Automating Data Exploration with R

Text

We need to find clever ways of turning text data into numbers. You can choose to ignore any text and just model off the numerical variables but you would be leaving a lot of intelligence on the table.

At a high level, there are two types of text data, free-form text and categorical text Free-form text is complicated and requires a lot more custom work to extract quantitative value out of it - though beyond the scope of this course, we'll look at some simple ways of capturing value out of them. Categorical data, on the other hand, is fair game and there are many ways to translate that into quantitative variables.

Free-Form Text

So, what about regular text? Without delving into natural language processing (NLP), there are a few easy things we can do on free-form text before discarding it. This comes down to feature engineering:

- Number of words
- Character count
- First word

Let's run through a few code snippets on how to extract these basic quantitative measures and then build a function to handle this automatically for our pipeline.

Number of words

Let's gather the number of words from the `Name` field of the Titanic data set:

```
# load the data set in case you haven't already done so
Titanic_dataset <- read.table('http://math.ucdenver.edu/RTutorial/titanic.txt', se
p='\t', header=TRUE, stringsAsFactors = FALSE)
head(Titanic_dataset)
```

```
##                                                 Name PClass   Age    Sex
## 1                    Allen, Miss Elisabeth Walton    1st 29.00 female
## 2                     Allison, Miss Helen Loraine    1st  2.00 female
## 3             Allison, Mr Hudson Joshua Creighton    1st 30.00   male
## 4 Allison, Mrs Hudson JC (Bessie Waldo Daniels)    1st 25.00 female
## 5                   Allison, Master Hudson Trevor    1st  0.92   male
## 6                              Anderson, Mr Harry    1st 47.00   male
##   Survived
## 1        1
## 2        0
## 3        0
## 4        0
## 5        1
## 6        1
```

```
Titanic_dataset_temp <- Titanic_dataset
Titanic_dataset_temp$Word_Count <- sapply(strsplit(Titanic_dataset_temp$Name, "
"), length)
print(head(Titanic_dataset_temp$Word_Count))
```

```
## [1] 4 4 5 7 4 3
```

Character count

Let's count the number of characters in the `Name` field of the Titanic data set:

```
Titanic_dataset_temp <- Titanic_dataset
Titanic_dataset_temp$Character_Count <-  nchar(as.character(Titanic_dataset_temp$N
ame))
print(head(Titanic_dataset_temp$Character_Count))
```

```
## [1] 28 27 35 45 29 18
```

First word

Finally, let's get the first word and treat it as a categorical variable:

```
Titanic_dataset_temp <- Titanic_dataset
Titanic_dataset_temp$First_Word <- sapply(strsplit(as.character(Titanic_dataset_te
mp$Name), " "), `[`, 1)
print(head(Titanic_dataset_temp$First_Word))
```

```
## [1] "Allen,"    "Allison,"  "Allison,"  "Allison,"  "Allison,"  "Anderson,"
```

This is just a small selection of possibilities. In the case of the `Name` field in the Titanic data set, pulling out the title and first name of the person would be worth a try. I am not adding it to our pipeline function as these are unique cases that will vary from data set to data set.

We need to make an important assumption here, what differentiates categorical text from free-form text? On the surface, the both appear as text entries, so here, I am considering data with more than 90% uniqueness as free-form text, and anything less, as categorical data. This is definitely one of the parameters that will need to be clearly exposed in your pipeline as it may require a lot of experimentation.

Let's build our function:

```
Get_Free_Text_Measures <- function(data_set, minimum_unique_threshold=0.9, feature
s_to_ignore=c()) {

     # look for text entries that are mostly unique
     text_features <- c(names(data_set[sapply(data_set, is.character)]), names(dat
a_set[sapply(data_set, is.factor)]))
     for (f_name in setdiff(text_features, features_to_ignore)) {
          f_vector <- as.character(data_set[,f_name])

          # treat as raw text if data over minimum_precent_unique unique
          if (length(unique(as.character(f_vector))) > (nrow(data_set) * minimum_u
nique_threshold)) {
               data_set[,paste0(f_name, '_word_count')] <- sapply(strsplit(f_vecto
r, " "), length)
               data_set[,paste0(f_name, '_character_count')]  <-  nchar(as.charact
er(f_vector))
               data_set[,paste0(f_name, '_first_word')] <- sapply(strsplit(as.char
acter(f_vector), " "), `[`, 1)
               # remove orginal field
               data_set[,f_name] <- NULL
          }
     }
     return(data_set)
}

Titanic_dataset_temp <- Get_Free_Text_Measures(data_set = Titanic_dataset, feature
s_to_ignore = c())
str(Titanic_dataset_temp)
```

```
## 'data.frame':    1313 obs. of  7 variables:
##  $ PClass              : chr  "1st" "1st" "1st" "1st" ...
##  $ Age                 : num  29 2 30 25 0.92 47 63 39 58 71 ...
##  $ Sex                 : chr  "female" "female" "male" "female" ...
##  $ Survived            : int  1 0 0 0 1 1 1 0 1 0 ...
##  $ Name_word_count     : int  4 4 5 7 4 3 4 4 6 3 ...
##  $ Name_character_count: int  28 27 35 45 29 18 32 22 44 22 ...
##  $ Name_first_word     : chr  "Allen," "Allison," "Allison," "Allison," ...
```