Homework 2

Aaron Politsky

October 9, 2015

5.1

Load Libraries and Source Dependencies

```
library(data.table)
library(ggplot2)
library(kknn)
source("docv.R")
```

Import the Used Car data

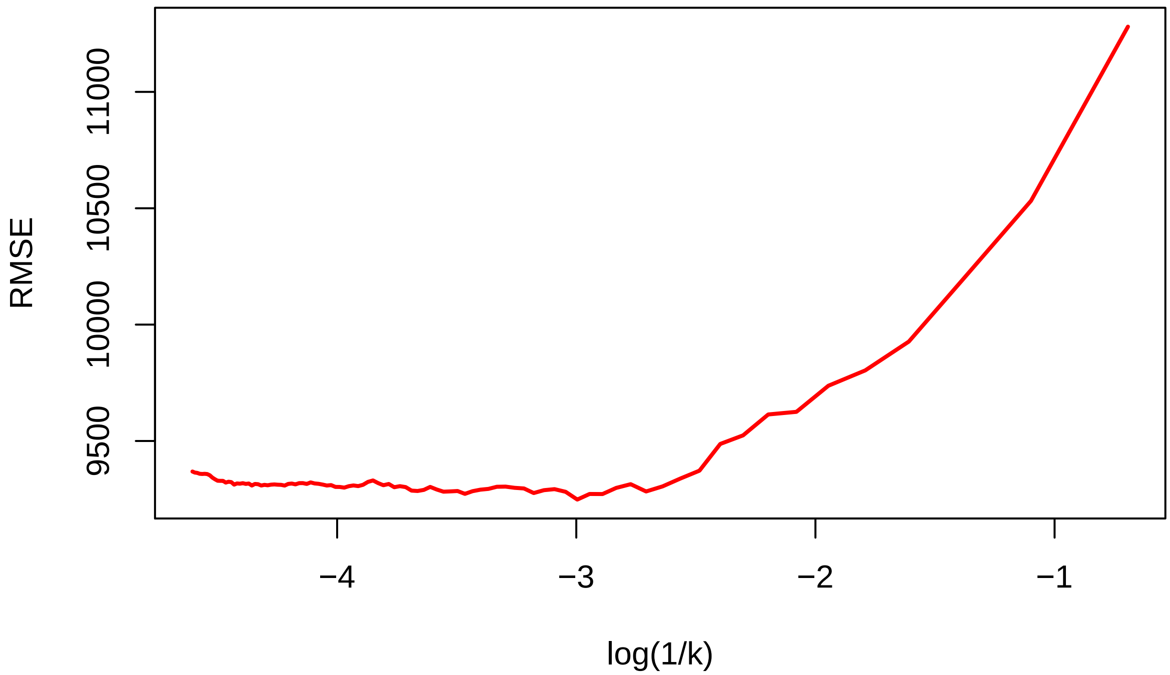
```
# download data and read data into data.table format
if(!exists("used_cars")) {
  used_cars <- fread(
    'https://raw.githubusercontent.com/ChicagoBoothML/DATA___UsedCars/master/UsedCars_small.csv')
}

# sort data set by increasing mileage
setkey(used_cars, mileage)
```

Cross-validate using 5-fold, for k from 2 to 100

```
n.folds <- 5

set.seed(99) # to be consistent with hw1
kv <- 2:100
cv <- docvknn(x=matrix(used_cars$mileage),
              y=used_cars$price,
              k=kv,
              nfold=n.folds)
```

```
## in docv: nset,n,nfold:  99 1000 5
## on fold:  1 , range:  1 : 200
## on fold:  2 , range:  201 : 400
## on fold:  3 , range:  401 : 600
## on fold:  4 , range:  601 : 800
## on fold:  5 , range:  801 : 1000
```

```r
# convert to RMSE
cv <- sqrt(cv/length(used_cars$price))

# how does our CV plot look?
rgy <- range(cv)
plot(log(1/kv),cv,type="l",col="red",ylim=rgy,lwd=2,
     xlab="log(1/k)", ylab="RMSE")
```

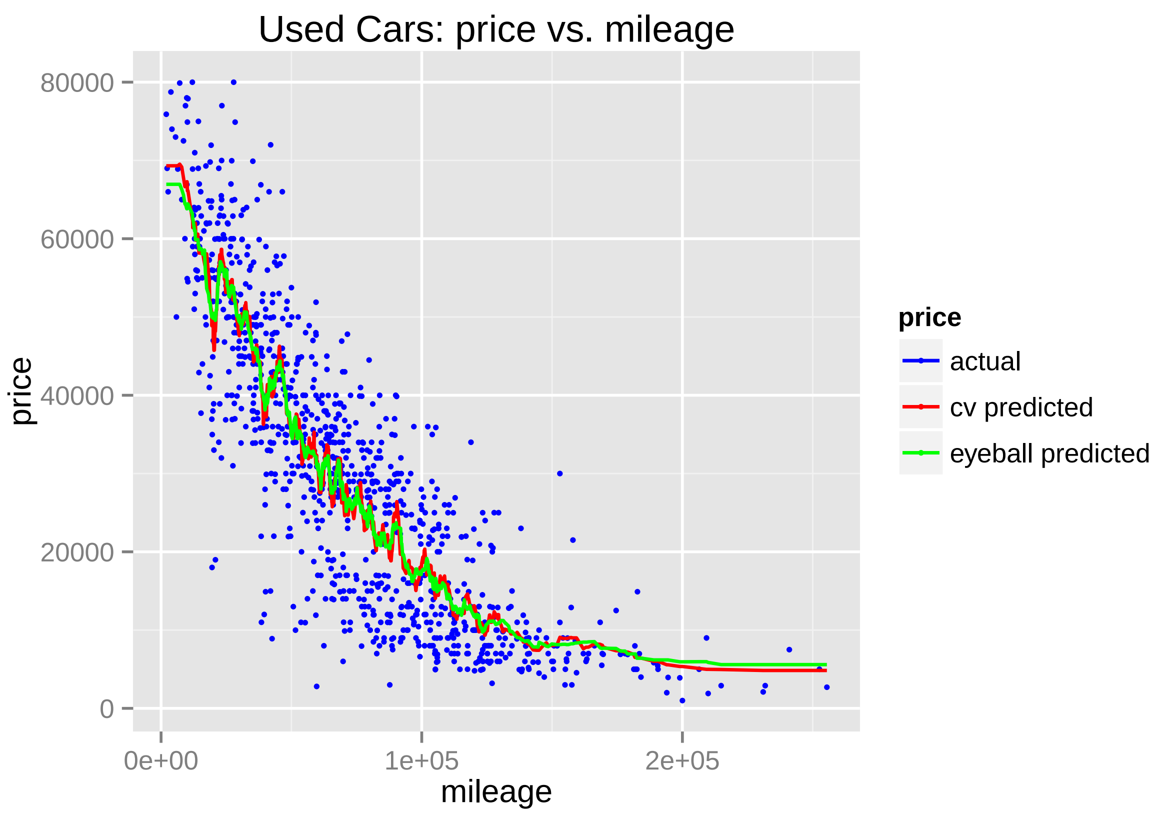



```r
# save this cv as cv.mileage
cv.mileage <- cv

# Choose the k with the minimum Cross-validation RMSE
k.cv <- which.min(cv)
```

We will use a k value of $k = 19. Let's fit using this value and the eyeball value of 30:

```r
# Produce a model with this k
cv.model <- kknn(price ~ mileage,
                 train=used_cars, test=used_cars[ , .(mileage)],
                 k=k.cv, kernel='rectangular')

# add the predicted values to our data.table
used_cars$cv.predicted <- cv.model$fitted.values

# Produce the eyeball fit from hw1
k <- 30
eyeball <- kknn(price ~ mileage,
                train=used_cars, test=used_cars[ , .(mileage)],
                k=k, kernel='rectangular')

# add this prediction to used_cars data.table
```

```
used_cars$eyeball.predicted <- eyeball$fitted.values

g <- ggplot(used_cars) +
  geom_point(aes(x=mileage, y=price, color='actual'), size=1) +
  ggtitle('Used Cars: price vs. mileage') +
  xlab('mileage') + ylab('price')
g <- g +
  geom_line(aes(x=mileage, y=cv.predicted, color='cv predicted'), size=0.6) +
  geom_line(aes(x=mileage, y=eyeball.predicted, color='eyeball predicted'), size=0.6) +
  scale_colour_manual(name='price',
                      values=c(actual='blue',
                               "cv predicted"='red',
                               "eyeball predicted"="green"))
plot(g)
```



In this case, our eyes have not deceived us. The fits are very close. This makes sense, because \$cv[19] = 9248.2401549 and \$cv[30] = 9284.1316686. The fit having the larger k value seemse to have slighly less variance (the green curve seems to be "within" the red curve) than than the fit with the smaller k value, which is consistent with the theory of knn.

Predicting the price of a car with 100k miles

Now, let's predict the price of a car having 100k miles using our cross-validation k to train a model using the entire used car dataset.

```r
# simulate a car with 100k miles
car.w.100k.miles <- data.frame(mileage=100000)
predicted.price.car.w.100k.miles <-
  kknn(price ~ mileage,
       train=used_cars, test=car.w.100k.miles,
       k=k.cv, kernel='rectangular')$fitted.values
predicted.price.car.w.100k.miles
```

```
## [1] 18662.63
```

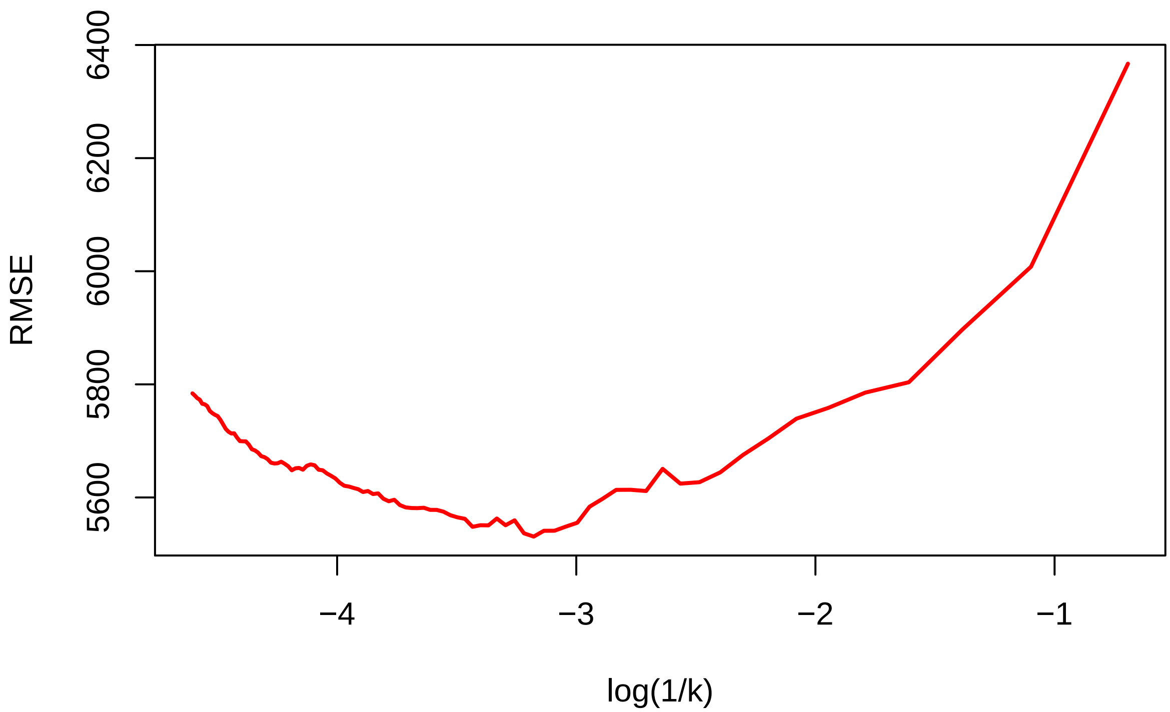
6.1

Use kNN to get a prediction for a 2008 car with 75k miles

```r
# Let's build a model including year and mileage in our covariates

# First, define our rescaling function
rescale <- function(x, xs) {
  (x - min(xs)) / (max(xs) - min(xs))
}

# first, lets scale our covariates
used_cars$normalized.mileage <- rescale(used_cars$mileage, used_cars$mileage)
used_cars$normalized.year <- rescale(used_cars$year, used_cars$year)

cv <- docvknn(x=used_cars[, .(normalized.mileage, normalized.year)],
              y=used_cars$price,
              k=kv,
              nfold=n.folds)
```

```
## in docv: nset,n,nfold:  99 1000 5
## on fold:  1 , range:  1 : 200
## on fold:  2 , range:  201 : 400
## on fold:  3 , range:  401 : 600
## on fold:  4 , range:  601 : 800
## on fold:  5 , range:  801 : 1000
```

```r
# convert to RMSE
cv <- sqrt(cv/length(used_cars$price))

# how does our CV plot look?
rgy <- range(cv)
plot(log(1/kv),cv,type="l",col="red",ylim=rgy,lwd=2,
     xlab="log(1/k)", ylab="RMSE")
```




```r
# Choose the k with the minimum Cross-validation RMSE
k.cv <- which.min(cv)
k.cv
```

```
## [1] 23
```

```r
# fit the whole model
predicted.mileage.year <-
  kknn(price ~ normalized.mileage + normalized.year,
     train=used_cars, test=used_cars[, .(normalized.mileage, normalized.year)],
     k=k.cv, kernel='rectangular')$fitted.values
```

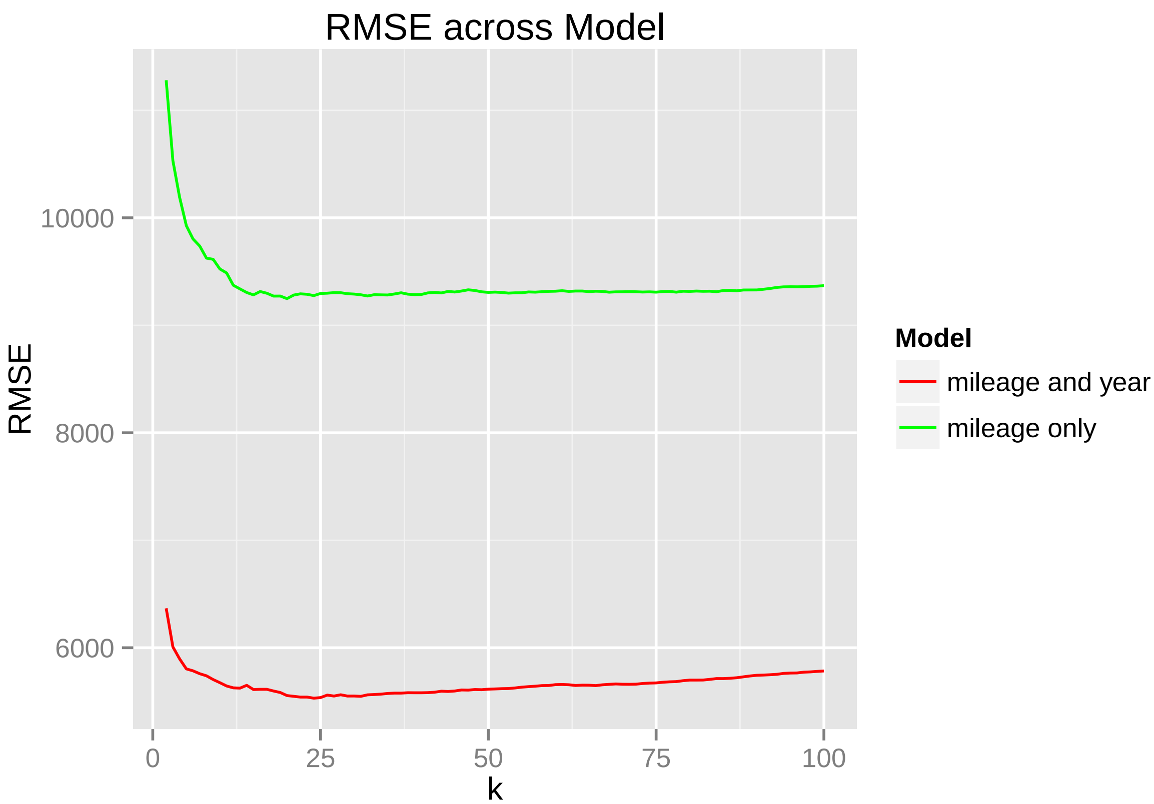
Let's define our test car

```r
# let's predict the value of a 2008 car having 75,000 miles
test.car <-
  data.frame(normalized.mileage=rescale(75000, used_cars$mileage),
             normalized.year=rescale(2008, used_cars$year))

predicted.price <-
  kknn(price ~ normalized.mileage + normalized.year,
       train=used_cars, test=test.car,
       k=k.cv, kernel='rectangular')$fitted.values
```

Is our predictive accuracy better using this model?

We could look at RMSE…

```r
# is our RMSE from using mileage and year better than our RMSE from using only mileage?
ggplot() +
  geom_line(aes(x=kv, y=cv, color='mileage and year')) +
  geom_line(aes(x=kv, y=cv.mileage, color='mileage only')) +
  ggtitle('RMSE across Model') +
  xlab('k') + ylab('RMSE') +
  scale_colour_manual(name='Model',
                      values=c("mileage and year"='red',
                               "mileage only"="green"))
```



But let's check the correlations of our predicted values vs. actual values instead:

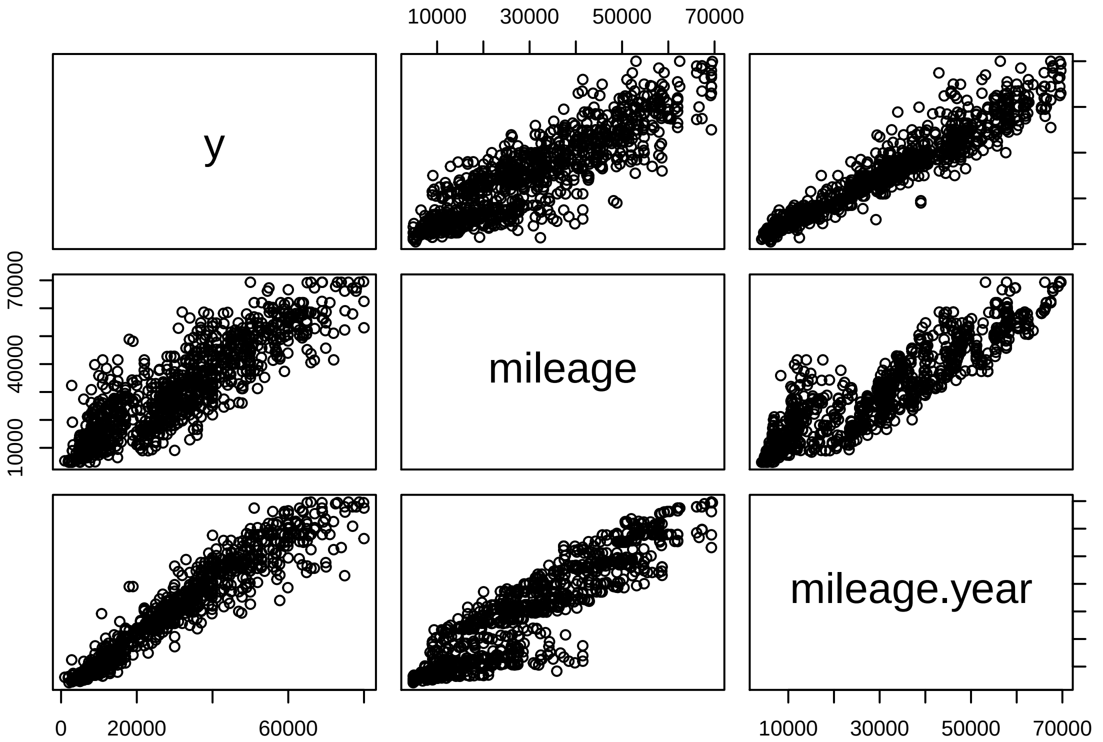
```r
fits <-
  data.frame(y=used_cars$price,
             mileage=used_cars$cv.predicted,
             mileage.year=predicted.mileage.year)

cor(fits)
```

```
##                      y   mileage mileage.year
## y            1.0000000 0.8775556    0.9583537
## mileage      0.8775556 1.0000000    0.9091338
## mileage.year 0.9583537 0.9091338    1.0000000
```

The correlation suggests that our mileage and year model provides a better fit than our bivariate model.

And finally,

```r
pairs(fits)
```



the plots confirm this. Our mileage and year model appears to have a much tighter fit.