realEstateDictionaries

Loading the data

The **listings** data set includes 24,446 listings from Atlanta, GA. The data set includes the price, square footage of living space, and public remarks about the property.

```
library(realEstateDictionaries)
data(listings)
str(listings)
```

```
## 'data.frame':    24446 obs. of  3 variables:
##  $ price  : num  106300 65000 255000 129900 180000 ...
##  $ sqft   : int  1424 724 3040 1583 2571 2059 1379 1870 1870 2769 ...
##  $ remarks: chr  "RARE 4 BEDROOM HOME WITH FRESH PAINT IN & OUT. NEW LIGHT CARPET AND VINYL
PLUS DUAL PANE WINDOWS. TAXES ARE INV"| __truncated__ "**********HOUSE IS VACANT -
ON LOCKBOX!!!!!!!********** **********OLDER "| __truncated__
"YOU'LL NEVER NEED MORE SPACE THAN THIS NEWER LIGHT & NEUTRAL EXECUTIVE HOME ON OVER 1/4 ACRE
WITH ULTRA SPACIOU"| __truncated__ "GREAT REMODEL WITH TONS OF FEATURES! KITCHEN REMODELED 3
YRS AGO ALONG WITH NEW HEAT/AC. ROOF 1 YR, WATERHEATER"| __truncated__ ...
```

Cleaning the Text

The **cleanText** function cleans the text by removing punctuation, numerical values, and stop words from the text. The raw text

```
text0 <- head(listings$remarks , n=3)
text0
```

```
## [1] "RARE 4 BEDROOM HOME WITH FRESH PAINT IN & OUT. NEW LIGHT CARPET AND VINYL PLUS DUAL
PANE WINDOWS. TAXES ARE INVESTORS. SELLER HAS NEVER OCCUPIED HOME."
## [2] "**********HOUSE IS VACANT - ON LOCKBOX!!!!!!!**********
**********OLDER HOME ON A HUGE LOT!!**********
**********SOLD IN 'AS IS' CONDITION!**********"
## [3] "YOU'LL NEVER NEED MORE SPACE THAN THIS NEWER LIGHT & NEUTRAL EXECUTIVE HOME ON OVER
1/4 ACRE WITH ULTRA SPACIOUS ROOMS THRUOUT, THAT RARE 4 CAR GARAGE AND REAL R/V PARKING WITH
CONCRETE SLAB FOR BOAT/RV AND EVEN A SPORT COURT FOR THE KIDS! 16' LIGHT STONE TILE IN ENTRY,
KITCHEN, LNDRY & DOWNSTAIRS BATH, 5TH BED IS DEN/ OFFICE WITH BUILT-IN CABINETRY, WOOD BLINDS,
WOOD WINDOW SILLS, STEREO & SURROUND PRE-WIRE, CENTRAL AUDIO PRE-WIRE, GARAGE SPEAKERS,
EXTENDED BACK PATIO,"
```

The cleaned text

```
text1 <- cleanText(text0)
text1
```

```
## [1] "rare bedroom home fresh paint new light carpet vinyl plus dual pane windows taxes
investors seller never occupied home"
## [2] "house vacant lockbox older home huge lot sold condition"
## [3] "ll never need space newer light neutral executive home acre ultra spacious rooms
thruout rare car garage real parking concrete slab boat rv even sport court kids light stone
tile entry kitchen lndry downstairs bath th bed den office built cabinetry wood blinds wood
window sills stereo surround pre wire central audio pre wire garage speakers extended back
patio"
```

Other options are available in the options of **cleanText**

```
text2 <- cleanText(text0 , removeStopWords = FALSE)
```

```
text2
```

```
## [1] "rare bedroom home with fresh paint in out new light carpet and vinyl plus dual pane
windows taxes are investors seller has never occupied home"
## [2] "house is vacant on lockbox older home on huge lot sold in as is condition"
## [3] "you ll never need more space than this newer light neutral executive home on over acre
with ultra spacious rooms thruout that rare car garage and real parking with concrete slab for
boat rv and even sport court for the kids light stone tile in entry kitchen lndry downstairs
bath th bed is den office with built in cabinetry wood blinds wood window sills stereo
surround pre wire central audio pre wire garage speakers extended back patio"
```

Creating a Token Matrix

You can also create a matrix M of indicator variables for the n-gram tokens. A token is a sequence of n consecutive phrases. The $n = 1$ or unigram corresponds to single words. The $n = 2$ or bigram corresponds to two-word phrases. **tokenMatrixMaker** can handle up to $n = 3$ or trigrams. You can specify n using **GRAM**. The default is 1-grams.

```
M1 <- tokenMatrixMaker(text1)
head(M1)
```

```
## 3 x 73 sparse Matrix of class "ngCMatrix"
##
## [1,] | | | | | | | | | | | | | | | | | . . . . . .
## [2,] | . . . . . | | | | | | | . . . . . .
## [3,] | | | | . . . . . | | | | | | | |
##
## [1,] . . . . . .
## [2,] . . . . . .
## [3,] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
##
## [1,] . . .
## [2,] . . .
## [3,] | | |
```

```
colnames(M1)
```

```
##  [1] "home"       "light"      "never"      "rare"       "bedroom"
##  [6] "carpet"     "dual"       "fresh"      "investors"  "new"
## [11] "occupied"   "paint"      "pane"       "plus"       "seller"
## [16] "taxes"      "vinyl"      "windows"    "condition"  "house"
## [21] "huge"       "lockbox"    "lot"        "older"      "sold"
## [26] "vacant"     "acre"       "audio"      "back"       "bath"
## [31] "bed"        "blinds"     "boat"       "built"      "cabinetry"
## [36] "car"        "central"    "concrete"   "court"      "den"
## [41] "downstairs" "entry"      "even"       "executive"  "extended"
## [46] "garage"     "kids"       "kitchen"    "lndry"      "need"
## [51] "neutral"    "newer"      "office"     "parking"    "patio"
## [56] "pre"        "real"       "rooms"      "sills"      "slab"
## [61] "space"      "spacious"   "speakers"   "sport"      "stereo"
## [66] "stone"      "surround"   "thruout"    "tile"       "ultra"
## [71] "window"     "wire"       "wood"
```

The columns of M are sorted from most frequent token to least frequent token. You can specify how many columns (K) to keep in M using **KTOKEN**. The default is $K = 500$.

```
M1 <- tokenMatrixMaker(text1 , GRAM=2 , KTOKEN=10)
head(M1)
```

```
## 3 x 10 sparse Matrix of class "ngCMatrix"
```

```
##
## [1,] | | | | | | | | | |
## [2,] . . . . . .
## [3,] . . . . . .
```

```
colnames(M1)
```

```
##  [1] "bedroom-home"     "carpet-vinyl"     "dual-pane"
##  [4] "fresh-paint"      "home-fresh"       "investors-seller"
##  [7] "light-carpet"     "never-occupied"   "new-light"
## [10] "occupied-home"
```

Identifying the Dictionary

You can identify a dictionary $S \subseteq \{1, \ldots, K\}$ using LASSO methods. You can also include other explanatory variables alongisde the tokens (square footage, age, date of sale, etc.).

An example

```
X <- as.matrix(cbind(listings$sqft,listings$sqft**2))
text0 <- listings$remarks
text1 <- cleanText(text0)
M <- tokenMatrixMaker(text1 , GRAM=1 , KTOKEN=500)
y <- log(listings$price)
fit <- lassoPostLasso(X,M,y)
```

lassoPostLasso includes prediction information

```
fit$predictionInformation
```

```
##       N   K Q_min Q_het rmse_baseline  rmse_min  rmse_het mae_baseline
## 1 24446 500   180   106     0.2822435 0.2430242 0.2469402     0.186479
##     mae_min   mae_het
## 1 0.1528359 0.1558884
```

fit includes the tokens selected by both LASSO procedures

```
head(fit$cvDictionary)
```

```
##        token estimate
## 1       acre    0.091
## 2      added   -0.008
## 3 appliances    0.014
## 4    arizona   -0.026
## 5   backyard    0.007
## 6    balcony   -0.049
```

```
head(fit$hetDictionary)
```

```
##        token estimate
## 1       acre    0.105
## 2 appliances    0.024
## 3        bbq    0.043
## 4       best    0.047
## 5        big   -0.030
## 6      block   -0.032
```

fit includes the predicted values for both LASSO procedures

```
head(fit$fittedValues)
```

```
##   original hetFitted cvFitted
## 1 11.57402  11.63285 11.61822
## 2 11.08214  11.23268 11.20849
## 3 12.44902  12.56801 12.63996
## 4 11.77452  11.49771 11.46436
## 5 12.10071  12.44141 12.42522
## 6 12.01370  12.07475 12.02610
```

realEstateDictionary

The function **realEstateDictionary** is a wrapper for the above procedures. Simply provide i) a data frame, ii) columns to use in the X matrix, iii) the name of the column that contains the text, iv) the name of the column that includes the dependent variable

```
listings$sqft2 <- listings$sqft**2
listings$logprice <- log(listings$price)
fitWrapper <- realEstateDictionary(XVARS=c("sqft","sqft2"),
                       TEXTVAR="remarks",
                       YVAR="logprice",
                       DATA=listings)
head(fitWrapper$hetDictionary)
```

```
##        token estimate
## 1       acre    0.105
## 2 appliances    0.024
## 3        bbq    0.043
## 4       best    0.047
## 5        big   -0.030
## 6      block   -0.032
```