A method for measuring the semantic value of words

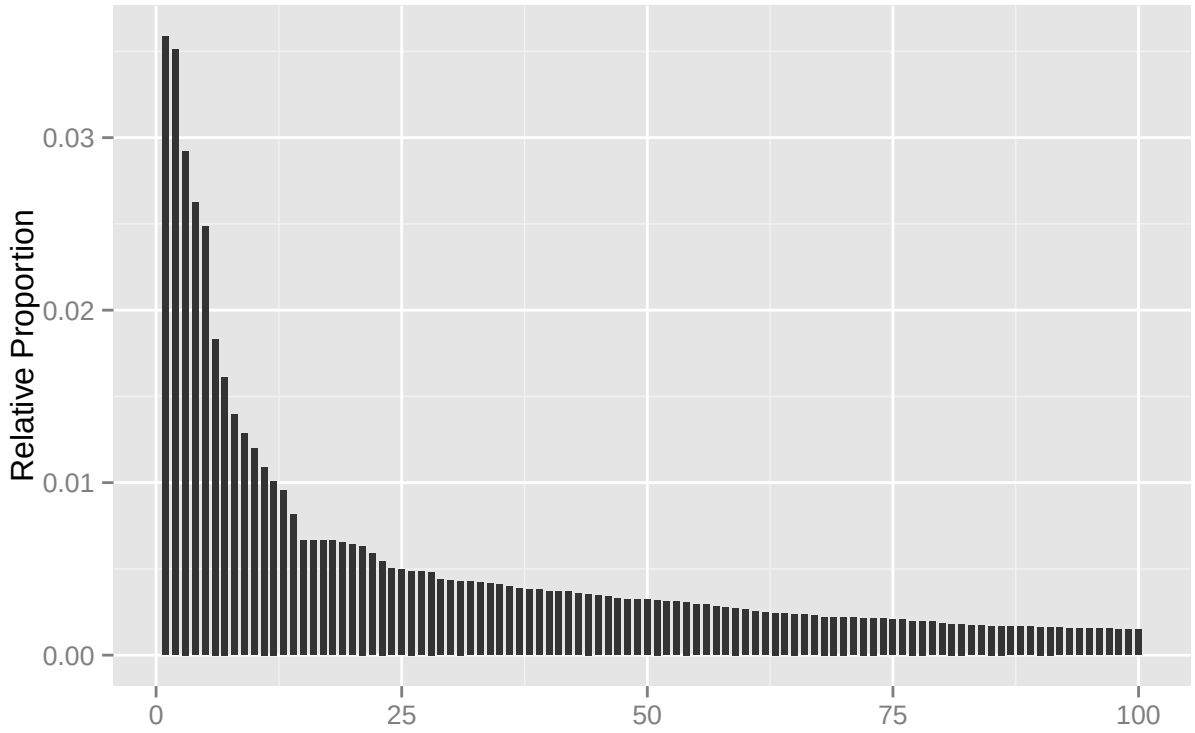
Dillon Niederhut

September 8, 2015

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:stats':
##
##     filter, lag
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Gram use follows Zipfian distribution

```
ggplot(data=top.uni, aes(x=rank,y=proportion)) +
  geom_bar(stat='identity', width=0.7) +
  xlab('') +
  ylab('Relative Proportion') +
  ggtitle('')
```



```
ggsave('output/fig1.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
top.uni[1:10, c('token','proportion')]
```

```
##    token proportion
## 1    and 0.03584499
## 2    the 0.03512560
## 3      i 0.02922695
## 4      a 0.02624904
## 5     to 0.02486015
## 6     it 0.01832433
## 7     of 0.01613009
## 8   that 0.01395472
## 9     in 0.01285227
## 10   you 0.01196060
```

Intuitively, words used less often are more informative

This is because they only apply to certain contexts

A single word can give you a lot of context, if it is used in a constrained manner

E.g. fedora

An informative word allows you to predict what other words will be used

'Context' is very hard to measure

We can substitute measuring context by looking at surrounding words

We use the change in the Zipfian distribution *given* that a word is in the utterance to quantify the extent to which that word informs you about the context

Changes in your ability to predict can be measured with chi-squared

Given that a word is used in an utterance:

the difference between the distribution of use of all words in a language, and the distribution of use of words in utterances that include that word will give you a measure of how informative that word is

we will call this the zipf-test

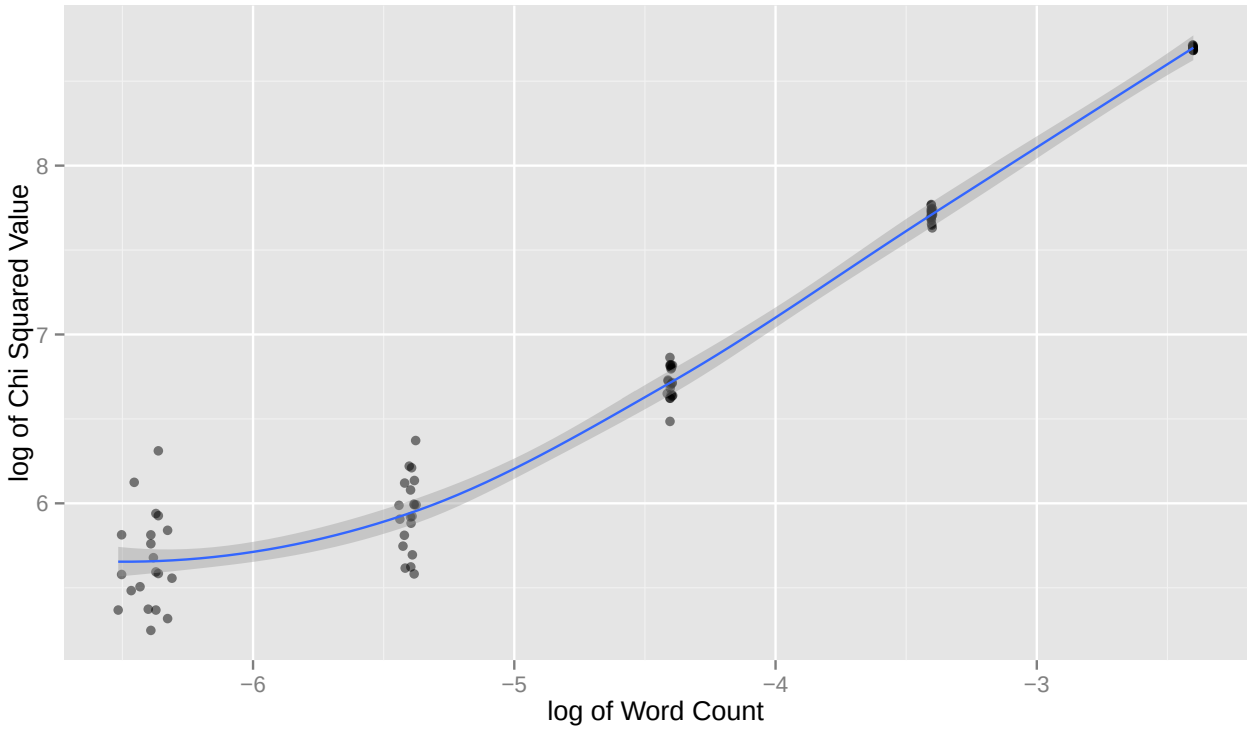
however, the test is additive, meaning that the overall numeric value of the test is dependent on the sum of words used, which is dependent on word frequency, which is what we are trying to measure

so we start with random sampling to find the sampling distribution of the zipf test

The sampling distribution of the Zipf-test

The distribution of randomly samples chi-squared values becomes linear when both variables are square-root transformed.

```
ggplot(data=control, aes(x=log10(tf), y=log10(chisq))) +
  geom_point(alpha=0.5) +
  stat_smooth(method='loess') +
  xlab('log of Word Count') +
  ylab('log of Chi Squared Value') +
  ggtitle('')
```

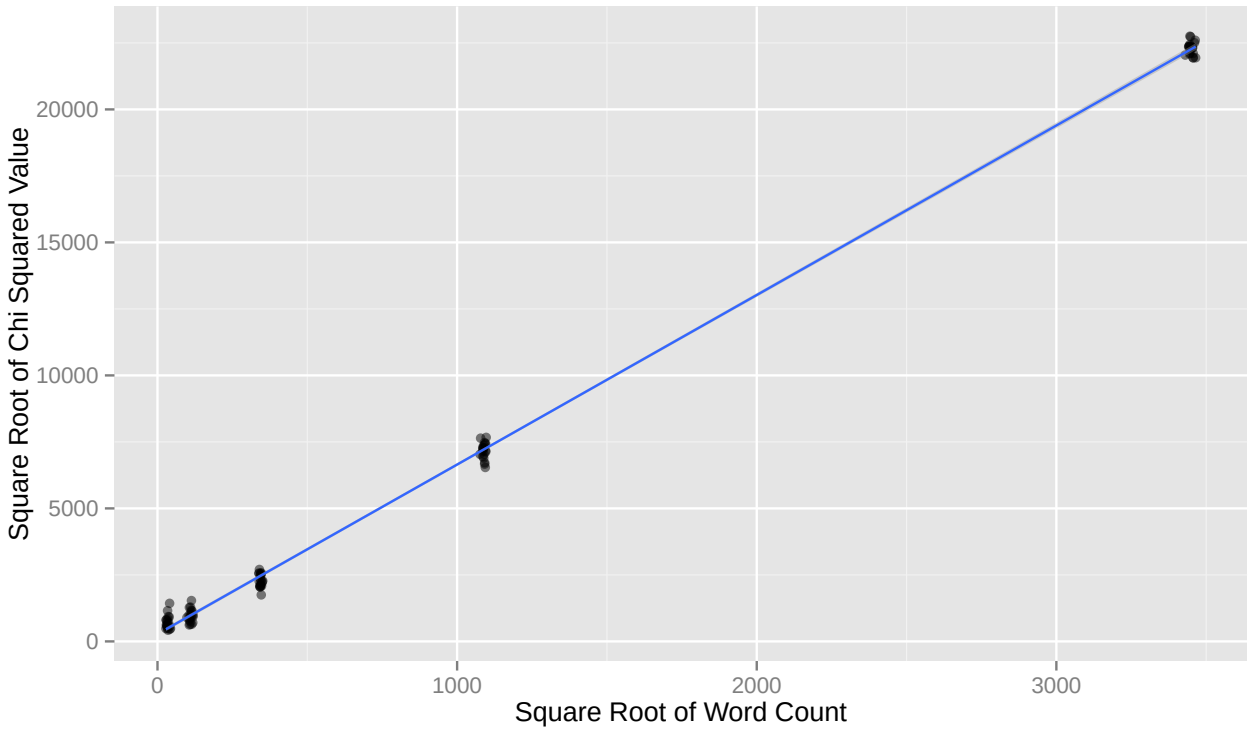



```
ggsave('output/fig2.png')
```

```
## Saving 8 x 5 in image
```

The square root of chi-squared values is almost perfectly predicted by the size (in number of grams) of each sample.

```
ggplot(data=control, aes(x=sqrt(counts), y=sqrt(chisq))) +
  geom_point(alpha=0.5) +
  stat_smooth(method='lm') +
  xlab('Square Root of Word Count') +
  ylab('Square Root of Chi Squared Value') +
  ggtitle('')
```




```
ggsave('output/fig3.png')
```

```
## Saving 8 x 5 in image
```

```
control$chisq.sqrt <- sqrt(control$chisq)
control$counts.sqrt <- sqrt(control$counts)
model.1 <- lm(chisq.sqrt ~ counts.sqrt, data = control)
summary(model.1)
```

```
##
## Call:
## lm(formula = chisq.sqrt ~ counts.sqrt, data = control)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -734.09 -175.63   27.78  156.13  897.72
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 274.85335   36.88229   7.452 3.67e-11 ***
## counts.sqrt   6.37361    0.02268 280.986  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 289.8 on 98 degrees of freedom
## Multiple R-squared:  0.9988, Adjusted R-squared:  0.9987
## F-statistic: 7.895e+04 on 1 and 98 DF,  p-value: < 2.2e-16
```

Additionally, the variance of the square root of the zipf test values is nearly constant across four orders of magnitude difference in the term frequency.

```
summarize(group_by(control, p), sd(chisq.sqrt))
```

```
## Source: local data frame [5 x 2]
##
##       p sd(chisq.sqrt)
## 1 1e-05       254.3478
## 2 1e-04       240.9732
## 3 1e-03       244.9472
## 4 1e-02       307.1144
## 5 1e-01       244.4642
```

```
table2 <- xtable(summarize(group_by(control, p), sd(chisq.sqrt)), digits=-2)
names(table2) <- c('Sampling probability', 'Std. Error')
print.xtable(table2, type='latex', file='output/table2.tex')
```

To create a test, the values are:

1. square root transformed; and
2. corrected for this linear relationship

```
control$c.chisq.sqrt <- control$chisq.sqrt - control$counts.sqrt * model.1$coefficients[[2]] - model.1$
control$z.chisq.sqrt <- (control$chisq.sqrt - control$counts.sqrt * model.1$coefficients[[2]] - model.1
data$chisq.sqrt <- sqrt(data$chisq)
data$counts.sqrt <- sqrt(data$counts)
data$c.chisq.sqrt <- data$chisq.sqrt - data$counts.sqrt * model.1$coefficients[[2]] - model.1$coefficie
data$z.chisq.sqrt <- (data$chisq.sqrt - data$counts.sqrt * model.1$coefficients[[2]] - model.1$coefficie
```

As a quick sanity check, we can look at the relative proportions and Zipf values of a spread of words

```
words <- c('my', 'day', 'feel', 'dowm', 'record', 'dill',
           'unclear', 'hither', 'omlette', 'multicollinear')
arrange(data[data$term %in% words, c('term', 'tf', 'z.chisq.sqrt')], z.chisq.sqrt)
```

```
##             term           tf z.chisq.sqrt
## 1             my 1.299537e-02  -94.1100352
## 2            day 2.115726e-03  -58.1961984
## 3           feel 1.711416e-03  -50.4174877
## 4         record 1.240189e-04   -7.1347174
## 5        unclear 1.108053e-05   -0.8122089
## 6         hither 6.094293e-07    0.6797409
## 7           dill 2.917874e-06    0.8254768
## 8        omlette 3.416498e-07    1.7794065
## 9 multicollinear 3.693511e-08    3.2798845
```

```
table3 <- xtable(arrange(data[data$term %in% words, c('term', 'tf', 'z.chisq.sqrt')], z.chisq.sqrt), di
names(table3) <- c('Term', 'Relative Proportion', 'Test Value')
print.xtable(table3, type='latex', file='output/table3.tex', include.rownames=FALSE)
```