Homework

Devin Etcitty

5/1/2017

```
library(tm)
```

```
## Loading required package: NLP
```

```
library(Matrix)
library(glmnet)
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-5
```

```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(tidyverse)
```

```
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ---------------------------------------------
```

```
## accumulate(): purrr, foreach
## annotate():   ggplot2, NLP
## expand():     tidyr, Matrix
## filter():     dplyr, stats
## lag():        dplyr, stats
## lift():       purrr, caret
## when():       purrr, foreach
```

```r
library(scales)
```

```
##
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
#setwd("~/columbia/APMA4990/msd-homework/homework/homework_3/problem_1")
setwd("~/Documents/Columbia/msd-apam4990/msd2017/homework/homework_3/problem_1")
```

read business and world articles into one data frame

```r
business <- read.table('business.tsv', quote="",header=TRUE, sep="\t", encoding= 'utf-8')
business <- business %>%
  mutate(section = 'business')
world <- read.table('world.tsv', quote="", header=TRUE, sep="\t", encoding = "utf-8")

world <- world %>%
  mutate(section = 'world')

paper_df <- bind_rows(business, world)
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character
```

```r
paper_df$section <- as.factor(paper_df$section)
```

create a Corpus from the article snippets

```r
corpus_paper <- DataframeSource(paper_df)

all_corpus <- Corpus(VectorSource(paper_df$snippet))
```

create a DocumentTermMatrix from the snippet Corpus

remove punctuation and numbers

```r
dtm.paper <- DocumentTermMatrix(all_corpus, control = list(removePunctuation = TRUE,
                                                            stopwords = TRUE))
```

convert the DocumentTermMatrix to a sparseMatrix, required by cv.glmnet

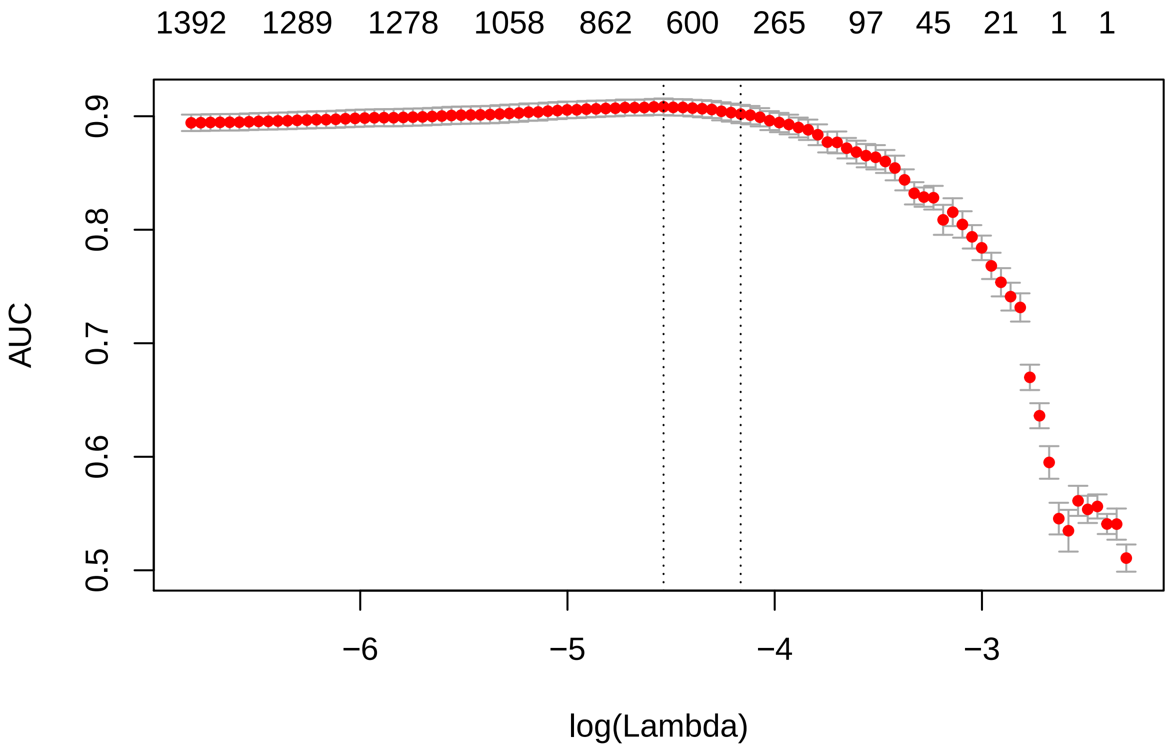
helper function

```
dtm_to_sparse <- function(dtm) {
 sparseMatrix(i=dtm$i, j=dtm$j, x=dtm$v, dims=c(dtm$nrow, dtm$ncol), dimnames=dtm$dimnames)
}

sparseMatrix.paper <- dtm_to_sparse(dtm.paper)
```

create a train / test split

```
set.seed(48)

train_percent <- 0.8

ndx_all <- sample(nrow(paper_df), floor(nrow(paper_df) * train_percent))

train_all <- paper_df[ndx_all, ]
test_all <- paper_df[-ndx_all,]

train_sparseMatrix <- sparseMatrix.paper[ndx_all, , drop=FALSE]
test_sparseMatrix <- sparseMatrix.paper[-ndx_all, , drop=FALSE]

fit <- cv.glmnet(train_sparseMatrix, train_all$section, family='binomial', type.measure = 'auc')

plot(fit, xvar = "dev", label = TRUE)
```

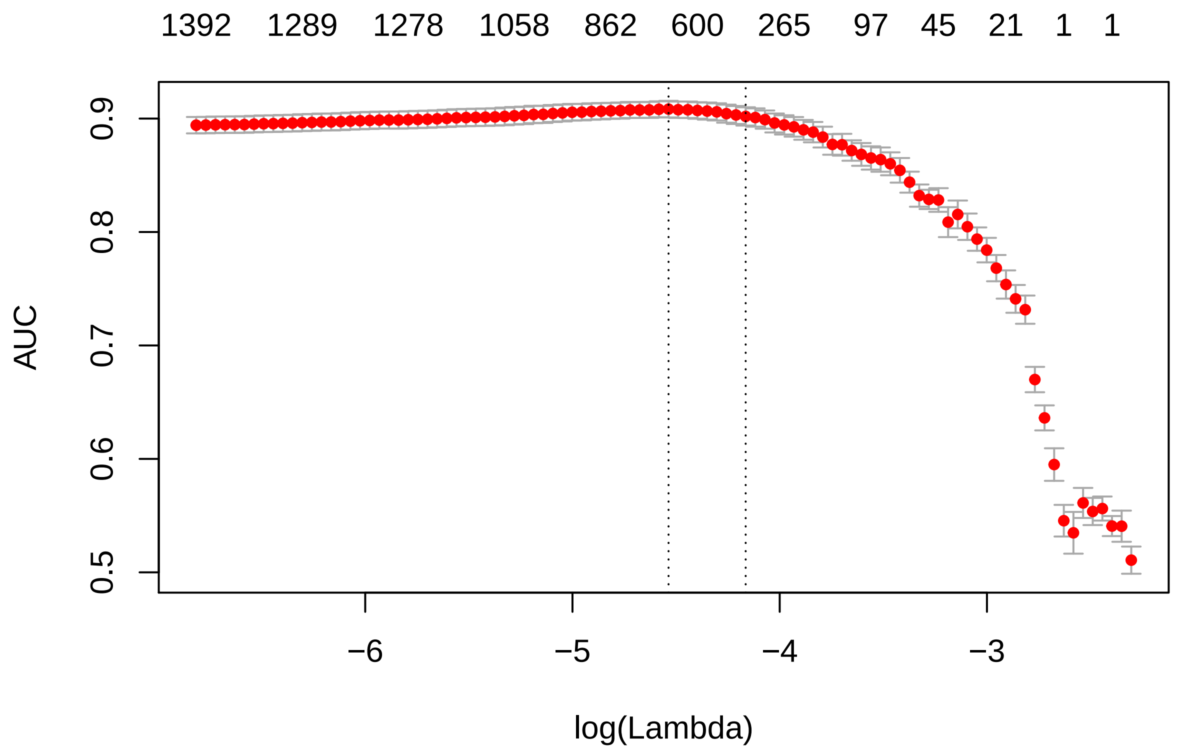
```
## Warning in plot.window(...): "xvar" is not a graphical parameter

## Warning in plot.window(...): "label" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "xvar" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "label" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "xvar" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "label" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "xvar" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "label" is not
## a graphical parameter

## Warning in box(...): "xvar" is not a graphical parameter

## Warning in box(...): "label" is not a graphical parameter

## Warning in title(...): "xvar" is not a graphical parameter
## Warning in title(...): "label" is not a graphical parameter
```



```
df <- data.frame(actual = test_all$section,
                 log_odds = predict(fit, test_sparseMatrix)) %>%

  mutate(pred = if_else(X1 > 0, 'world', 'business'))
```

evaluate performance for the best-fit model

```
plot(fit)
```



```
head(df)
```

```
##     actual         X1     pred
## 1 business -0.9763070 business
## 2 business -3.3673430 business
## 3 business -0.5455109 business
## 4 business -1.5792096 business
## 5 business -2.8015241 business
## 6 business -0.4732008 business
```

```
table(actual = df$actual, predicted = df$pred)
```

```
##           predicted
## actual     business world
##   business      149    51
##   world          30   170
```

accuracy: fraction of correct classifications

```
df %>%
  summarize(acc = mean(pred == actual))
```

```
##      acc
## 1 0.7975
```

precision: fraction of positive predictions that are actually true

```r
df %>%
  filter(pred == 'business') %>%
  summarize(prec = mean(actual == 'business'))
```

```
##        prec
## 1 0.8324022
```

recall: fraction of true examples that we predicted to be positive

aka true positive rate, sensitivity

```r
df %>%
  filter(actual == 'business') %>%
  summarize(recall = mean(pred == 'business'))
```

```
##   recall
## 1  0.745
```

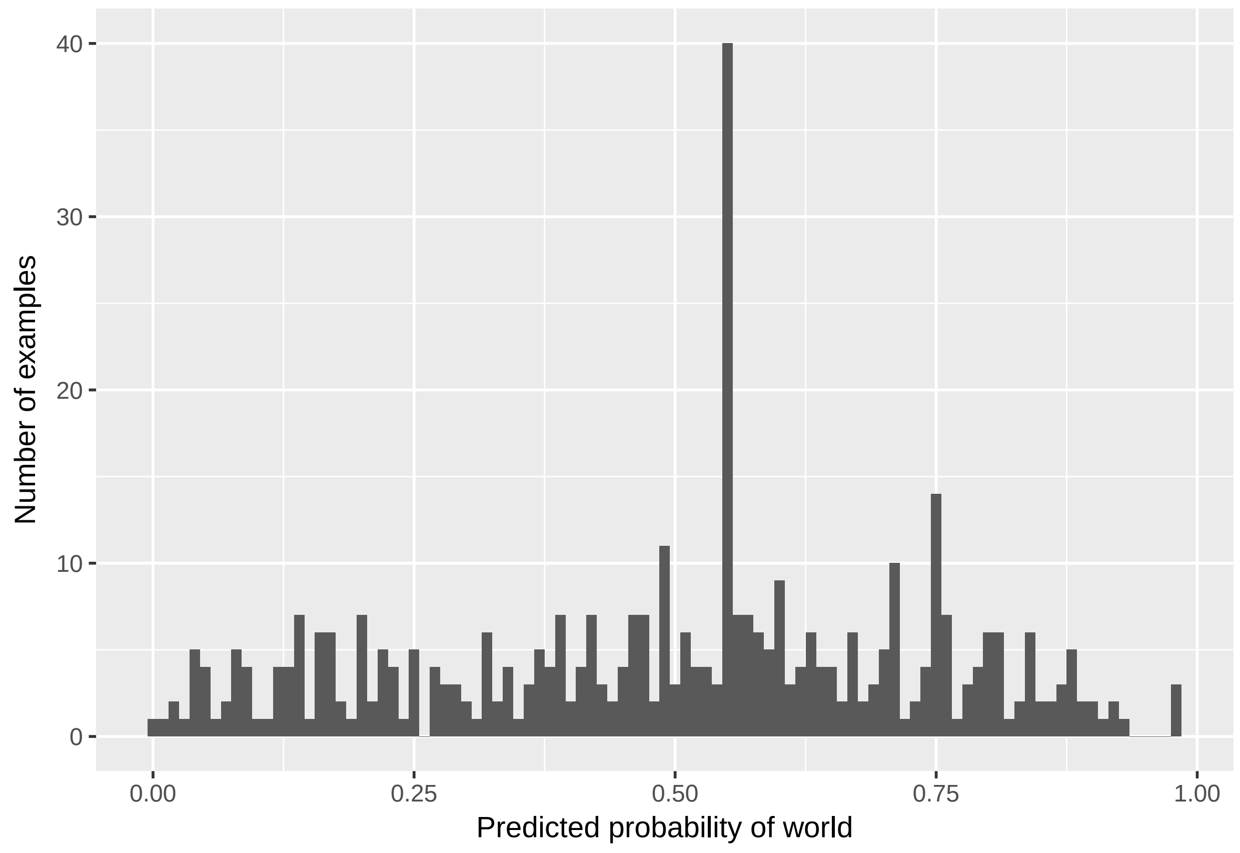
false positive rate: fraction of false examples that we predicted to be positive

```r
df %>%
  filter(actual == 'world') %>%
  summarize(fpr = mean(pred == 'business'))
```

```
##    fpr
## 1 0.15
```

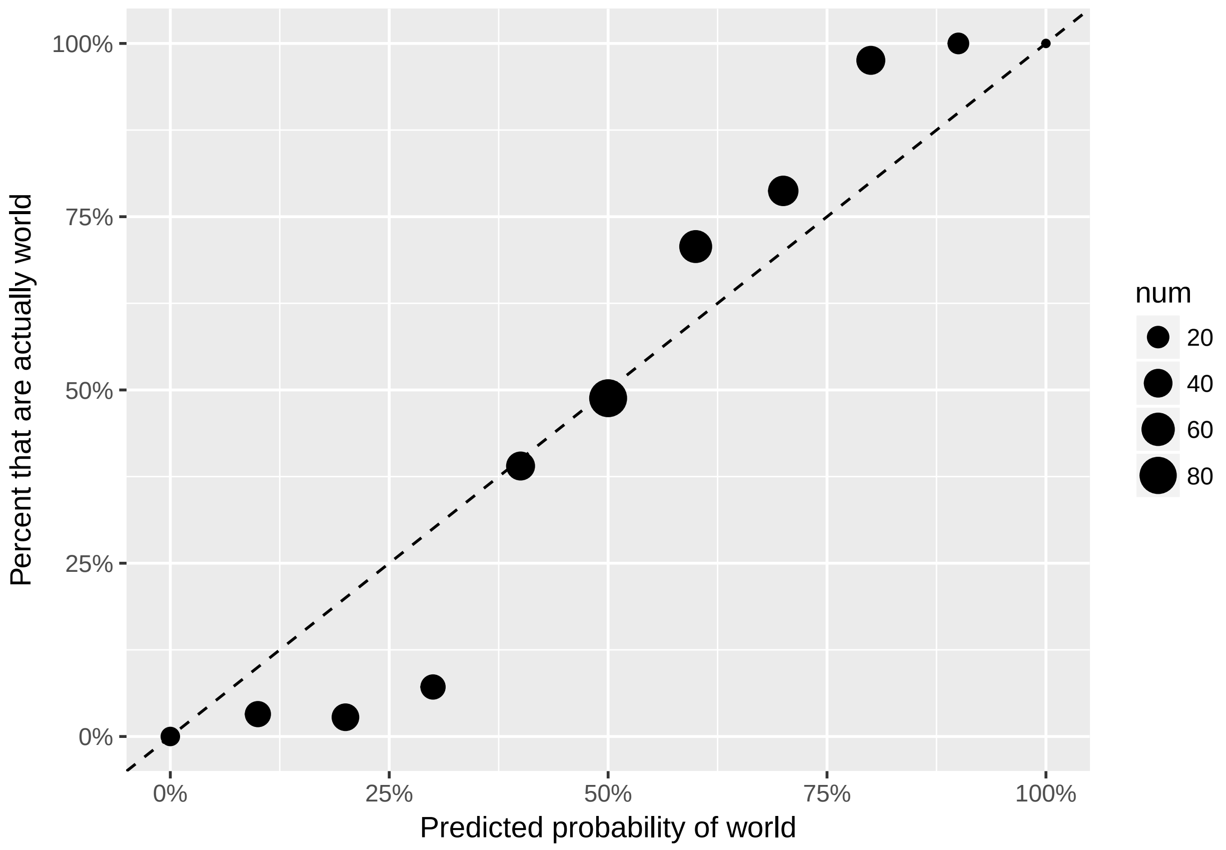
plot ROC curve and output accuracy and AUC

```r
plot_data = test_all
plot_data$probs <- predict(fit, test_sparseMatrix, type="response")
ggplot(plot_data, aes(x = probs)) +
  geom_histogram(binwidth = 0.01) +
  xlab('Predicted probability of world') +
  ylab('Number of examples')
```

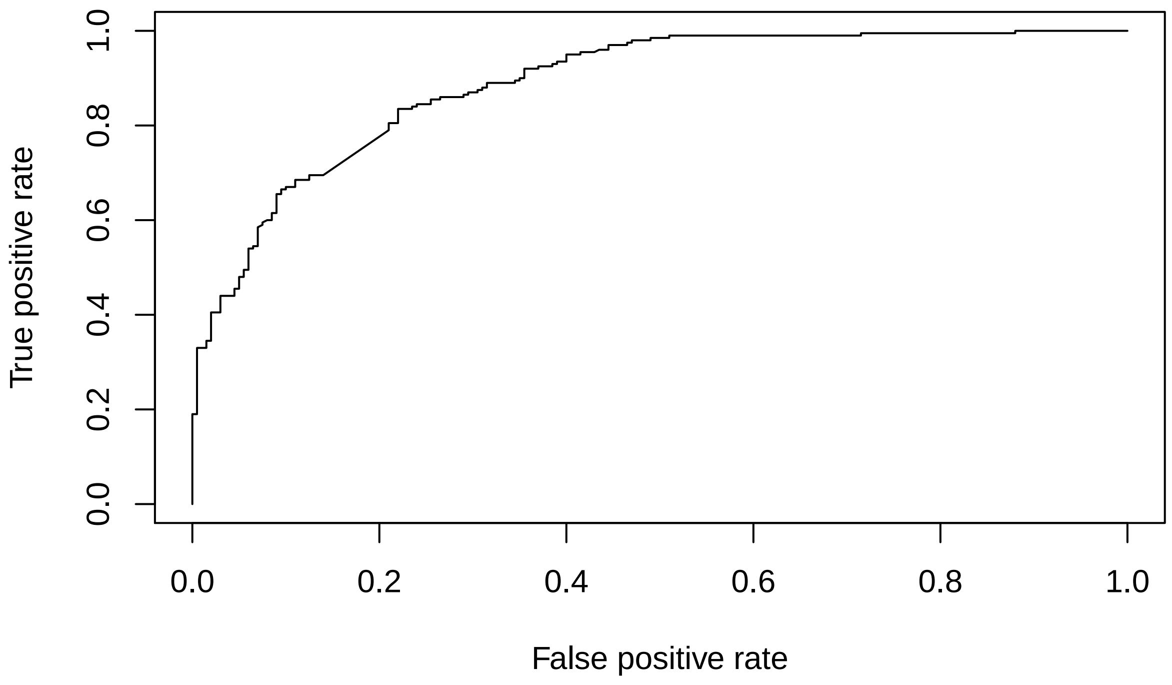


plot calibration

```r
data.frame(predicted=plot_data$probs, actual=test_all$section) %>%
  group_by(X1=round(X1*10)/10) %>%
  summarize(num=n(), actual=mean(actual == "world")) %>%
  ggplot(data=., aes(x=X1, y=actual, size=num)) +
  geom_point() +
  geom_abline(linetype=2) +
  scale_x_continuous(labels=percent, lim=c(0,1)) +
  scale_y_continuous(labels=percent, lim=c(0,1)) +
  xlab('Predicted probability of world') +
  ylab('Percent that are actually world')
```



```r
pred <- prediction(plot_data$probs, test_all$section)
perf_lr <- performance(pred, measure='tpr', x.measure='fpr')
plot(perf_lr)
```



```
performance(pred, 'auc')
```

```
## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.8874875
##
##
## Slot "alpha.values":
## list()
```

```
predicted <- plot_data$probs
actual <- test_all$section == "world"
ndx_pos <- sample(which(actual == 1), size=100, replace=T)
ndx_neg <- sample(which(actual == 0), size=100, replace=T)
mean(predicted[ndx_pos] > predicted[ndx_neg])
```

```
## [1] 0.89
```

extract coefficients for words with non-zero weight

helper function

```
get_informative_words <- function(crossval) {
  coefs <- coef(crossval, s="lambda.min")
  coefs <- as.data.frame(as.matrix(coefs))
  names(coefs) <- "weight"
  coefs$word <- row.names(coefs)
  row.names(coefs) <- NULL
  subset(coefs, weight != 0)
}
```

show weights on words with top 10 weights for business

```
weights <- get_informative_words(fit)

business_weights <- weights %>%
  arrange(weight) %>%
  top_n(-10, weight)

business_weights
```

```
##       weight       word
## 1  -2.314069        tax
## 2  -2.251331    company
## 3  -1.826375  company's
## 4  -1.761670        fox
## 5  -1.722934      thiel
## 6  -1.681047  financial
## 7  -1.666536  companies
## 8  -1.627890   business
## 9  -1.623296 california
## 10 -1.521687       bank
```

show weights on words with top 10 weights for world

```
world_weights <- weights %>%
  arrange(desc(weight)) %>%
  top_n(10, weight)

world_weights
```

```
##      weight       word
## 1  1.802957 competitor
## 2  1.605099     russia
## 3  1.496914     leader
## 4  1.420827    islands
## 5  1.320920     bribes
## 6  1.264667    killing
## 7  1.248337     merkel
## 8  1.229267 candidates
## 9  1.198614  democracy
## 10 1.172691     canada
```