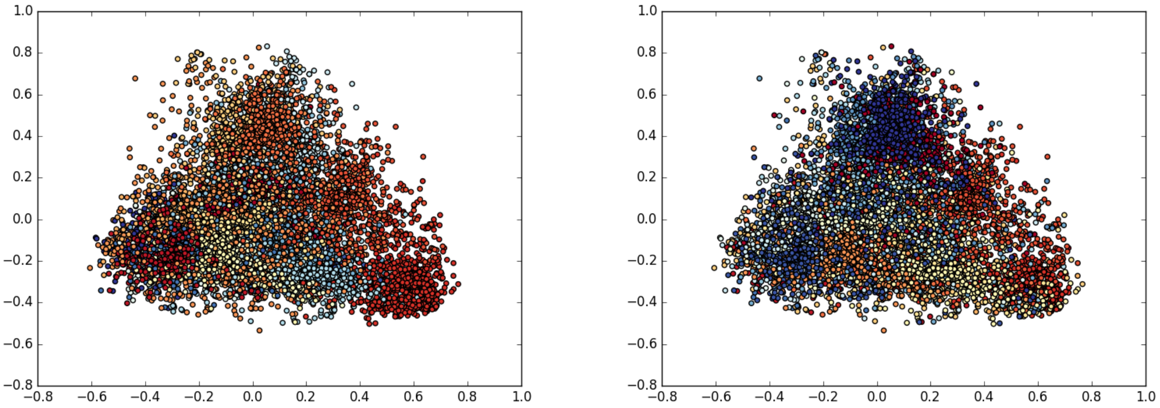
Cluster #	Most frequent	Second place	Third place
Cluster 0	ajax (2574)	jquery (355)	using (222)
Cluster 1	haskell (2544)	type (485)	function (443)
Cluster 2	svn (2034)	file (964)	subversion (924)
Cluster 3	apache (2077)	use (630)	modrewrite (347)
Cluster 4	magento (3182)	product (464)	products (307)
Cluster 5	using (1337)	way (486)	does (482)
Cluster 6	hibernate (2914)	mapping (262)	query (226)
Cluster 7	drupal (3013)	content (315)	node (304)
Cluster 8	spring (2811)	bean (264)	using (260)
Cluster 9	matlab (2943)	array (254)	matrix (233)
Cluster 10	excel (3132)	data (517)	vba (465)
Cluster 11	oracle (2708)	sql (327)	table (302)
Cluster 12	wordpress (3153)	page (432)	post (356)
Cluster 13	bash (2416)	script (883)	command (373)
Cluster 14	sharepoint (2643)	list (556)	web (496)
Cluster 15	mac (1690)	os (1212)	cocoa (675)
Cluster 16	qt (2051)	window (218)	application (193)
Cluster 17	scala (2773)	java (240)	class (202)
Cluster 18	visual (2632)	studio (2472)	2008 (466)
Cluster 19	linq (3015)	sql (654)	query (608)

The number in the parenthesis is the TF-IDF score. To calculate the TF-IDF score for each word in the cluster, I first collect all the titles that are classified as the same cluster, then I treat these titles as a new document and calculate the term frequency of each word. The inverse document frequency, however, are calculated over the whole input, not only the current document. Finally, we can collect the TF-IDF score for each word, and sort them by their scores.

It can be noted that those words that appear the most frequently are usually the tags themself. However, there are some cases like cluster 5, which does not extract much useful information. By comparing the those most frequent words with the ground-truth tags, we know cluster 5 should be ”cocoa”, which appears in cluster 15. This makes sense because mac, os x, and cocoa contain quite similar information and are often mentioned along with each other.

Problem 2

The two figures below show the tags of each title predicted by my model (a) and the ground-truth tags (b). These original high dimensional features vectors are projects to 2D by PCA for visualization. Each color in the two figures represent one cluster, however, the color in figure (a) does not relate to the color in (b). That is, a cluster represented by the color red in (a) has no relationships with that represented by the same color in (b). By comparing the clusters in the two figures, we can see that most clusters identified by my model has a



(a) Predicted labels

(b) Ground-truth labels

corresponding ground-truth cluster. Also, in both figures many clusters overlap with each other, and that is because we project a high dimensional data to a 2D plane.

Problem 3

1. **Model 1** BoW features extraction + LSA + K-Means: Only term frequency is used. In LSA, features are reduced to 20 dimensions.

2. **Model 2** TF-IDF features extraction + LSA + K-Means: Inverse document frequency is used in addition to term frequency. In LSA, features are reduced to 20 dimensions.

3. **Model 3** Word2Vec features extraction + K-Means: Use the Word2Vec package provided by gensim to train word vectors on ”docs.txt”. Then the features vectors of a title are averaged to be used as the feature vector of the title.

4. **Model 4** TF-IDF + PCA + K-Means: Same as Model 2, except that PCA is used to reduce data to 20 dimensions.

Model	Model 1	Model 2	Model 3	Model 4
Score	0.62	0.71	0.4	0.27

Model 1 v.s. Model 2: The difference between the two models is the consideration of inverse document frequency (IDF). The idea of IDF is to reduce the score of words that appear very often since these words may offer little information. Indeed, in our task, keywords like matlab, scala and so on appear less often then words like ”the” and ”using”...
Also, it should be noted that in order to achieve better performance, k-means in both models are set to find 25 clusters instead of 20. There will be more discussions on this point in the next question.

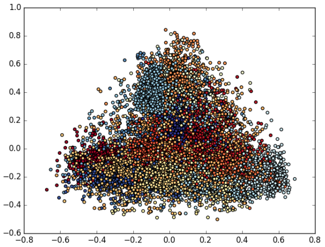
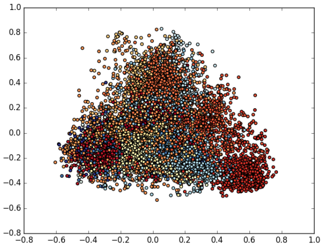
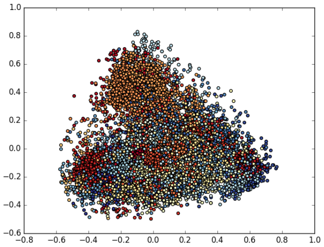
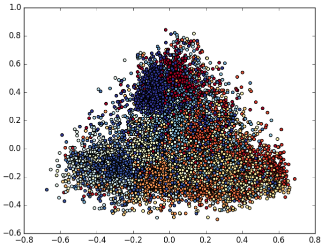
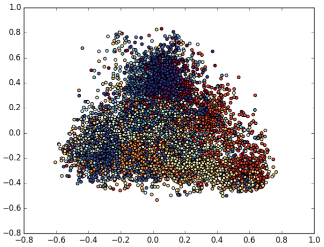
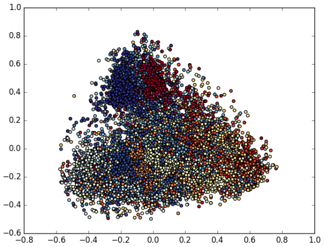
Model 2 v.s. Model 3: Though Word2Vec are perceived to be the best and the most popular word feature extractor now, using Word2Vec does not improve the results in my experiments. In fact, the performance of Model 3 is the worst among the three models. The first reason is that I possibly did not find the best parameters. In fact, I spent a lot of time tuning the parameters of Model2 and observed that these parameters are determining factors of the performance. Yet another reason is that the corpus on which the word vector is trained contains a lot of code blocks. Since the programming language syntax and variable names are usually not common english words and there is no easy way to distinguish them from natural language, the word vector model may thus be influenced.

Model 2 v.s. Model 4: From my results, LSA is more preferable. LSA utilizes a singular value decomposition (SVD) on the term by document matrix, while PCA utilizes principal components analysis on the term by term matrix. Since, we already have TF-IDF, we would certainly use LSA instead of PCA.

Problem 4

The performance of three different cluster numnbers: 20, 25, and 100 are compared below.

Cluster #	20	25	50
Score	0.58	0.71	0.81

Cluster	20	25	50
Predicted			
Ground -truth			

In general, increasing cluster number improves the results. This is because our concern is only to determine if two titles share the same tag. Using more clusters means that the algorithm is more capable to model more complex data distribution.