Pen-Based Recognition of Handwritten Digits

Leopold Hillah

Tuesday, September 30th, 2014

DATA SOURCE

This training data set was made available by the UCI Machine Learning Repository at the folowing location: http://archive.ics.uci.edu/ml/machine-learning-databases/pendigits/pendigits.tra. The data is stored in csv format.

ANALYSIS

1. DATA LOADING & PRE-PROCESSING

The data load assumes that the csv file is downloaded and stored in the current working directory, whose value can be obtained by the `getwd()` R command. The data is then loaded into the dfmovies data frame using the `read.csv` R command.

Setting global options

```
# set global chunk options: images will be 24x10 inches
knitr::opts_chunk$set(cache=TRUE, echo=TRUE, message=FALSE, fig.width=24, fig.height=10)
```

Setting up the R environment

```
# Clearing the cache
rm(list = ls())

# Loading required libraries
if ((!require(ggplot2)) | (!require(dplyr)) | (!require(reshape2))) install.packages('ggplot2', 'dplyr'
```

Loading and preprocessing the data

```
# Set working directoy here for csv file loading
filepath <- getwd()

# Load csv data data frames

dfpen <- read.csv(paste(filepath, "pendigits.tra", sep="/"), header = FALSE, quote="", comment="", stri
str(dfpen)
```

```
## 'data.frame':    7494 obs. of  17 variables:
##  $ V1 : int  47 0 0 0 0 100 0 0 13 57 ...
##  $ V2 : int  100 89 57 100 67 100 100 39 89 100 ...
##  $ V3 : int  27 27 31 7 49 88 3 2 12 22 ...
##  $ V4 : int  81 100 68 92 83 99 72 62 50 72 ...
##  $ V5 : int  57 42 72 5 100 49 26 11 72 0 ...
##  $ V6 : int  37 75 90 68 100 74 35 5 38 31 ...
##  $ V7 : int  26 29 100 19 81 17 85 63 56 25 ...
##  $ V8 : int  0 45 100 45 80 47 35 0 0 0 ...
##  $ V9 : int  0 15 76 86 60 0 100 100 4 75 ...
##  $ V10: int  23 15 75 34 60 16 71 43 17 13 ...
##  $ V11: int  56 37 50 100 40 37 73 89 0 100 ...
##  $ V12: int  53 0 51 45 40 0 97 99 61 50 ...
##  $ V13: int  100 69 28 74 33 73 65 36 32 75 ...
##  $ V14: int  90 2 25 23 20 16 49 100 94 87 ...
##  $ V15: int  40 100 16 67 47 20 66 0 100 26 ...
##  $ V16: int  98 6 0 0 0 20 0 57 100 85 ...
##  $ V17: int  8 2 1 4 1 6 4 0 5 0 ...
```

```r
head(dfpen)
```

```
##    V1  V2 V3  V4  V5  V6  V7  V8 V9 V10 V11 V12 V13 V14 V15 V16 V17
## 1  47 100 27  81  57  37  26   0  0  23  56  53 100  90  40  98   8
## 2   0  89 27 100  42  75  29  45 15  15  37   0  69   2 100   6   2
## 3   0  57 31  68  72  90 100 100 76  75  50  51  28  25  16   0   1
## 4   0 100  7  92   5  68  19  45 86  34 100  45  74  23  67   0   4
## 5   0  67 49  83 100 100  81  80 60  60  40  40  33  20  47   0   1
## 6 100 100 88  99  49  74  17  47  0  16  37   0  73  16  20  20   6
```

Once the data is loaded, an initial exploration shows that the first 16 variables are input features and the 17th variable is the class.

All variables are integer variables.

null values per column

We can look at the number of null values in each column of the data set:

```r
dfnulls <- colSums(is.na(dfpen))
print(dfnulls)
```

```
##  V1  V2  V3  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 V17
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

There is no null values in the data set.

Graph class distribution

We can next look at the distribution of class values among the entire data set using a histogram:

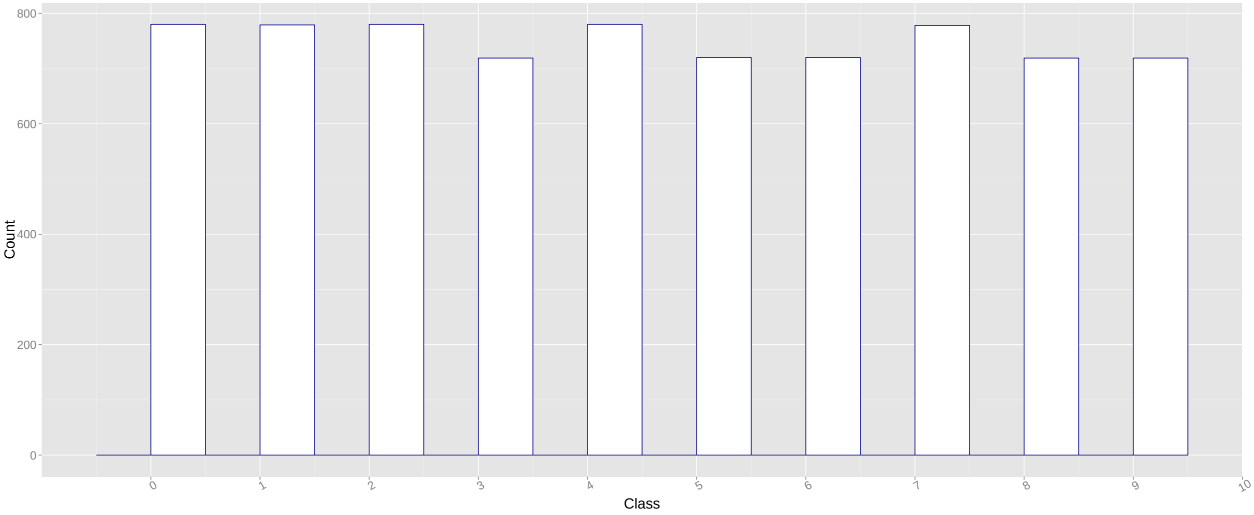
```r
# Graph class distribution

dfclass <- summarise (group_by(dfpen, V17), count = n())
print(dfclass)
```

```
## Source: local data frame [10 x 2]
##
##    V17 count
## 1    0   780
## 2    1   779
## 3    2   780
## 4    3   719
## 5    4   780
## 6    5   720
## 7    6   720
## 8    7   778
## 9    8   719
## 10   9   719
```

```r
p1 <- ggplot(dfpen, aes(x=V17)) +
  geom_histogram(binwidth=0.5, colour="darkblue", fill="white")+
  xlab("Class")+
  ylab("Count")+
  scale_x_continuous(breaks=seq(0, 17, 1))+
  theme(text = element_text(size=20),axis.text.x = element_text(angle=30, vjust=1))


print(p1)
```



Class value distribution are almost uniform.

Correlation among variables

We next look at the correlation among variables (except class or V17):

```
# Correlation among variables

dfcor <- matrix(cor(dfpen[,-17]), ncol=16)
str(dfcor)
```

```
##  num [1:16, 1:16] 1 0.341 0.263 0.116 -0.45 ...
```

```
# Set diagonals to 0
diag(dfcor) <- 0

for (row in 1:16){
        for (col in 1:16) {
                if (row < col & abs(dfcor[row, col]) > 0.7) print(paste(row, col, dfcor[row, col], sep=
        }
                }
```

```
## [1] "4:12:-0.727397842507734"
## [1] "6:8:0.775769520173762"
## [1] "6:14:-0.792337348660727"
## [1] "8:14:-0.709324383970509"
## [1] "14:16:0.857142926283263"
```

Highly correlated variables (70% or more) are : – V4 and V12, – V6 and V8, – V6 and V14, – V8 and V14, – V14 and V16