Leverage and Big Data

Aimee Barciauskas, Harihara Subramanyam, Roger Cuscó

December 21, 2015

1. SimDat1: Simulate a dataset from a linear regression model as follows. Generate 100K replications of 30 inputs using a 30-dim Gaussian with mean 0 and covariance the identity matrix. Generate the output as a linear regression on the inputs (without intercept) with coefficients randomly drawn from a multivariate Gaussian with mean 0 and variance 3. This will be SimDat1.

```
library(mvtnorm)
library(scales)
library(psych)
library(MASS)
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60))

# Generate 100k replications of 30 inputs
m <- 30
n <- 100000

sigma <- diag(m)
simdat1.xs <- rmvnorm(n, mean = rep(0, m), sigma = sigma)
```

2. Revise and briefly describe what is a mixture of Gaussian distributions (see e.g. Bishop) and describe a procedure for simulating from one.

A mixture of Gaussian distributions is a linear combination of multiple Gaussian Normal distributions, each having a mixing coefficient where the sum of mixture coefficients is equal to 1. Intuitively, the mixture coefficient k is the probability of an observation having been drawn from distribution k.

To simulate n observations from a mixture of k Gaussians, generate n random samples from a multinomial distribution having k outcomes with corresponding probabilities according to the target mixture. Assign the *ith* simulated observation to a distribution from the mixture according to these random draws, and make a random draw from the assigned distribution.

3. SimDat2: Simulate another linear regression dataset as follows. Generate 100K replications of 30 inputs from a mixture of 2 30-dim Gaussians; the first with mean 0 and covariance the identity matrix; the second with mean 0 and covariance 10 times the identity. The first component has weight 0.95 and the second 0.05. Generate the output as a linear regression on the inputs (without intercept) with coefficients randomly drawn from a multivariate Gaussian with mean 0 and variance 3. This will be SimDat2.

```
# Assign observations to mixtures
mixture_assignment <- rmultinom(n = n, size = 1, prob = c(0.95,0.05))
assignments <- apply(mixture_assignment, 2, function(col) { match(1,col)})

sigma.gaussian1 <- diag(m)
```

```
sigma.gaussian2 <- diag(m)*10

rdraw.gaussian1 <- function() {
  rmvnorm(1, mean = rep(0, m), sigma = sigma.gaussian1)
}

rdraw.gaussian2 <- function() {
  rmvnorm(1, mean = rep(0, m), sigma = sigma.gaussian2)
}

simdat2.xs <- matrix(0, nrow = n, ncol = m)

for (idx in 1:length(assignments)) {
  assignment <- assignments[idx]
  if (assignment == 1) {
    simdat2.xs[idx,] <- rdraw.gaussian1()[1,]
  } else {
    simdat2.xs[idx,] <- rdraw.gaussian2()[1,]
  }
}
```

4. Describe briefly what is leverage for linear regression, how it is computed and propose a linear transformation of the leverage that makes it numerically comparable across datasets with different number of inputs and different number of replications.

The leverage of the ith observation is the value $H_{i,i}$ of the hatmatrix H:

$H = \phi(\phi^T\phi)^{-1}\phi^T$

So all the values of leverage would be the diagonal entries of the hatmatrix.

The value of leverage of an individual datapoint:

$h_{i,i} = H[i,i]$

can be interpreted as the "number of paramaters" being used to fit the corresponding observation. In the case of extreme overfitting, where $m = n$, $h_{i,i}$ for each datapoint will be equal to 1. As n grows large, the influence each observation has on the parameters decreases. However, observations with high leverage will have a disproportionately large contribution to the values of the parameters.

Leverages follow a χ^2-squared distribution when the covariance equation:

X^TX

is scaled by the number of observations

$\frac{X^TX}{N}$

So the calculation for the scaled hatmatrix is:

$X^T(\frac{X^TX}{N})^{-1}X$

These scaled datasets follow a χ^2 distribution, with degrees of freedom equal to m.

To compare leverages across different data sets, we can evaluate the scaled leverages against the standard deviation of the corresponding χ^2 distribution.

The expected leverage is $(m+1)/n$, where m is the dimension of the inputs and n is the number of observations. However as shown in real datasets below, usually the average leverage is lower, and subsequently the variance higher, when compared with the corresponding χ^2, as relatively few observations have very high leverage.

Below we define functions for calculating and plotting scaled leverages.

```
inner.matrix <- function(phi, scaled = TRUE) {
  n <- nrow(phi)
  if (scaled == TRUE) {
    solve((1/n)*(t(phi)%*%phi))
  } else {
    solve((t(phi)%*%phi))
  }
}

hatmatrix <- function(phi, scaled = TRUE) {
  n <- nrow(phi)
  if (scaled == TRUE) {
    phi %*% inner.matrix(phi, scaled = TRUE) %*% t(phi)
  } else {
    phi %*% inner.matrix(phi, scaled = FALSE) %*% t(phi)
  }
}
leverages <- function(phi, scaled = TRUE) diag(hatmatrix(phi, scaled))

# Unscaled function for calculating inner matrix sequentially
# So far this is unused, it would only be necessary with many features and won't be comparable with the
sequential.inner.matrix <- function(file, max.size = Inf, only.cols = c()) {
  chunk.index <- 0
  chunk.size <- 1000
  data <- read.csv(file, nrow = chunk.size, skip = chunk.index)
  chunk.index <- chunk.index + chunk.size
  data <- data[,only.cols]
  X.current <- as.matrix(data)

  C.inv.current <- ginv(t(X.current) %*% X.current)
  C.inv.prev <- C.inv.current
  # read file from index to chunk and update vars
  while ((chunk.index+chunk.size) <= max.size) {
    # Read chunk of file as X_{k+1}
    data <- read.csv(file, nrow = chunk.size, skip = chunk.index)
    data <- data[,only.cols]
    X.current <- as.matrix(data)
    print(chunk.index)

    # Update C
    C.inv.current <- C.inv.prev - C.inv.prev %*% t(X.current) %*%
      ginv(diag(1, nrow(X.current)) + X.current %*% C.inv.prev %*% t(X.current)) %*%
      X.current %*% C.inv.prev

    # If size of current chunk is less than chunk size, break
    if (nrow(X.current) < chunk.size) break
    C.inv.prev <- C.inv.current
    chunk.index <- chunk.index + chunk.size
  }
  C.inv.current
}
```

```
leverages.sequential <- function(data.mat, inner.mat) {
  leverages <- c()
  for (i in 1:nrow(data.mat)) {
    leverages <- append(leverages, t(data.mat[i,]) %*% inner.mat %*% data.mat[i,])
  }
  leverages
}

add.alpha <- function(col, alpha=1){
  if(missing(col))
    stop("Please provide a vector of colours.")
  apply(sapply(col, col2rgb)/255, 2,
                     function(x)
                       rgb(x[1], x[2], x[3], alpha=alpha))
}

myColours = c("steelblue", "#FFBB00", rgb(0.4, 0.2, 0.3))
myColoursAlpha <- add.alpha(myColours, alpha=0.4)

plot.leverages.density <- function(m, leverages, dataset.title, leverages.2 = c()) {
  n <- length(leverages)
  lev.sd <- sqrt(2*m)
  threshold <- qchisq(0.90, df = m)
  rchisqs <- rchisq(1000, df = m)
  max.plot.y <- ifelse(length(leverages.2) > 0,
                       max(max(density(leverages.2)$y), max(density(leverages)$y)),
                       max(density(leverages)$y))
  plot(density(sample(leverages, 1000)),
       col = myColoursAlpha[3],
       main = paste(dataset.title, 'Leverage Density'),
       ylim = c(0,max.plot.y),
       xlab = '',
       xlim = c(0, ((m+1)/n) + 24*lev.sd))
  if (length(leverages.2 > 0)) lines(density(leverages.2), col = myColoursAlpha[1])

  lines(density(rchisqs), col = 'grey23')
  lines(x = rep(threshold, 1001), y = seq(0,1,.001), col = 'gray31', lty = 5)
  lines(x = rep((3*lev.sd + m), 1001), y = seq(0,1,.001), col = 'gray31', lty = 2)
  legend('topright',
         c('3 std dev','90th percentile'),
         lty=5:2,
         cex = 0.5,
         lwd= 1,
         bty = 'n',
         col=c('grey31','grey31'))
}

plot.leverages <- function(m, leverages, dataset.title, leverages.2 = c()) {
  n <- length(leverages)
  lev.sd <- sqrt(2*m)
  threshold <- qchisq(0.90, df = m)
  plot(leverages,
       col = myColoursAlpha[3],
```

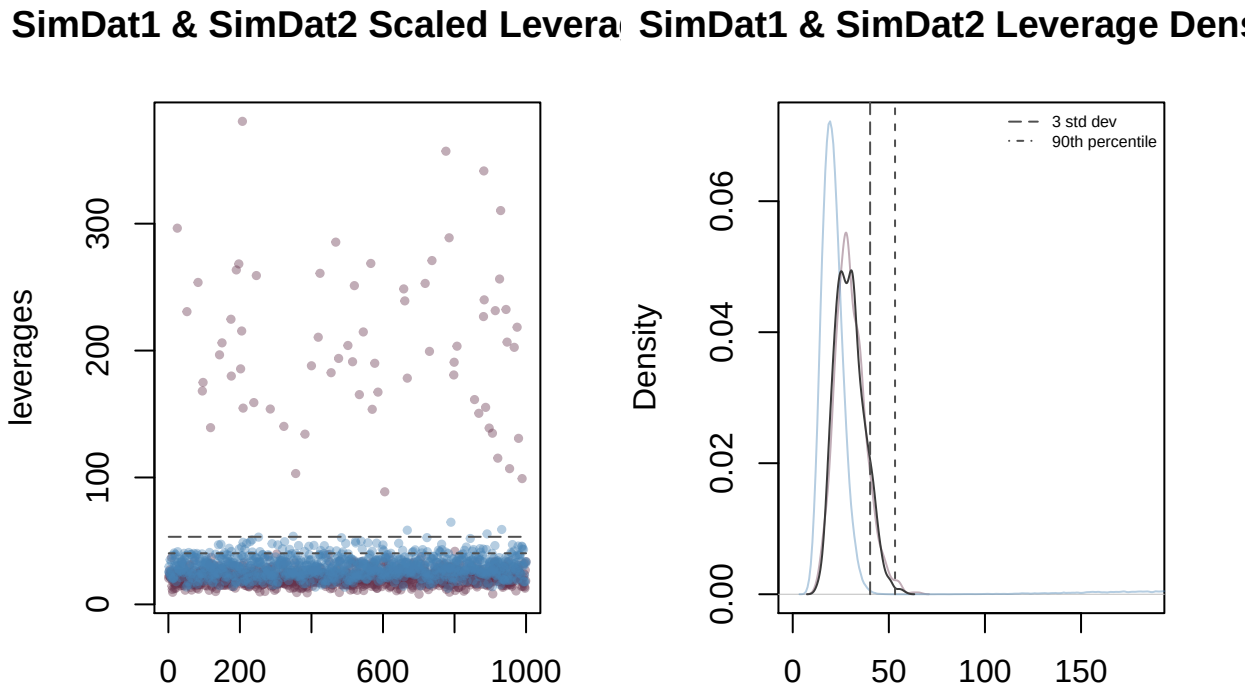
```
       cex = 0.5,
       pch = 19,
       main = paste(dataset.title, 'Scaled Leverages'),
       xlab = '')
  if (length(leverages.2 > 0)) {
    points(x = leverages.2,
         cex = 0.5,
         col = myColoursAlpha[1],
         pch = 19)
  }
  # Thresholds
  lines(y = rep((3*lev.sd + m), n), x = seq(1,n,1), col = 'gray31', lty = 5)
  lines(y = rep(threshold, n), x = seq(1,n,1), col = 'gray31', lty = 2)
}
```

The functions above are used to calculate the leverages for the simulated data and the leverages are compared with a χ^2-distribution having 30 degrees of freedom.

To visualize which data points qualify as "high-leverage", we plotof SimDat1 and SimDat2 with grey lines marking three standard deviations above the mean and the 90th precentile of the comparison χ^2 distribution. Three standard deviations and the 90th precentile provide 2 options for qualifying data points as "high leverage".

In these plots, you can clearly see that nearly all the values above the three-standard-deviations threshold are from the distribution with a variance of 10. SimDat1 has a higher average value than the comparison χ^2 distribution. This makes sense: given all the leverages should add up to the number of features, if a data set has a few very high leverage observations, the others will be necessarily lower.

```
simdat1.inner.mat <- inner.matrix(simdat1.xs)
simdat2.inner.mat <- inner.matrix(simdat2.xs)
simdat1.leverages <- leverages.sequential(simdat1.xs, simdat1.inner.mat)
simdat2.leverages <- leverages.sequential(simdat2.xs, simdat2.inner.mat)

par(mfrow=c(1,2))

plot.leverages(m, sample(simdat2.leverages, 1000),
               'SimDat1 & SimDat2',
               sample(simdat1.leverages, 1000))
plot.leverages.density(ncol(simdat1.xs),
                       simdat1.leverages,
                       'SimDat1 & SimDat2',
                       leverages.2 = simdat2.leverages)
```



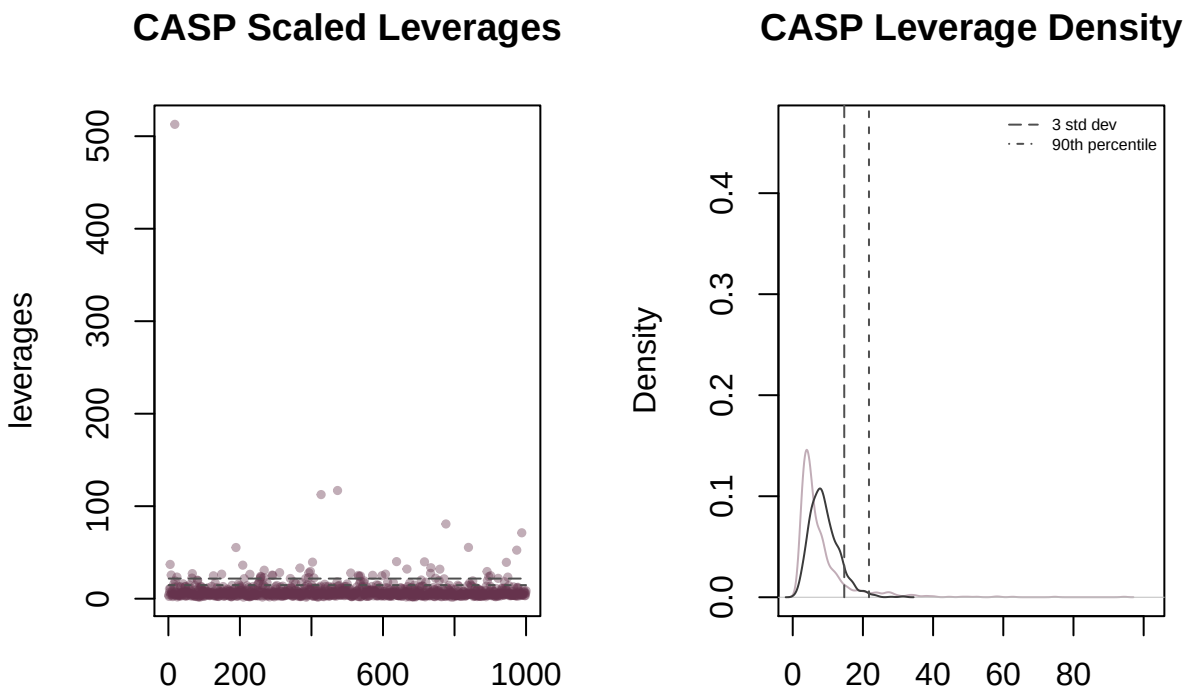
5. Find from ML (or otherwise) data repositories (e.g. UCI) a number of big datasets (maximum 10) with continuous output and continuous inputs (maximum work with 50 inputs). For each of these datasets compute the leverages.

6. The main output of this project is a visualization of the leverage distributions, one for each of the ML datasets as well as SimDat1 and SimDat2, that allows direct comparison between them. The aim is to to uncover structure that might be common in big data sets amenable to regression analysis.

CASP Data: Physicochemical Properties of Protein Tertiary Structure

Dataset Homepage

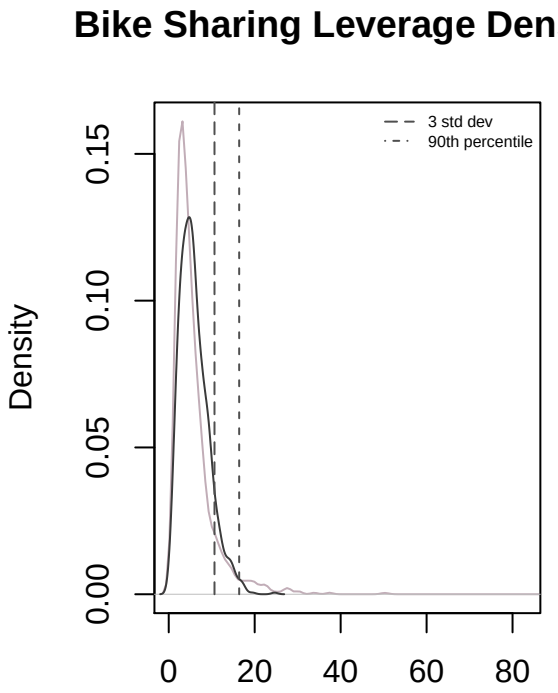
```
casp.data <- read.csv('~/Projects/data_files/CASP.csv')
casp.features <- as.matrix(casp.data[,2:ncol(casp.data)])
casp.inner.mat <- inner.matrix(casp.features)
casp.leverages <- leverages.sequential(casp.features, casp.inner.mat)

par(mfrow=c(1,2))
plot.leverages(ncol(casp.features), sample(casp.leverages, 1000), 'CASP')
plot.leverages.density(ncol(casp.features), casp.leverages, 'CASP')
```

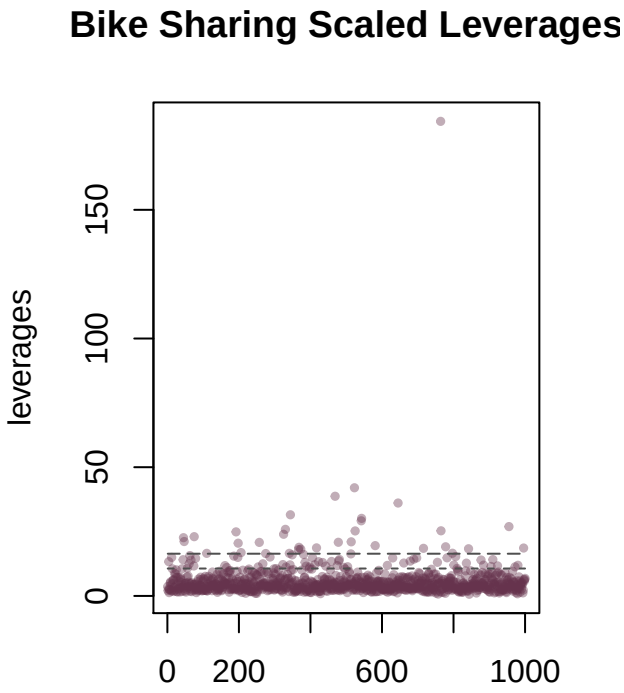


Bike Sharing Dataset

Dataset Homepage

```
bikes.data <- read.csv('~/Projects/data_files/Bike-Sharing-Dataset/hour.csv')
bikes.features <- bikes.data[,c('temp','atemp','hum','windspeed','casual', 'registered')]

bikes.leverages <- leverages(as.matrix(bikes.features))

par(mfrow=c(1,2))
plot.leverages.density(ncol(bikes.features), sample(bikes.leverages, 1000), 'Bike Sharing')
plot.leverages(ncol(bikes.features), sample(bikes.leverages, 1000), 'Bike Sharing')
```

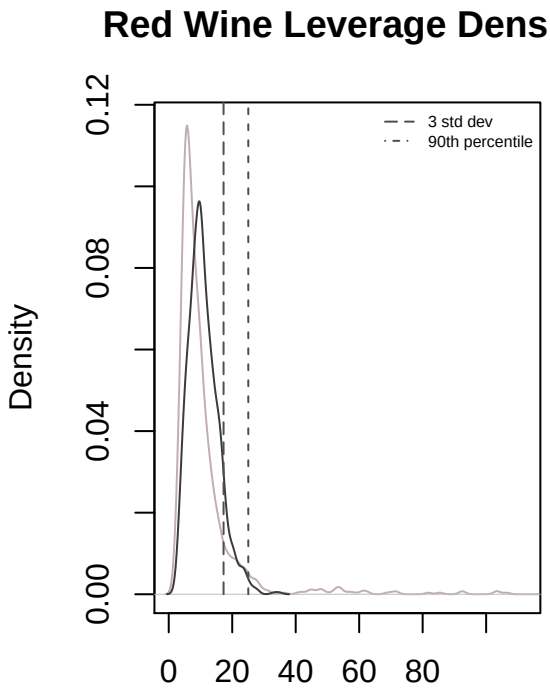


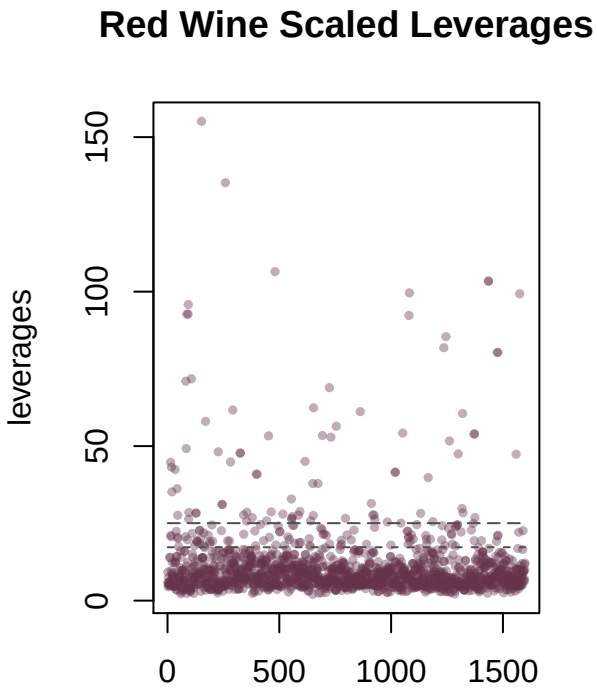


Wine Quality Dataset

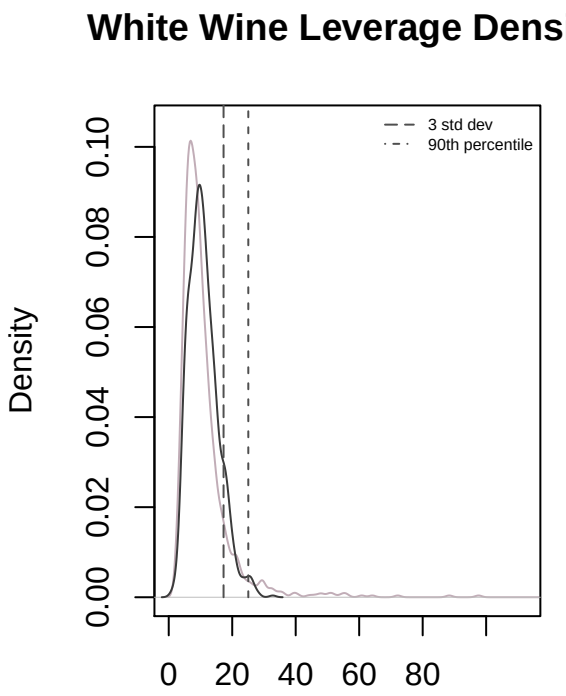
Dataset Homepage

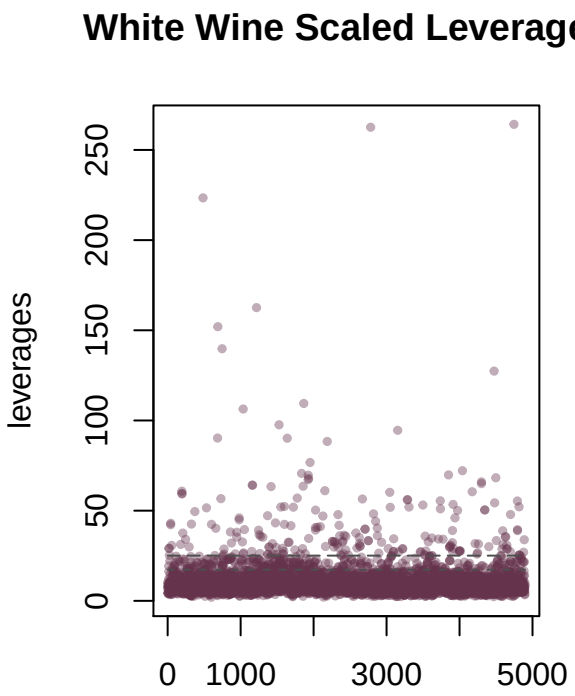
```
redwine.data <- read.delim('~/Projects/data_files/wine/winequality-red.csv', sep=';')
redwine.features <- redwine.data[,setdiff(colnames(redwine.data), 'quality')]
redwine.leverages <- leverages(as.matrix(redwine.features))

whitewine.data <- read.delim('~/Projects/data_files/wine/winequality-white.csv', sep=';')
whitewine.features <- whitewine.data[,setdiff(colnames(whitewine.data), 'quality')]
whitewine.leverages <- leverages(as.matrix(whitewine.features))

par(mfrow=c(1,2))
plot.leverages.density(ncol(redwine.features), redwine.leverages, 'Red Wine')
plot.leverages(ncol(redwine.features), redwine.leverages, 'Red Wine')
```






```
par(mfrow=c(1,2))
plot.leverages.density(ncol(whitewine.features), whitewine.leverages, 'White Wine')
plot.leverages(ncol(whitewine.features), whitewine.leverages, 'White Wine')
```

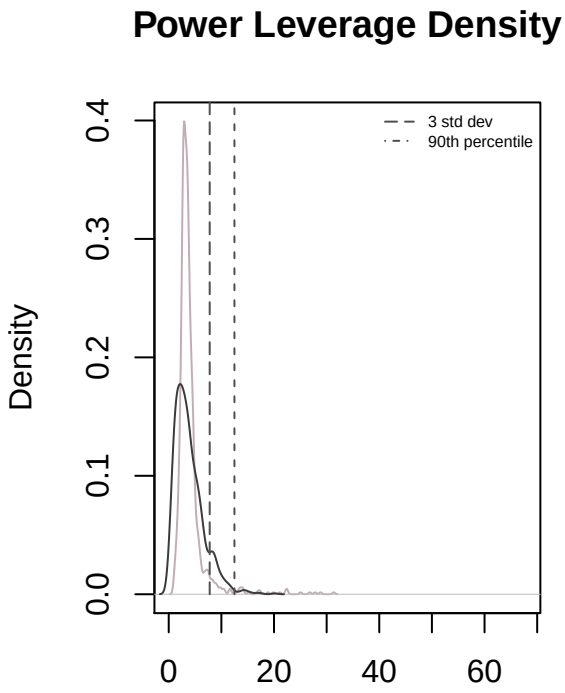


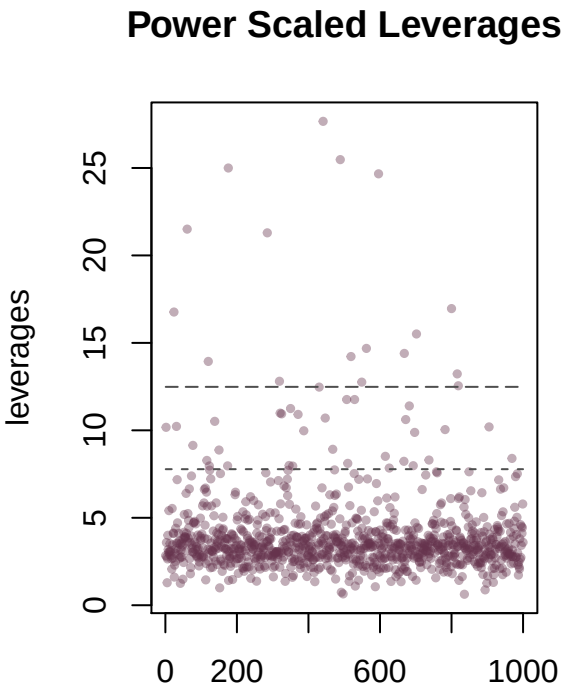


Individual household electric power consumption Data Set

Dataset Homepage

```
power.data <- read.delim('~/Projects/data_files/household_power_consumption.txt', sep = ';')
power.features <- power.data[1:50000,c('Global_active_power',
                                       'Global_reactive_power',
                                       'Voltage',
                                       'Global_intensity')]
power.features <- data.matrix(power.features)
power.inner.mat <- inner.matrix(power.features)

power.leverages <- leverages.sequential(power.features, power.inner.mat)

par(mfrow=c(1,2))
plot.leverages.density(ncol(power.features), sample(power.leverages, 1000), 'Power')
plot.leverages(ncol(power.features), sample(power.leverages, 1000), 'Power')
```

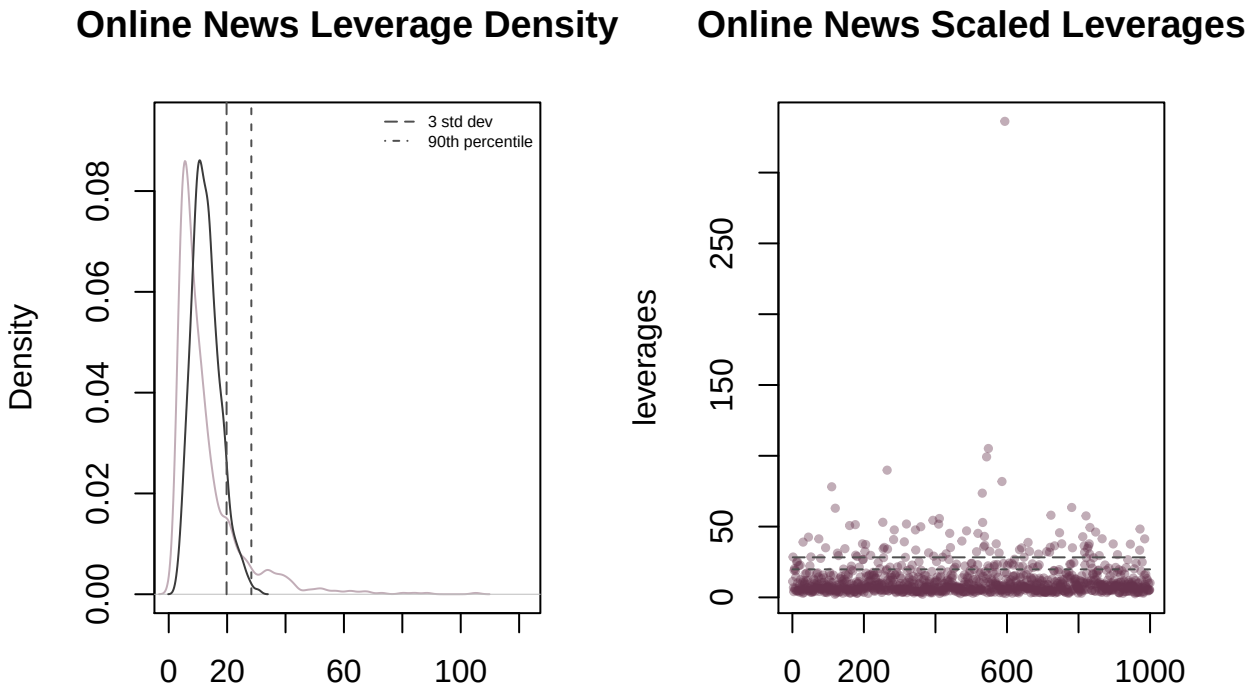




Online News Popularity

Dataset Homepage

```
news.data <- read.csv('~/Projects/data_files/OnlineNewsPopularity/OnlineNewsPopularity.csv')
news.features <- news.data[,c('n_tokens_title', 'n_tokens_content','num_hrefs', 'num_imgs', 'num_videos
news.inner.mat <- inner.matrix(as.matrix(news.features))
news.leverages <- leverages.sequential(as.matrix(news.features), news.inner.mat)

par(mfrow=c(1,2))
plot.leverages.density(ncol(news.features), news.leverages, 'Online News')
plot.leverages(ncol(news.features), sample(news.leverages, 1000), 'Online News')
```

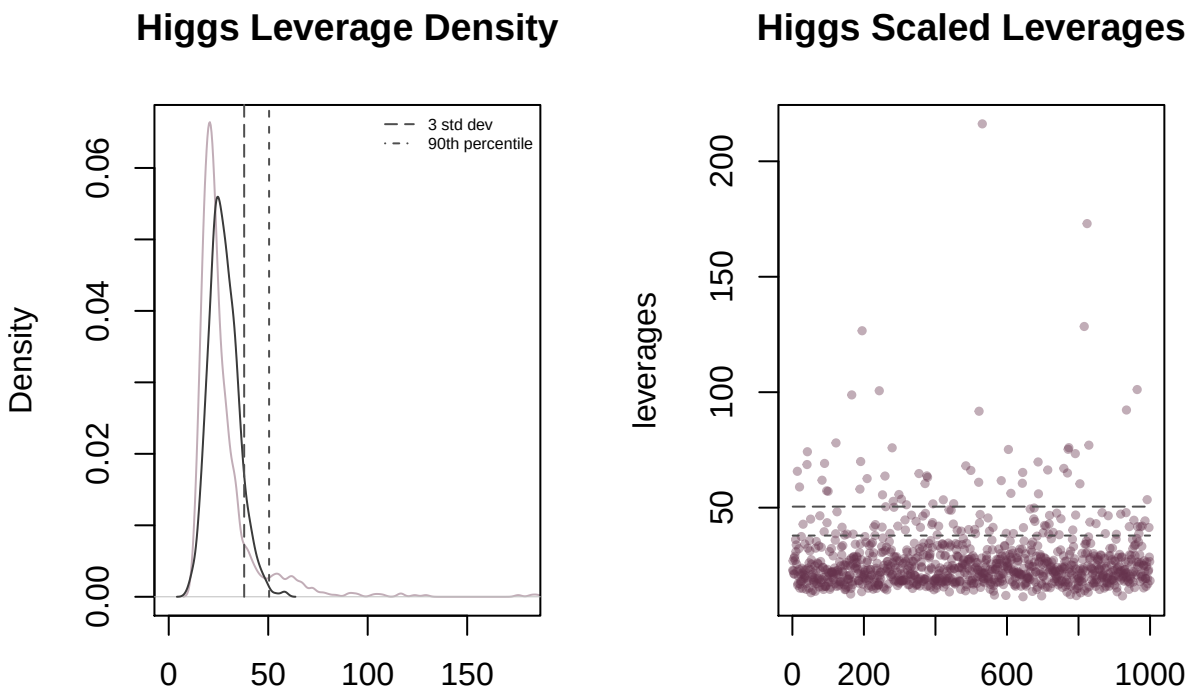


Higgs

Dataset Homepage

```
require(ffbase)

### COMMENT OUT IF YOU HAVE ALREADY DONE THIS STEP
# system("gunzip -kdv ~/Projects/data_files/HIGGS.csv.gz")
# ## Read the HIGGS data in a ffdf object: Had to wait ~ 15 minutes
# ## on my laptop for the following to complete
# varnames <- c("signal", paste0("feature", 1:21), paste0("HLfeature",1:7))
# Higgs_ffdf <- read.csv.ffdf(file = "~/Projects/data_files/HIGGS.csv", header = FALSE, VERBOSE = TRUE,
# ## set test variable (0 if observation is for training, 1 if for test)
# Higgs_ffdf$test <- c(ff(0, 10500000), ff(1, 500000))
# ## Write ffdf object to the disk and go back to the working directory
# save.ffdf(Higgs_ffdf, dir = "~/Projects/data_files/HIGGSffdf", overwrite = TRUE)
### END COMMENT OUT IF YOU HAVE ALREADY DONE THIS STEP

load.ffdf("~/Projects/data_files/HIGGSffdf")
testdat_ffdf <- subset(Higgs_ffdf, test == 1)
# Remove signal and test columns
testdat_ffdf <- testdat_ffdf[1:50000,2:(ncol(testdat_ffdf)-1)]

higgs.inner.mat <- inner.matrix(as.matrix(testdat_ffdf))
higgs.leverages <- leverages.sequential(as.matrix(testdat_ffdf), higgs.inner.mat)
```

```
par(mfrow=c(1,2))
plot.leverages.density(ncol(testdat_ffdf), higgs.leverages, 'Higgs')
plot.leverages(ncol(testdat_ffdf), sample(higgs.leverages, 1000), 'Higgs')
```



Conclusions

Scaling leverages by the number of observations enables comparison with an expected distribution: the χ^2 distribution can be used to inspect if the distribution of leverages is a "balanced" distribution or has many extreme values.

The real datasets above demonstrated varying levels of imbalance in comparison with the corresponding χ^2 distribution. A commonality was a lower mean and higher variance. This is the result of the property leverages must add up to the number of parameters. Thus a few extreme high-leverages must be "balanced out" by a lower mean for the remaining observations.