School of Computing and Information Systems
The University of Melbourne
COMP30027 MACHINE LEARNING (Semester 1, 2019)

Practical exercises: Week 9

Being able to process text using `scikit-learn` has suddenly become relevant to your interests[1]. Don't forget that we expect you to refer to the API (`http://scikit-learn.org/stable/modules/classes.html`) where necessary.

1. `scikit-learn` has an in-built text dataset, the "20 newsgroups corpus" (`http://scikit-learn.org/stable/datasets/twenty_newsgroups.html`), which contains a number of documents classified with a topic, based on the newsgroup in which it was posted.

   (a) Choose a couple of newsgroups that you think it would be interesting to discriminate between, like `'rec.autos'` and `'rec.motorcycles'`.
   We will use `'alt.atheism'` and `'talk.religion.misc'` as an example.

   (b) Load a training and test dataset, for example:

```
>>> from sklearn.datasets import fetch_20newsgroups
categories = ['alt.atheism','talk.religion.misc']
data_train = fetch_20newsgroups(subset='train', categories=categories,
                                shuffle=True, random_state=30027)
data_test = fetch_20newsgroups(subset='test', categories=categories,
                               shuffle=True, random_state=30027)
X_train = data_train.data
y_train = data_train.target
X_test = data_test.data
y_test = data_test.target
```

   (c) Examine a couple of documents, by referencing the list (`X_train[0]`). Can you accurately predict the class (`y_train[0]`) based on the text alone?

2. The document is currently a string, which `scikit-learn` can't use directly. However, it does have a number of "vectorizers" to help us construct a usable data representation.

   (a) Refresh yourself on the `DictVectorizer` — it is useful for mixed-type datasets, where we might wish to use numerical attributes directly, but transform nominal attributes into their "one-hot" representation. (We saw this for *Adult Census Income* in Week 7.)

   (b) More convenient for our needs is the `CountVectorizer`, which actually does quite a few preprocessing steps, but eventually gives us a dictionary which associates each word[2] in a text document with its frequency in that document:

```
from sklearn.feature_extraction.text import CountVectorizer
vectoriser = CountVectorizer()
X_train = vectoriser.fit_transform(X_train)
X_test = vectoriser.transform(X_test)
```

   (c) After "vectorizing" the data, what is the `shape` of `X_train` and `X_test`?

   (d) Are there any documents in `X_test` whose values are all 0? Why might this happen?

[1] While you may see this as an unpleasant consequence of having a linguist co-ordinating the subject, let me assure you that text is everywhere. Becoming familiar with some of its particularities may very well pay off in your future endeavours!

[2] The words are encoded as integers, as is typical for nominal features in `scikit-learn`.

3. Recall that we tried **feature filtering** in Week 6, using `SelectKBest`.

   (a) Find out what the best 10 features[3] were for your dataset, according to χ^2:

   ```
   from sklearn.feature_selection import SelectKBest, chi2
   x2 = SelectKBest(chi2, k=10)
   X_train = x2.fit_transform(X_train,y_train)
   for feat_num in x2.get_support(indices=True):
       print(vectoriser.get_feature_names()[feat_num])
   ```

   Do they correspond to your intuitions? Is there any evidence of the biases inherent in χ^2, referred to in the lectures? What if you look at deeper than 10?

   (b) Do the same thing for Mutual Information, instead of χ^2 (note that you want the classification version, not the regression version).

4. Build a classifier on the training dataset, and evaluate its Accuracy on the test set. Consider k-NN, and perhaps Naive Bayes or Decision Trees.

   (a) It's likely that the dataset is still small enough that you can build a model on the entire feature set (after the `CountVectorizer`, but before the `SelectKBest`) without crashing your computer. How well do these models predict the test data, using all of the features?

   (b) How does this compare with just the top 10 features?

   (c) Try some different values for the cut-off for `SelectKBest` — is it possible to improve upon the Accuracy observed for the models which use the entire feature set? Is this more true for some learners than others? Does your choice between χ^2 and Mutual Information make a difference?

[3] Remember, everything is a number, so we need to do some work to get the name back.