2019-03-21-Handout

March 16, 2019

```
In [1]: %pylab inline
        from sklearn import datasets, decomposition, feature_extraction
        news = datasets.fetch_20newsgroups()
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
In [2]: news.keys()
```

```
Out[2]: dict_keys(['data', 'filenames', 'target_names', 'target', 'DESCR'])
```

```
In [3]: print(news["DESCR"])
```

```
.. _20newsgroups_dataset:

The 20 newsgroups text dataset
------------------------------

The 20 newsgroups dataset comprises around 18000 newsgroups posts on
20 topics split in two subsets: one for training (or development)
and the other one for testing (or for performance evaluation). The split
between the train and test set is based upon a messages posted before
and after a specific date.

This module contains two loaders. The first one,
:func:`sklearn.datasets.fetch_20newsgroups`,
returns a list of the raw texts that can be fed to text feature
extractors such as :class:`sklearn.feature_extraction.text.CountVectorizer`
with custom parameters so as to extract feature vectors.
The second one, :func:`sklearn.datasets.fetch_20newsgroups_vectorized`,
returns ready-to-use features, i.e., it is not necessary to use a feature
extractor.

**Data Set Characteristics:**

    =================   ==========
    Classes                     20
    Samples total            18846
```

```
    Dimensionality               1
    Features                  text
    =================   ==========
```

```
Usage
~~~~~

The :func:`sklearn.datasets.fetch_20newsgroups` function is a data
fetching / caching functions that downloads the data archive from
the original `20 newsgroups website`_, extracts the archive contents
in the ``~/scikit_learn_data/20news_home`` folder and calls the
:func:`sklearn.datasets.load_files` on either the training or
testing set folder, or both of them::

  >>> from sklearn.datasets import fetch_20newsgroups
  >>> newsgroups_train = fetch_20newsgroups(subset='train')

  >>> from pprint import pprint
  >>> pprint(list(newsgroups_train.target_names))
  ['alt.atheism',
   'comp.graphics',
   'comp.os.ms-windows.misc',
   'comp.sys.ibm.pc.hardware',
   'comp.sys.mac.hardware',
   'comp.windows.x',
   'misc.forsale',
   'rec.autos',
   'rec.motorcycles',
   'rec.sport.baseball',
   'rec.sport.hockey',
   'sci.crypt',
   'sci.electronics',
   'sci.med',
   'sci.space',
   'soc.religion.christian',
   'talk.politics.guns',
   'talk.politics.mideast',
   'talk.politics.misc',
   'talk.religion.misc']

The real data lies in the ``filenames`` and ``target`` attributes. The target
attribute is the integer index of the category::

  >>> newsgroups_train.filenames.shape
  (11314,)
  >>> newsgroups_train.target.shape
  (11314,)
  >>> newsgroups_train.target[:10]
```

```
  array([ 7,  4,  4,  1, 14, 16, 13,  3,  2,  4])

It is possible to load only a sub-selection of the categories by passing the
list of the categories to load to the
:func:`sklearn.datasets.fetch_20newsgroups` function::

  >>> cats = ['alt.atheism', 'sci.space']
  >>> newsgroups_train = fetch_20newsgroups(subset='train', categories=cats)

  >>> list(newsgroups_train.target_names)
  ['alt.atheism', 'sci.space']
  >>> newsgroups_train.filenames.shape
  (1073,)
  >>> newsgroups_train.target.shape
  (1073,)
  >>> newsgroups_train.target[:10]
  array([0, 1, 1, 1, 0, 1, 1, 0, 0, 0])

Converting text to vectors
~~~~~~~~~~~~~~~~~~~~~~~~~~

In order to feed predictive or clustering models with the text data,
one first need to turn the text into vectors of numerical values suitable
for statistical analysis. This can be achieved with the utilities of the
``sklearn.feature_extraction.text`` as demonstrated in the following
example that extract `TF-IDF`_ vectors of unigram tokens
from a subset of 20news::

  >>> from sklearn.feature_extraction.text import TfidfVectorizer
  >>> categories = ['alt.atheism', 'talk.religion.misc',
  ...               'comp.graphics', 'sci.space']
  >>> newsgroups_train = fetch_20newsgroups(subset='train',
  ...                                       categories=categories)
  >>> vectorizer = TfidfVectorizer()
  >>> vectors = vectorizer.fit_transform(newsgroups_train.data)
  >>> vectors.shape
  (2034, 34118)

The extracted TF-IDF vectors are very sparse, with an average of 159 non-zero
components by sample in a more than 30000-dimensional space
(less than .5% non-zero features)::

  >>> vectors.nnz / float(vectors.shape[0])       # doctest: +ELLIPSIS
  159.01327...

:func:`sklearn.datasets.fetch_20newsgroups_vectorized` is a function which
returns ready-to-use token counts features instead of file names.
```

```
.. _`20 newsgroups website`: http://people.csail.mit.edu/jrennie/20Newsgroups/
.. _`TF-IDF`: https://en.wikipedia.org/wiki/Tf-idf


Filtering text for more realistic training
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

It is easy for a classifier to overfit on particular things that appear in the
20 Newsgroups data, such as newsgroup headers. Many classifiers achieve very
high F-scores, but their results would not generalize to other documents that
aren't from this window of time.

For example, let's look at the results of a multinomial Naive Bayes classifier,
which is fast to train and achieves a decent F-score::

  >>> from sklearn.naive_bayes import MultinomialNB
  >>> from sklearn import metrics
  >>> newsgroups_test = fetch_20newsgroups(subset='test',
  ...                                      categories=categories)
  >>> vectors_test = vectorizer.transform(newsgroups_test.data)
  >>> clf = MultinomialNB(alpha=.01)
  >>> clf.fit(vectors, newsgroups_train.target)
  MultinomialNB(alpha=0.01, class_prior=None, fit_prior=True)

  >>> pred = clf.predict(vectors_test)
  >>> metrics.f1_score(newsgroups_test.target, pred, average='macro')  # doctest: +ELLIPSIS
  0.88213...

(The example :ref:`sphx_glr_auto_examples_text_plot_document_classification_20newsgroups.py` shu
the training and test data, instead of segmenting by time, and in that case
multinomial Naive Bayes gets a much higher F-score of 0.88. Are you suspicious
yet of what's going on inside this classifier?)

Let's take a look at what the most informative features are:

  >>> import numpy as np
  >>> def show_top10(classifier, vectorizer, categories):
  ...     feature_names = np.asarray(vectorizer.get_feature_names())
  ...     for i, category in enumerate(categories):
  ...         top10 = np.argsort(classifier.coef_[i])[-10:]
  ...         print("%s: %s" % (category, " ".join(feature_names[top10])))
  ...
  >>> show_top10(clf, vectorizer, newsgroups_train.target_names)
  alt.atheism: edu it and in you that is of to the
  comp.graphics: edu in graphics it is for and of to the
  sci.space: edu it that is in and space to of the
  talk.religion.misc: not it you in is that and to of the
```

```
You can now see many things that these features have overfit to:

- Almost every group is distinguished by whether headers such as
  ``NNTP-Posting-Host:`` and ``Distribution:`` appear more or less often.
- Another significant feature involves whether the sender is affiliated with
  a university, as indicated either by their headers or their signature.
- The word "article" is a significant feature, based on how often people quote
  previous posts like this: "In article [article ID], [name] <[e-mail address]>
  wrote:"
- Other features match the names and e-mail addresses of particular people who
  were posting at the time.

With such an abundance of clues that distinguish newsgroups, the classifiers
barely have to identify topics from text at all, and they all perform at the
same high level.

For this reason, the functions that load 20 Newsgroups data provide a
parameter called **remove**, telling it what kinds of information to strip out
of each file. **remove** should be a tuple containing any subset of
``('headers', 'footers', 'quotes')``, telling it to remove headers, signature
blocks, and quotation blocks respectively.

  >>> newsgroups_test = fetch_20newsgroups(subset='test',
  ...                                      remove=('headers', 'footers', 'quotes'),
  ...                                      categories=categories)
  >>> vectors_test = vectorizer.transform(newsgroups_test.data)
  >>> pred = clf.predict(vectors_test)
  >>> metrics.f1_score(pred, newsgroups_test.target, average='macro')  # doctest: +ELLIPSIS
  0.77310...

This classifier lost over a lot of its F-score, just because we removed
metadata that has little to do with topic classification.
It loses even more if we also strip this metadata from the training data:

  >>> newsgroups_train = fetch_20newsgroups(subset='train',
  ...                                       remove=('headers', 'footers', 'quotes'),
  ...                                       categories=categories)
  >>> vectors = vectorizer.fit_transform(newsgroups_train.data)
  >>> clf = MultinomialNB(alpha=.01)
  >>> clf.fit(vectors, newsgroups_train.target)
  MultinomialNB(alpha=0.01, class_prior=None, fit_prior=True)

  >>> vectors_test = vectorizer.transform(newsgroups_test.data)
  >>> pred = clf.predict(vectors_test)
  >>> metrics.f1_score(newsgroups_test.target, pred, average='macro')  # doctest: +ELLIPSIS
  0.76995...
```

```
Some other classifiers cope better with this harder version of the task. Try
running :ref:`sphx_glr_auto_examples_model_selection_grid_search_text_feature_extraction.py` wit
the ``--filter`` option to compare the results.

.. topic:: Recommendation

  When evaluating text classifiers on the 20 Newsgroups data, you
  should strip newsgroup-related metadata. In scikit-learn, you can do this by
  setting ``remove=('headers', 'footers', 'quotes')``. The F-score will be
  lower because it is more realistic.

.. topic:: Examples

   * :ref:`sphx_glr_auto_examples_model_selection_grid_search_text_feature_extraction.py`

   * :ref:`sphx_glr_auto_examples_text_plot_document_classification_20newsgroups.py`
```

1 Let's do some LDA work

We work with data from the 20 Newsgroups dataset. First, use TF-IDF to vectorize the text, and then train an LDA model on the result.

In [4]:
```
seed(42)
vectorizer = feature_extraction.text.TfidfVectorizer()
```

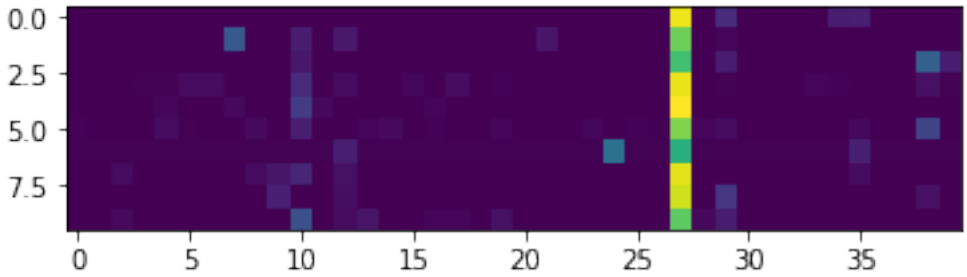
In [5]:
```
news_train = datasets.fetch_20newsgroups(subset="train")
vectors = vectorizer.fit_transform(news_train["data"])

news_test = datasets.fetch_20newsgroups(subset="test")
vectors_test = vectorizer.transform(news_test["data"])

feature_names = vectorizer.get_feature_names()
```

In [6]:
```
lda = decomposition.LatentDirichletAllocation(n_components=40)
lda.fit(vectors)
```

Out[6]:
```
LatentDirichletAllocation(batch_size=128, doc_topic_prior=None,
                          evaluate_every=-1, learning_decay=0.7,
                          learning_method='batch', learning_offset=10.0,
                          max_doc_update_iter=100, max_iter=10, mean_change_tol=0.001,
                          n_components=40, n_jobs=None, n_topics=None, perp_tol=0.1,
                          random_state=None, topic_word_prior=None,
                          total_samples=1000000.0, verbose=0)
```

In [7]:
```
imshow(lda.transform(vectors_test[:10,:]))
```

Out[7]:
```
<matplotlib.image.AxesImage at 0x117524eb8>
```




```
In [8]: for topic_idx, topic in enumerate(lda.components_):
            print(f"Topic #{topic_idx} ", end="")
            print(" ".join([feature_names[i]
                            for i in topic.argsort()[:-10:-1]]))
```

```
Topic #0 ashok iti allen ssto sherzer biochem aws needles biochemistry
Topic #1 arromdee wharton jyusenkyou seirio sp1marse dixie compstat marco eder
Topic #2 dresden inf beck tu irzr17 andre_beck andre tude irs
Topic #3 serdar argic zuma sera extermination armenians 1920 1919 ohanus
Topic #4 dartmouth uvic nubus hades pds gainey hrivnak lafibm gtd597a
Topic #5 wfu den mmc clesun ricardo 68070 mcmains tsa delcoelect
Topic #6 handheld jmd arras cube jr0930 kinsey arcade alec 1708
Topic #7 buffalo acsu b30 dtmedin catbyte ubvmsb absolutes hammerl weiss
Topic #8 fist iscp bellcore vera pierson noye shakala noyes reedr
Topic #9 almanac sunysb cjackson curtin wilson stony brook infante chicogo
Topic #10 scsi israeli ide controller drives bus berkeley hd killed
Topic #11 astein crohn isi immunization jas apollo immunizations oulu ncratl
Topic #12 uicvm uic shuttle alomar higgins ico resurrection baalke jpl
Topic #13 oracle mcguire ebosco borden davewood ualberta selective uclink unauthenticated
Topic #14 ax miavx1 muohio sensor chimps maine quicktime mydisplay jpc
Topic #15 harris ssd synoptics omen zeos neutral outlet dorin marka
Topic #16 geb gordon banks pitt cadre skepticism surrender chastity n3jxp
Topic #17 jake hamburg bontchev bony bony1 livni hydro informatik hernlem
Topic #18 alaska aurora nsmca acad3 pyron shostack dseg skndiv jacked
Topic #19 feustel drake sq acad engin gehrels esin beckman lee
Topic #20 uga ai covington mcovingt georgia athens aisun3 irq eliot
Topic #21 lib xmu libxmu wg2 waii testing boell jhesse hesse
Topic #22 lurie luriem liberalizer 1280x1024 allegheny alleg sepinwall sepinwal smale
Topic #23 oasys dt nswc carderock relays bethesda alphacdc scicom wats
Topic #24 hcf cview petch zisfein jhunix journalism useragent nuntius xxdate
Topic #25 towers oswego imag alaa plants ozonehole belton ching karish
Topic #26 srl ford slee01 corn chi vcu cabell bos lang
Topic #27 the to of and in is that it edu
Topic #28 betz idbsu gozer deane ncratl brandeis bms randall skybridge
Topic #29 mouse pittsburgh printer nec gld pa rangers font montreal
```

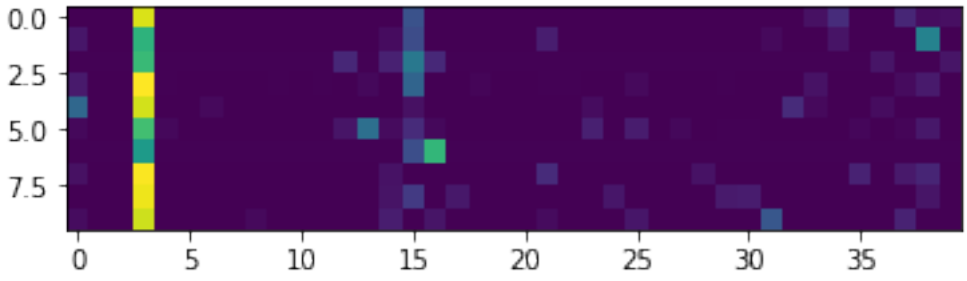
```
Topic #30 shai guday mule gt0523e cursor niguma nlm charlie auvm
Topic #31 gt1091a oeinck mont bezier curves timucin detecting lsid ferdinand
Topic #32 noring erickson steinly topaz telix gordian infj guyd deltabox
Topic #33 henry zoology zoo spencer utzoo kipling gec arens yigal
Topic #34 unm wpi carina ___ pettefar npet ripem erics noah
Topic #35 uv utarlg nuet_ke uta pts wv msstate isis frip
Topic #36 br steveh hendricks libertarians dominance vida regulation thor mdavcr
Topic #37 halat hiram cosmo angmar pooh bears alfalfa pro kou
Topic #38 clipper encryption cramer optilink armenians intercon mathew gay drug
Topic #39 seas risc gwu dsu qazi louray instruction bitzm eniac
```

Notice how topic 27 has almost all the weight in the predictions, and has the top words

```
Topic #27 the to of and in is that it edu
```

We will need to use **stop words** to clean up the dataset - ALL documents will have plenty of `the`, `to`, `of`, ...

```
In [9]: vectorizer = feature_extraction.text.TfidfVectorizer(stop_words="english")

In [10]: vectors = vectorizer.fit_transform(news_train["data"])
         vectors_test = vectorizer.transform(news_test["data"])
         feature_names = vectorizer.get_feature_names()

In [11]: lda.fit(vectors)
         imshow(lda.transform(vectors_test[:10,:]))

Out[11]: <matplotlib.image.AxesImage at 0x11753e7f0>
```




```
In [12]: for topic_idx, topic in enumerate(lda.components_):
             print(f"Topic #{topic_idx} ", end="")
             print(" ".join([feature_names[i]
                             for i in topic.argsort()[:-10:-1]]))
```

```
Topic #0 keith livesey sgi caltech schneider morality solntze wpd allan
Topic #1 nada kth cooling towers nicho sehari hemul tower jwa
Topic #2 spss caronni ndw dong dlneal apgea dougb dil clutchless
Topic #3 edu com subject lines organization writes article university posting
Topic #4 behanna higgins fnal syl duke absolutes ashok fnalf bitzm
Topic #5 jayne kulikauskas wpi mmalt liturgy guild relays mydisplay ching
Topic #6 drake acad steiner utoledo ssc lerc vu aberystwyth 1964
Topic #7 mahan tgv louray nuet_ke gwu panayiotakis sussex lazarus pts
Topic #8 stratus sw cdt helmet ati tavares synoptics ripem vos
Topic #9 dyer pyron dseg sabbath ti stafford spdcc skndiv winona
Topic #10 mcguire sutherland ether evans mjs deane bezier blaine bgardner
Topic #11 howland hydro jody levine curt k_p jlevine yadlowsky pmy
Topic #12 fpu skidmore expose whaley tuinstra oasys dt buzz carderock
Topic #13 clarinet br candida brad noring nist yeast steveh raider
Topic #14 kaldis unm dwyer cnsvax d012s658 sni uwec mchp gsh7w
Topic #15 team scsi game edu players israel pitt ca stanford
Topic #16 petch grass valley gvg mpce mq gvg47 chuck almanac
Topic #17 mangoe cosmo angmar wingate fourd lafibm lafayette charley vb30
Topic #18 jake bony bony1 livni b30 uio ifi cpr dtmedin
Topic #19 risc instruction tufts bmd mu batman bissell vera liar
Topic #20 benedikt i3150101 dbstu1 rz mcs rosenau aludra tu erics
Topic #21 ranck bmug fisher coprocessor sdsu babb gec nthu joesbar
Topic #22 shearson hulman pmetzger philips metzger rickert jarthur claremont regulated
Topic #23 llnl crohn pythagorean amoco mont rauser sanderson lawrence migraine
Topic #24 alchemy chem utoronto gerald golchowy olchowy sherri journalism nichols
Topic #25 hernlem omen crypt 8051 chess lebanese hezbollah alaa arens
Topic #26 mikey cacs sq usl pgf comet jupiter fraering radford
Topic #27 ax feustel polygon ipser diablo routine technet csd marka
Topic #28 nysernet broward horne astein dealy an030 logo cview stein
Topic #29 infante tony lib ch981 trial beckman waii duke wg2
Topic #30 hiram qualcomm gainey shai svoboda plymouth guday rdippold qualcom
Topic #31 intercon amanda gtoal uicvm uic walker kratz greek toal
Topic #32 bontchev hamburg dresden informatik fbihh tu inf vesselin beck
Topic #33 ai uga covington mcovingt georgia aisun3 athens lehigh 542
Topic #34 ists stpl rauser dchhabra jr0930 europeans puck ryerson tude
Topic #35 mcmaster espn jb maccs videocart adrian scorer holly telix
Topic #36 okcforum handheld osrhe jmd spacecraft betz conner gozer idbsu
Topic #37 gld umich engin centerline easter hallam dare azerbaijan cunixb
Topic #38 henry alaska zoo spencer israel lunar toronto solar zoology
Topic #39 egalon huot nubus cray gajarsky ricardo njin convenient mcmains
```

Now we see that almost all topics are predicted to be number 3. The top predictive words for this category are `edu` and `com`.

...so we are basically recognizing sender email addresses. Ooops.

Let's take a look at a single entry, to see where those domain names are coming from?

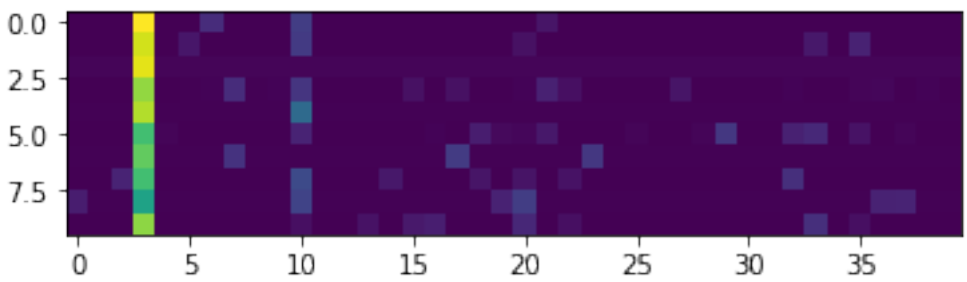
```
In [13]: print(news_train["data"][0])
```

```
From: lerxst@wam.umd.edu (where's my thing)
Subject: WHAT car is this!?
Nntp-Posting-Host: rac3.wam.umd.edu
Organization: University of Maryland, College Park
Lines: 15

 I was wondering if anyone out there could enlighten me on this car I saw
the other day. It was a 2-door sports car, looked to be from the late 60s/
early 70s. It was called a Bricklin. The doors were really small. In addition,
the front bumper was separate from the rest of the body. This is
all I know. If anyone can tellme a model name, engine specs, years
of production, where this car is made, history, or whatever info you
have on this funky looking car, please e-mail.

Thanks,
- IL
   ---- brought to you by your neighborhood Lerxst ----
```

Oh look! We are including both a **header** with metadata and a **footer** with stereotypical information.

Let's not do that.

```
In [14]: news_train = datasets.fetch_20newsgroups(subset="train", remove=("headers", "footers",
         vectors = vectorizer.fit_transform(news_train["data"])
         news_test = datasets.fetch_20newsgroups(subset="test", remove=("headers", "footers", "q
         vectors_test = vectorizer.transform(news_test["data"])
         feature_names = vectorizer.get_feature_names()

In [15]: lda.fit(vectors)
         imshow(lda.transform(vectors_test[:10,:]))

Out[15]: <matplotlib.image.AxesImage at 0x117a7a278>
```



```
In [16]: for topic_idx, topic in enumerate(lda.components_):
             print(f"Topic #{topic_idx} ", end="")
             print(" ".join([feature_names[i]
                             for i in topic.argsort()[:-10:-1]]))
```

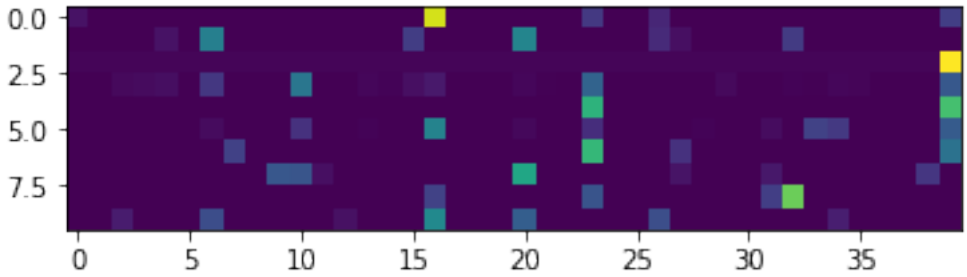
```
Topic #0 ax planes lucifer magellan bm grows larc onur yalcin
Topic #1 allah islam rushdie jb cview dad retarded chris islamic
Topic #2 bos chi yankees cal det van que hash md5
Topic #3 just like know don people think does use thanks
Topic #4 geb cadre skepticism shameful intellect n3jxp chastity dsl pitt
Topic #5 sabbath ceremonial deeds printer tie ghetto gentiles kovalev ver
Topic #6 gl nist subscribe ncsl borland hillary cheat widget csrc
Topic #7 satan irq angels mom mouse com1 freewill com3 cobb
Topic #8 ghostscript selfish slavery wc wayne 360k blacks tartar workspaces
Topic #9 ink observations bj 27 deskjet cell 1st wdve stevens
Topic #10 moral fast folks morality sin women blood lord hours
Topic #11 config font bat __ pitching autoexec kc lopez offense
Topic #12 lib usr alomar postscript icon icons baerga phigs compiled
Topic #13 chicago angeles los detroit vancouver montreal pts finals calgary
Topic #14 ssf network chop ini lite numlock p1 cci hartmann
Topic #15 scsi mouse printer modem disks microsoft fonts motherboard audio
Topic #16 stephanopoulos bait rtrace dictionary sentiments danny naval astemizole ocean
Topic #17 hitter aids ites tree lankford clayton pinch q700 dean
Topic #18 ati candida 3401 deletion toshiba turbo manta yeast vgalogo
Topic #19 bmp auto jpeg extended bitmap lssu ottoman fun maine
Topic #20 dma tek ico blew randy domain manhattan bobbe sank
Topic #21 armenian armenians turkish arabs turks turkey armenia army des
Topic #22 test weaver fundamentalist supreme detector font fonts bound handguns
Topic #23 uv ripem nada kth yzerman tte deluxe revelation marital
Topic #24 adam das shostack harvard rr speakers survivor woof tenants
Topic #25 tanks __ atari dpi mask subaru 2600 bitmap headline
Topic #26 clemens meaningless xarchie mil tcora coradeschi pica leg xservers
Topic #27 qur winmarks adaptor azerbaijani islam fpu sumgait muslims tt
Topic #28 authority alarm 231 dortmund povray searching vesa xv congruent
Topic #29 gateway nick biker gang yeast ton piece lud 1069
Topic #30 displays rec implement wiring cpsr leftover winners pointer savard
Topic #31 ulf mf cubs answered idacom critus posed ext reboot
Topic #32 vcr 04 jays pit 03 mpeg shack device tor
Topic #33 adaptec modem 32bis jpg rz350 pmp 1964 xv 001
Topic #34 feustel morris n9myi obp alomar rbi slg ozzie motorcycles
Topic #35 espn lunar played cup wins gm jose traded caps
Topic #36 dtmedin catbyte ingr b30 ditto 205 accelerator gm accelerators
Topic #37 hello instruction ampere birthday bel unc null sleeve risc
Topic #38 nubus pds quakers kkeller keller ivy champs upenn sas
```

```
Topic #39 s1 s2 buggy creed purchased ironic u1 islam invisible
```

But wait a moment here now. LDA works on a **word count matrix** whereas TF-IDF produces a weighting on the word counts.

So the frequency of each word in each document is not currently matching the probabilistic model in LDA. Let's fix that.

```
In [17]: vectorizer = feature_extraction.text.CountVectorizer(stop_words="english")

In [18]: vectors = vectorizer.fit_transform(news_train["data"])
         vectors_test = vectorizer.transform(news_test["data"])
         feature_names = vectorizer.get_feature_names()

In [19]: lda.fit(vectors)
         imshow(lda.transform(vectors_test[:10,:]))

Out[19]: <matplotlib.image.AxesImage at 0x1179decc0>
```




```
In [20]: for topic_idx, topic in enumerate(lda.components_):
             print(f"Topic #{topic_idx} ", end="")
             print(" ".join([feature_names[i]
                             for i in topic.argsort()[:-10:-1]]))
```

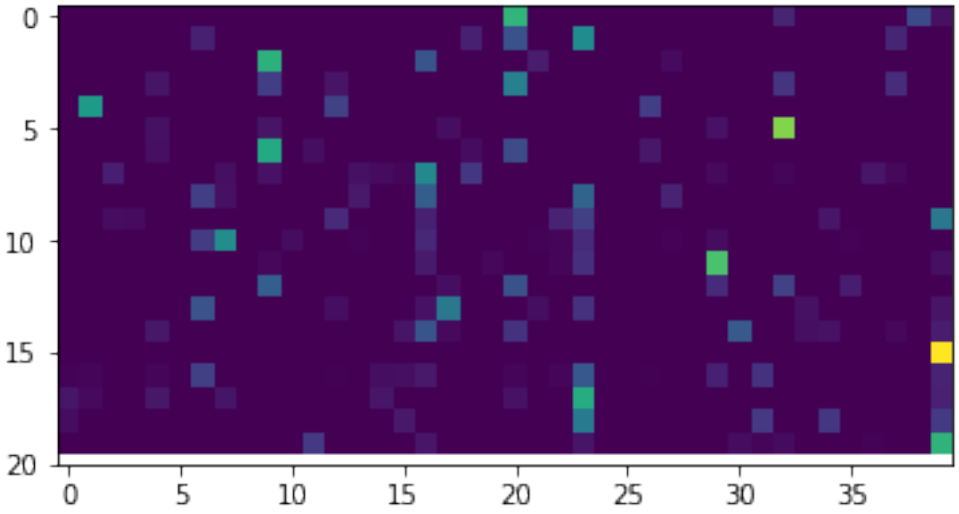
```
Topic #0 lib libxmu xmu doug symbol university undefined wip com3
Topic #1 water just com dept use right fans don fan
Topic #2 00 said 02 armenian san new 03 000 01
Topic #3 anonymous posting henrik bm people anonymity service want anon
Topic #4 file right states amendment gun militia congress good control
Topic #5 ax b8f 145 a86 max 1d9 0t 2di 34u
Topic #6 said didn just don know people like went time
Topic #7 team 10 game 25 11 12 15 season 16
Topic #8 tyre mydisplay june lines 1968 gc 1969 cell draw
Topic #9 drive scsi disk card windows use hard dos pc
Topic #10 israel jews israeli turkish jewish people greek arab like
```

```
Topic #11 theory universe larson light van uv tt physical het
Topic #12 game players cubs alomar baseball lot suck games like
Topic #13 500 000 250 kk 333 400 100 200 win
Topic #14 000 committee firearms batf pope sb echo hojali hb
Topic #15 don just know gay people new men dog p2
Topic #16 car like just good don time bike know ve
Topic #17 cross allocation edu picture unit linked cview 45 sleeve
Topic #18 00 wire ground 50 use circuit power wiring new
Topic #19 insurance attack private row maria col points time op_cols
Topic #20 mail edu image ftp files information data software available
Topic #21 file entry output program entries ripem section rules printf
Topic #22 mr stephanopoulos president 00 think know don going dos
Topic #23 people don think government like just make good time
Topic #24 cx w7 c_ uw t7 ck chz lk hz
Topic #25 ax max g9v pl b8f a86 1t 3t bhj
Topic #26 monitor video color screen vga card bit apple like
Topic #27 db mov bh cs si section al byte di
Topic #28 ra comments machines type lucifer m5 mv canon algebra
Topic #29 key chip keys clipper encryption law bit use des
Topic #30 space nasa earth launch orbit shuttle satellite lunar moon
Topic #31 gun guns crime koresh control use rate self don
Topic #32 edu window com server use file windows motif available
Topic #33 technology privacy information encryption 1993 new research use security
Topic #34 edu msg food patients disease gordon banks soon com
Topic #35 armenian armenians turkish genocide people armenia russian turks soviet
Topic #36 magi mt mv __ gl mh m0 0c mg
Topic #37 just radar don south ve think does like war
Topic #38 mp reds mc precision atheism mf m_ mw mq
Topic #39 god people jesus think does say believe don just
```

Now **FINALLY** we are getting some spread in our topic allocations!

Let's have a look at how our 40 detected topics distribute among the newsgroup identities, shall we?

```
In [21]: topic_distr = zeros((len(news["target_names"]),lda.n_components))
         message_distr = lda.transform(vectors_test)
         for i in range(len(news_test["target"])):
             topic_distr[news_test["target"][i],:] = message_distr[i,:]

In [22]: imshow(topic_distr)
         yticks([0,5,10,15,20])

Out[22]: ([<matplotlib.axis.YTick at 0x11755ca90>,
           <matplotlib.axis.YTick at 0x1179e4358>,
           <matplotlib.axis.YTick at 0x1174a45c0>,
           <matplotlib.axis.YTick at 0x1174e9c88>,
           <matplotlib.axis.YTick at 0x117a355c0>],
          <a list of 5 Text yticklabel objects>)
```




```
In [23]: (topic_distr.argmax(axis=0) == 4).nonzero()

Out[23]: (array([ 1,  5,  8, 12, 24, 25, 26, 28]),)

In [24]: print("Topic keywords by newsgroup")
         print("==========================\n\n")

         for ix, ng in enumerate(news_test["target_names"]):
             print(f"{ng}")
             for topic_idx in (topic_distr.argmax(axis=0) == ix).nonzero()[0]:
                 print(f"\tTopic #{topic_idx} ({topic_distr[ix,topic_idx]:.2f}): ", end="")
                 print(" ".join([feature_names[i]
                     for i in lda.components_[topic_idx,:].argsort()[:-8:-1]]))


Topic keywords by newsgroup
==========================


alt.atheism
        Topic #20 (0.64): mail edu image ftp files information data
        Topic #38 (0.22): mp reds mc precision atheism mf m_
comp.graphics
comp.os.ms-windows.misc
        Topic #9 (0.62): drive scsi disk card windows use hard
        Topic #21 (0.07): file entry output program entries ripem section
comp.sys.ibm.pc.hardware
        Topic #37 (0.12): just radar don south ve think does
comp.sys.mac.hardware
```

```
        Topic #1 (0.53): water just com dept use right fans
        Topic #5 (0.00): ax b8f 145 a86 max 1d9 0t
        Topic #8 (0.00): tyre mydisplay june lines 1968 gc 1969
        Topic #12 (0.19): game players cubs alomar baseball lot suck
        Topic #24 (0.00): cx w7 c_ uw t7 ck chz
        Topic #25 (0.00): ax max g9v pl b8f a86 1t
        Topic #26 (0.18): monitor video color screen vga card bit
        Topic #28 (0.00): ra comments machines type lucifer m5 mv
comp.windows.x
        Topic #32 (0.79): edu window com server use file windows
misc.forsale
rec.autos
        Topic #2 (0.08): 00 said 02 armenian san new 03
        Topic #16 (0.46): car like just good don time bike
        Topic #18 (0.16): 00 wire ground 50 use circuit power
        Topic #36 (0.06): magi mt mv __ gl mh m0
rec.motorcycles
        Topic #13 (0.07): 500 000 250 kk 333 400 100
        Topic #27 (0.10): db mov bh cs si section al
rec.sport.baseball
        Topic #3 (0.03): anonymous posting henrik bm people anonymity service
        Topic #22 (0.10): mr stephanopoulos president 00 think know don
rec.sport.hockey
        Topic #7 (0.48): team 10 game 25 11 12 15
        Topic #10 (0.03): israel jews israeli turkish jewish people greek
sci.crypt
        Topic #19 (0.02): insurance attack private row maria col points
        Topic #29 (0.69): key chip keys clipper encryption law bit
sci.electronics
        Topic #35 (0.08): armenian armenians turkish genocide people armenia russian
sci.med
        Topic #6 (0.25): said didn just don know people like
        Topic #17 (0.39): cross allocation edu picture unit linked cview
        Topic #33 (0.04): technology privacy information encryption 1993 new research
sci.space
        Topic #4 (0.06): file right states amendment gun militia congress
        Topic #30 (0.27): space nasa earth launch orbit shuttle satellite
soc.religion.christian
        Topic #39 (0.97): god people jesus think does say believe
talk.politics.guns
talk.politics.mideast
        Topic #0 (0.06): lib libxmu xmu doug symbol university undefined
        Topic #14 (0.06): 000 committee firearms batf pope sb echo
        Topic #23 (0.60): people don think government like just make
talk.politics.misc
        Topic #15 (0.07): don just know gay people new men
        Topic #31 (0.16): gun guns crime koresh control use rate
        Topic #34 (0.16): edu msg food patients disease gordon banks
```

```
talk.religion.misc
        Topic #11 (0.16): theory universe larson light van uv tt
```

In [ ]: