-
-
-
title: “Machine learning 2"

Aims of this tutorial:

- finding stylistic patterns in YouTube vlogs
- examining content commonalities in missing person details
- replicating an authorship prediction example
- finding patterns in your own writing

Note: this tutorial assumes that you have installed the `cluster` package.

## Task 1: Examining style patterns in the YouTube corpus

For this task, we will use a larger version of the YouTube corpus. Access the corpus from https://github.com/ben-aaron188/X0050/blob/master/data/vlogs_corpus_10k.RData and load the .RData file called `youtube_corpus_10k.RData` which contains 10,000 raw transripts of YouTube vlogs.

```{r}
# load the data
load('/Users/GitHub/X0050/data/vlogs_corpus_10k.RData')
```

Now use the techniques learned in the past weeks and extract, for each transcript, the readability (choose a metric yourself) and the number of words. Also check whether you might need to exclude some data.

~~~
```{r}
library(quanteda)
library(stringr)
# no. of tokens
vlogs_corpus_10k$ntok = ntoken(vlogs_corpus_10k$text)
# readability: note that given the non-punnctuated nature of the vlog
transcripts, we could
# decide to choose our own metric. Here, we simply use the chars/word
vlogs_corpus_10k$char_per_word = (nchar(vlogs_corpus_10k$text) -
str_count(vlogs_corpus_10k$text, " "))/vlogs_corpus_10k$ntok
# note that we removed the white space here by counting the occurrences and
subtracting it from the number of characters (nchar counts white space as a
character!)
head(vlogs_corpus_10k[, 4:5])
```
~~~

Use these two variables with unsupervised machine learning (here: k-means) to assess whether there are underlying patterns in the way in which language (operationalised here through the readability and length of the transcript) is used:

Specifically, follow these steps:

1. plot the data
2. determine k
3. build a final k-means model
4. plot the data with cluster assignment _(Hint: you can get the clusters from the final model object)_
5. interpret the clusters

Note: should the preliminary analysis not be conclusive about $k$, then you run the model with $k=3$.

~~~
```{r echo=T}
# plotting the data (here: only two dimensions)
{plot(x = vlogs_corpus_10k$ntok
  , y = vlogs_corpus_10k$char_per_word
~~~

~~~
  , pch = 19)}
```
~~~

~~~
```{r message=FALSE, warning=FALSE,  echo=T}
# determine k
wss = numeric()
for(i in 1:10){
 kmeans_model_temp = kmeans(x = vlogs_corpus_10k[, 4:5]
            , centers = i
            , iter.max = 10
            , nstart = 10)
 wss[i] = kmeans_model_temp$tot.withinss
}
{plot(wss)}
```
~~~

~~~
```{r echo=T}
# settle for k=2 for the final model
final_kmeans = kmeans(x = vlogs_corpus_10k[, 4:5]
          , centers = 2
          , iter.max = 10
          , nstart = 10)
```
~~~

~~~
```{r echo=TRUE}
# plot the data with cluster assignment
# we add a new variable to our main data frame that includes the cluster
membership
vlogs_corpus_10k$cluster = final_kmeans$cluster
# plot the data with colour-coded clusters
{plot(x = vlogs_corpus_10k$ntok
  , y = vlogs_corpus_10k$char_per_word
  , col = vlogs_corpus_10k$cluster
~~~

~~~
  , pch=19)}
```
~~~

~~~
```{r}
# interpret clusters: look at the cluster means
final_kmeans$centers
# We see that cluster 1 has almost twice the size (at its geometric center)
of cluster 2 in terms of the number of tokens, but is slighlty smaller on
the characters per word.
```
~~~

## Task 2: Common topics in FBI missing persons data

Use the data collected in Task 1 of the [tutorial of week 2](https://raw.githack.com/ben-aaron188/X0050/master/tutorials/week_3/week3_tutorial_secu0050_nb.html). The free text details information of the missing person data might give us some indication about commonalities in the descriptions.

Use the techniques from the previous weeks to extract common (here: use a sparsity correciton yourself) unigrams and bigrams from the text fields.

~~~
```{r}
# Use your own data here.
# Steps you need to follow:
# - extract ngrams
# - use ngrams frequencies as features
# - run the k-means model pipeline (as detailed in Task 1) on that data
# - interpret the clusters by their most prominent ngram differences
```
~~~

Once you have done so, use unsupervised learning to assess whether the use of these frequent unigrams and bigrams is patterned.

What do you conclude?

## Task 3: Authorship prediction example

For your final project, you are asked to engage in author attribution. After the data acquisition, you might decide to extract some linguistic features and build machine learning models.

One (of the various) approaches to do such a project is used in the quanteda tutorial on authorship prediction.

Try to replicate the example from https://quanteda.io/articles/pkgdown/replication/qss.html.

```{r}
# The replication is self-explanatory
```

## Task 4: Finding patterns in your own writing

Now use the mini-corpus of your own coursework that you prepared in the previous weeks. With that corpus, run unsupervised models to examine whether there are patterns in the content and style of your work.

```{r}
# You can follow the same steps as in task 1 and task 2.
```