ST340 Programming for Data Science - Assignment 3

Matthew Nunes & Oliver Harvey

Friday 8 December 2017

Student number(s): 1508437 &

Student name(s): Matthew Nunes, Oliver Harvey

Q1

Here is a function that does gradient descent to find local minima:

```
gradient.descent <- function(f, df, x0, iterations=1000, alpha=0.2) {
  x<-x0
  for (i in 1:iterations) {
    cat(i,"/",iterations,": ",x," ",f(x),"\n")
    x<-x-alpha*df(x)
  }
  return(x)
}
```

Example:

```
f <-function(x) { sum(x^2) }
df<-function(x) { 2*x }
gradient.descent(f,df,c(10,20),10,0.2)
```

Q1a

Write a *short* function that uses `gradient.descent` to find a local *maxima*. (For the purpose of this question, `gradient.descent` is a "black box". Don't worry about the printed output, just the return value matters.)

i.e.

```
gradient.ascent <- function(f, df, x0, iterations=1000, alpha=0.2) {
... use gradient.descent(...) here ...
}
f <-function(x) { (1+x^2)^(-1) }
df<-function(x) { -2*x*(1+x^2)^(-2) }
gradient.ascent(f,df,3,40,0.5)
```

Q1b

Consider the function $f : R^2 \to R$

```
f <- function(x) (x[1]-1)^2 + 100*(x[1]^2-x[2])^2
```

(1) Give a short mathematical proof that f has a a unique minima.
(2) Write a function df to calculate the gradient of f, i.e.
`df <- function(x) { ... use x[1] and x[2] ...}`

(3) Starting from the point `x0=c(3,4)`, try to find the minimum using gradient descent. `gradient.descent(f,df,c(3,4), ... , ...)`

Q1c

Write a function to do gradient descent with momentum. Starting from the point `x0=c(3,4)`, use your function to the minimum of the function from part b.

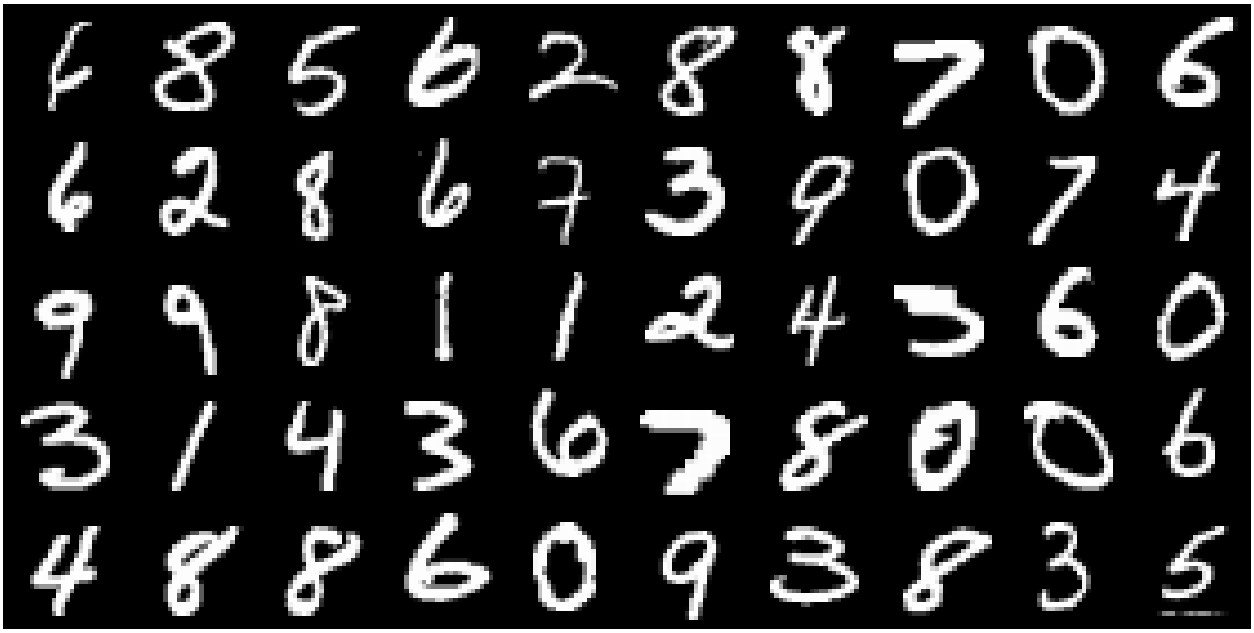
Q2

Load the tiny MNIST dataset:

```
load("mnist.tiny.RData")
train.X=train.X/255
test.X=test.X/255
```

show some digits:

```
library(grid)
grid.raster(  array(aperm(array(train.X[1:50,],c(5,10,28,28)),c(4,1,3,2)),c(140,280)),
              interpolate=F)
```



Use 3-fold cross validation on the training set to compare SVMs with linear kernels, polynomial kernels and RBF kernels. i.e.

```
library(e1071)
svm(train.X,train.labels,type="C-classification",kernel="linear",cross=3)$tot.accuracy
```

```
## Warning in svm.default(train.X, train.labels, type = "C-classification", :
## Variable(s) 'X1' and 'X2' and 'X3' and 'X4' and 'X5' and 'X6' and 'X7' and
## 'X8' and 'X9' and 'X10' and 'X11' and 'X12' and 'X13' and 'X14' and 'X15'
## and 'X16' and 'X17' and 'X18' and 'X19' and 'X20' and 'X21' and 'X22' and
## 'X23' and 'X24' and 'X25' and 'X26' and 'X27' and 'X28' and 'X29' and 'X30'
## and 'X31' and 'X32' and 'X33' and 'X34' and 'X35' and 'X36' and 'X37' and
```

```
## 'X38' and 'X42' and 'X43' and 'X44' and 'X45' and 'X46' and 'X47' and 'X51'
## and 'X52' and 'X53' and 'X54' and 'X55' and 'X56' and 'X57' and 'X58' and
## 'X59' and 'X60' and 'X61' and 'X62' and 'X80' and 'X81' and 'X82' and 'X83'
## and 'X84' and 'X85' and 'X86' and 'X87' and 'X88' and 'X89' and 'X110' and
## 'X111' and 'X112' and 'X113' and 'X114' and 'X115' and 'X116' and 'X138'
## and 'X139' and 'X140' and 'X141' and 'X142' and 'X143' and 'X168' and
## 'X169' and 'X170' and 'X171' and 'X196' and 'X197' and 'X224' and 'X225'
## and 'X252' and 'X253' and 'X280' and 'X281' and 'X308' and 'X309' and
## 'X335' and 'X336' and 'X337' and 'X338' and 'X364' and 'X365' and 'X366'
## and 'X367' and 'X392' and 'X393' and 'X394' and 'X395' and 'X420' and
## 'X421' and 'X422' and 'X423' and 'X448' and 'X449' and 'X450' and 'X451'
## and 'X476' and 'X477' and 'X478' and 'X479' and 'X504' and 'X505' and
## 'X506' and 'X533' and 'X534' and 'X561' and 'X562' and 'X588' and 'X589'
## and 'X590' and 'X616' and 'X617' and 'X618' and 'X642' and 'X643' and
## 'X644' and 'X645' and 'X646' and 'X670' and 'X671' and 'X672' and 'X673'
## and 'X674' and 'X675' and 'X698' and 'X699' and 'X700' and 'X701' and
## 'X702' and 'X703' and 'X725' and 'X726' and 'X727' and 'X728' and 'X729'
## and 'X730' and 'X731' and 'X732' and 'X733' and 'X734' and 'X753' and
## 'X754' and 'X755' and 'X756' and 'X757' and 'X758' and 'X759' and 'X760'
## and 'X761' and 'X762' and 'X766' and 'X767' and 'X768' and 'X776' and
## 'X777' and 'X781' and 'X782' and 'X783' and 'X784' constant. Cannot scale
## data.
```

```
## [1] 85.6
```

```
svm(train.X,train.labels,type="C-classification",kernel="poly",
    degree=2,coef=1,cross=3)$tot.accuracy
```

```
## Warning in svm.default(train.X, train.labels, type = "C-classification", :
## Variable(s) 'X1' and 'X2' and 'X3' and 'X4' and 'X5' and 'X6' and 'X7' and
## 'X8' and 'X9' and 'X10' and 'X11' and 'X12' and 'X13' and 'X14' and 'X15'
## and 'X16' and 'X17' and 'X18' and 'X19' and 'X20' and 'X21' and 'X22' and
## 'X23' and 'X24' and 'X25' and 'X26' and 'X27' and 'X28' and 'X29' and 'X30'
## and 'X31' and 'X32' and 'X33' and 'X34' and 'X35' and 'X36' and 'X37' and
## 'X38' and 'X42' and 'X43' and 'X44' and 'X45' and 'X46' and 'X47' and 'X51'
## and 'X52' and 'X53' and 'X54' and 'X55' and 'X56' and 'X57' and 'X58' and
## 'X59' and 'X60' and 'X61' and 'X62' and 'X80' and 'X81' and 'X82' and 'X83'
## and 'X84' and 'X85' and 'X86' and 'X87' and 'X88' and 'X89' and 'X110' and
## 'X111' and 'X112' and 'X113' and 'X114' and 'X115' and 'X116' and 'X138'
## and 'X139' and 'X140' and 'X141' and 'X142' and 'X143' and 'X168' and
## 'X169' and 'X170' and 'X171' and 'X196' and 'X197' and 'X224' and 'X225'
## and 'X252' and 'X253' and 'X280' and 'X281' and 'X308' and 'X309' and
## 'X335' and 'X336' and 'X337' and 'X338' and 'X364' and 'X365' and 'X366'
## and 'X367' and 'X392' and 'X393' and 'X394' and 'X395' and 'X420' and
## 'X421' and 'X422' and 'X423' and 'X448' and 'X449' and 'X450' and 'X451'
## and 'X476' and 'X477' and 'X478' and 'X479' and 'X504' and 'X505' and
## 'X506' and 'X533' and 'X534' and 'X561' and 'X562' and 'X588' and 'X589'
## and 'X590' and 'X616' and 'X617' and 'X618' and 'X642' and 'X643' and
## 'X644' and 'X645' and 'X646' and 'X670' and 'X671' and 'X672' and 'X673'
## and 'X674' and 'X675' and 'X698' and 'X699' and 'X700' and 'X701' and
## 'X702' and 'X703' and 'X725' and 'X726' and 'X727' and 'X728' and 'X729'
## and 'X730' and 'X731' and 'X732' and 'X733' and 'X734' and 'X753' and
## 'X754' and 'X755' and 'X756' and 'X757' and 'X758' and 'X759' and 'X760'
## and 'X761' and 'X762' and 'X766' and 'X767' and 'X768' and 'X776' and
## 'X777' and 'X781' and 'X782' and 'X783' and 'X784' constant. Cannot scale
```

```
## data.
```

```
## [1] 79.9
```

etc, etc.

For the RBF kernels, write a grid search function that takes two lists, `log.C.range` and `log.gamma.range`, and for each pair (lc,lg) of entries in the pair of lists attempts cross-validation with parameters `cost = exp(lc)` and `gamma=exp(lg)`. Once you have found the model with the best cross-validation error, train it on the full training set and then test it on the test set.