Model assessment

We'll now look at different model assessment techniques, focusing on variance/bias of their estimate and computational cost.

We'll generate data from a linear model (with white noise) and use KNN to estimate the curve.

```r
# Dataset generation
f1dim<-function(x){ sin(8*x)/(1+(4*x)^2) }

DataGenerator <- function(n, p, sd.x, sd.eps) {
  X <- replicate(p, rnorm(n, sd = sd.x))
  eps <- rnorm(n, sd=sd.eps)
  Y <- f1dim(X[,1]) + eps
  return(data.frame(Y = Y, X = X))
}
```

```r
library(kknn)
```

```
## Warning: package 'kknn' was built under R version 3.6.3
```

```r
train <- DataGenerator(n=200, p=10, sd.x=5, sd.eps=1)
test <- DataGenerator(n=100, p=10, sd.x=5, sd.eps=1)
knn.8 <- kknn(formula=Y~. , train = train, test = test, k = 8)
```

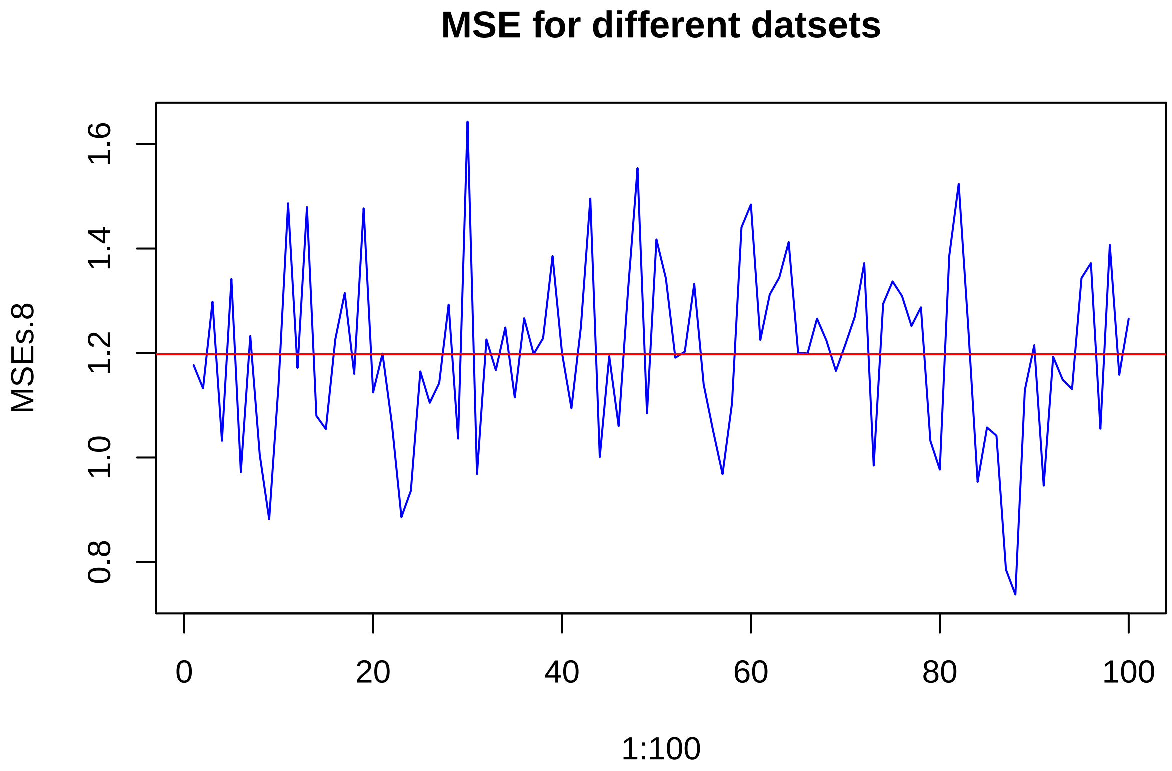
```r
test.preds <- predict(knn.8)
MSE.test <- sum((test$Y - test.preds)**2)/100
MSE.test
```

```
## [1] 1.442654
```

Now we'll simulate 100 datasets to approximate the test MSE.

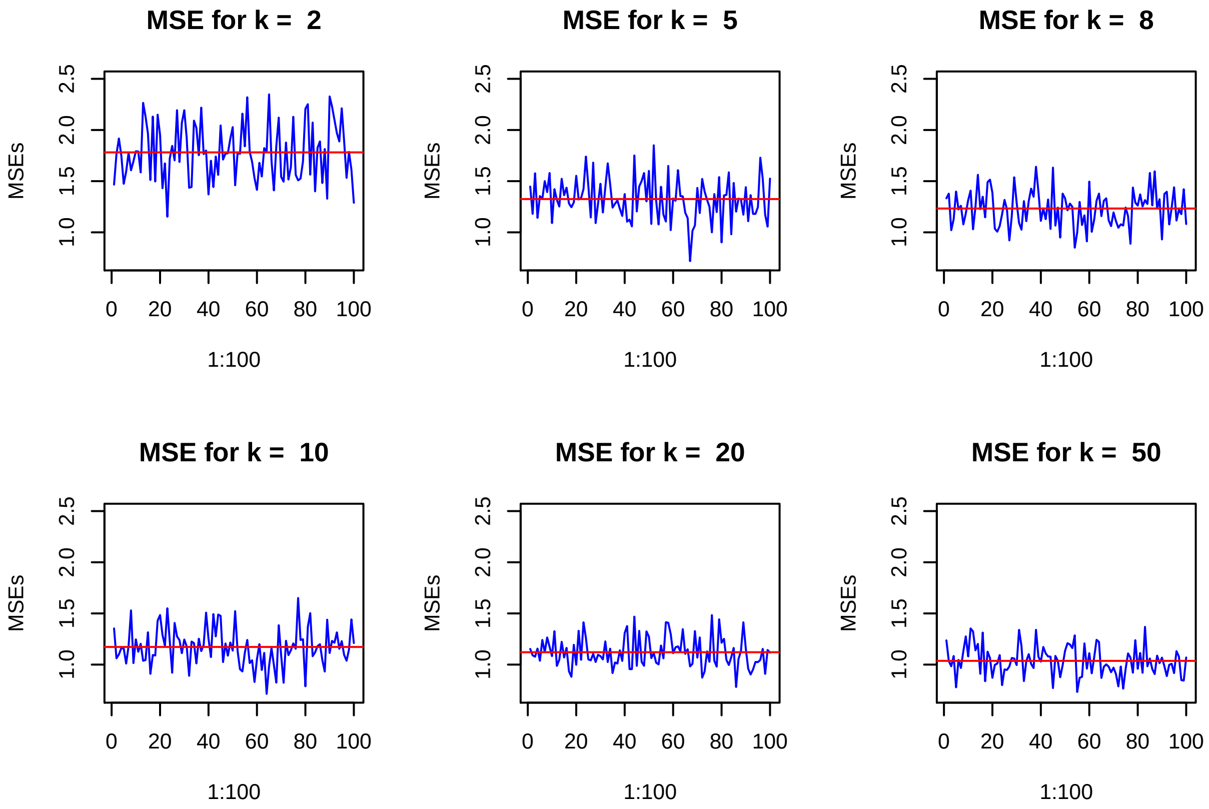
```r
nsim <- 100
simulate.knn <- function(nsim, k, sd.x=5, sd.eps=1){
  MSEs <- matrix(nrow=nsim, ncol=1)
  for(i in 1:nsim){
    train <- DataGenerator(n=200, p=10, sd.x=sd.x, sd.eps=sd.eps)
    test <- DataGenerator(n=100, p=10, sd.x=sd.x, sd.eps=sd.eps)
    knn.8 <- kknn(formula=Y~. , train = train, test = test, k = k)
    test.preds <- predict(knn.8)
    MSE.test <- sum((test$Y - test.preds)**2)/100
    MSEs[i]<-MSE.test
  }
  return(MSEs)
}
```

```r
MSEs.8 <- simulate.knn(100, 8)
plot(1:100, MSEs.8, type="l", main="MSE for different datsets", col="blue")
abline(h=mean(MSEs.8), col="red")
```

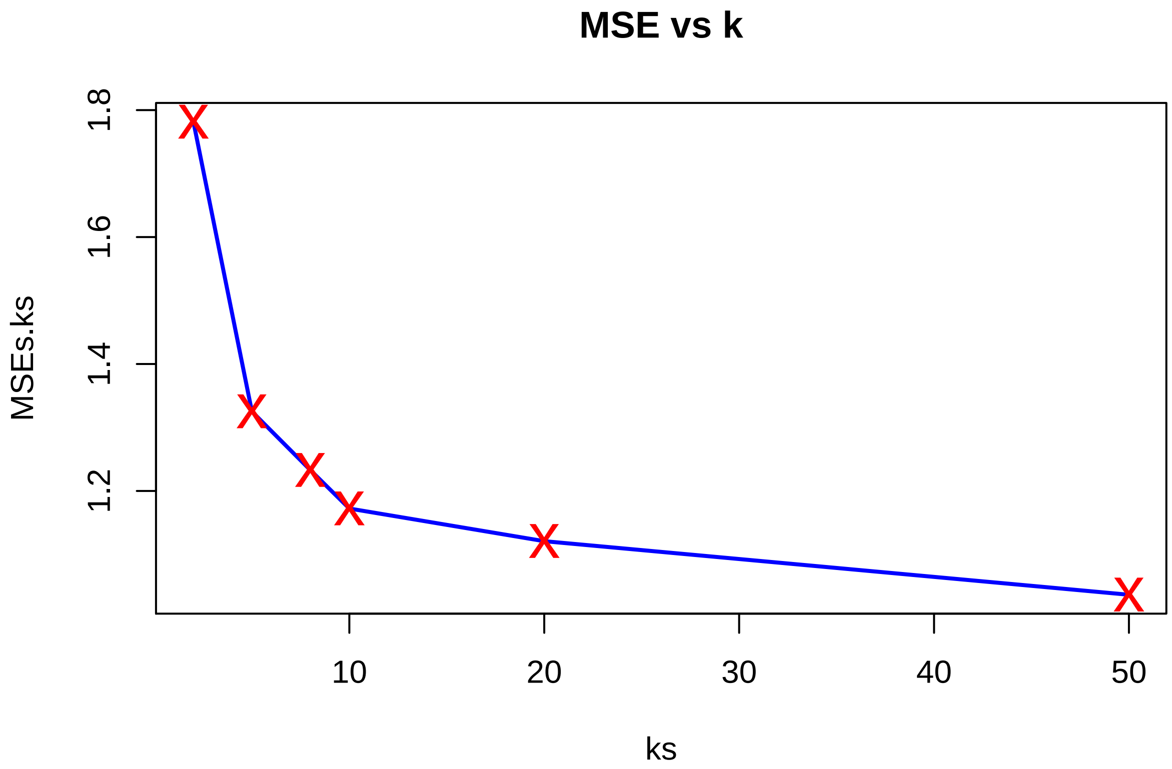


Now let's investigate how the MSE would change for different k.

```
ks <- c(2,5,8,10,20,50)
par(mfrow=c(2,3))
MSEs.ks <- matrix(nrow = 6, ncol=1)
for(i in 1:length(ks)){
  k <- ks[i]
  MSEs <- simulate.knn(100, k)
  plot(1:100, MSEs, type="l", main=paste("MSE for k = ",k), col="blue", ylim =c(0.7,2.5))
  abline(h=mean(MSEs), col="red")
  MSEs.ks[i] <- mean(MSEs)
}
```

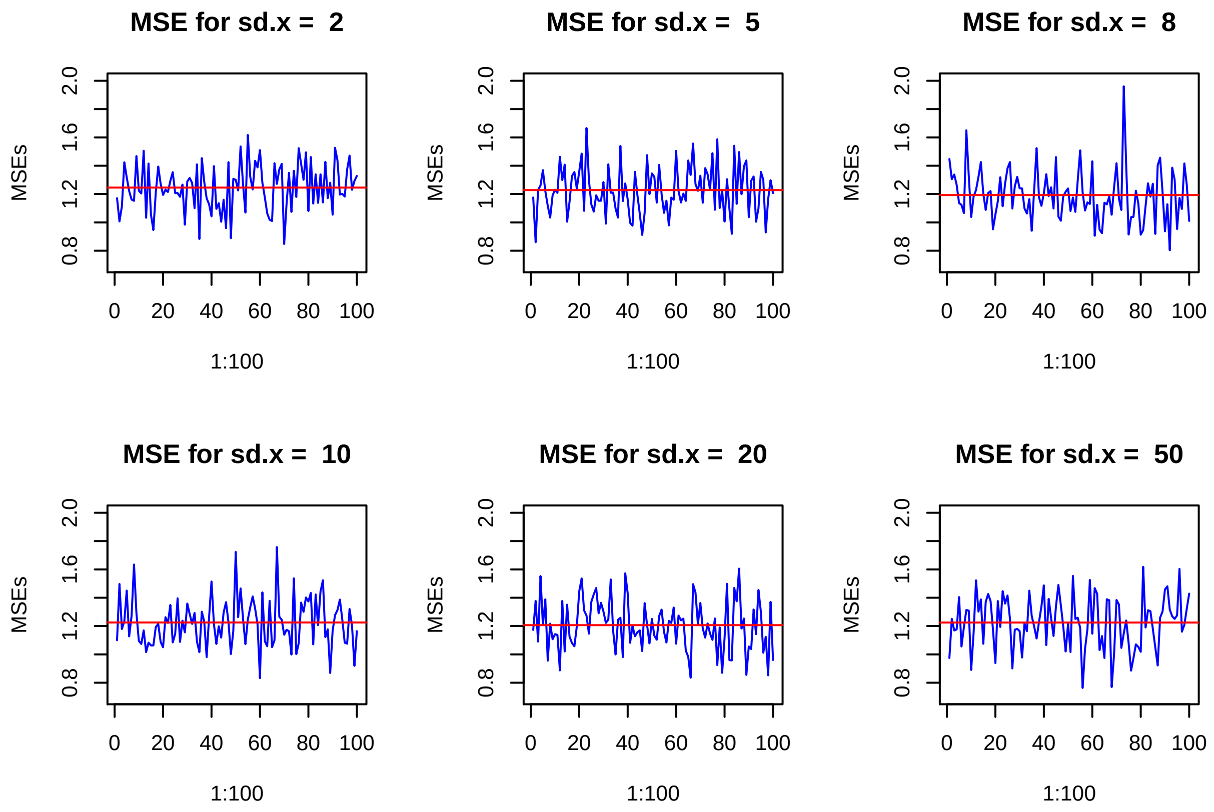


```r
par(mfrow=c(1,1))
plot(ks, MSEs.ks, type="l", main = "MSE vs k", col="blue", lwd=2)
points(ks, MSEs.ks, pch="x", cex=2, col="red")
```

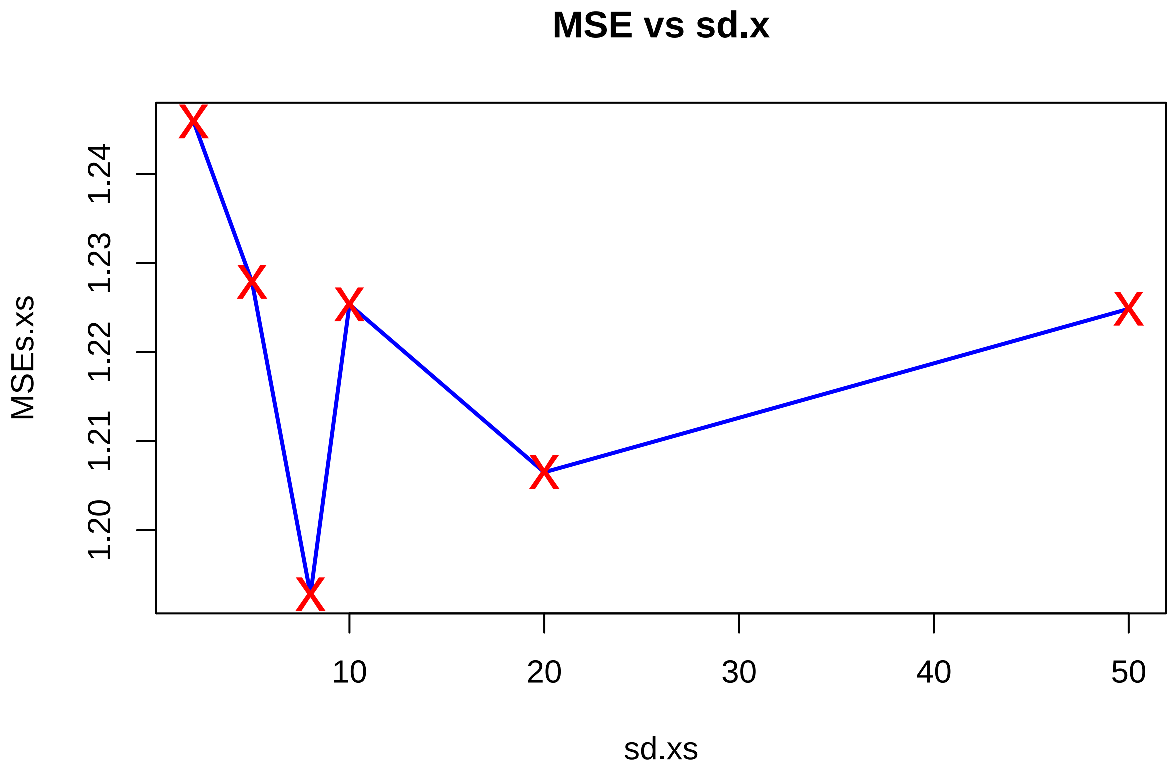


What if we keep k fixed but change the variance in x?

```r
sd.xs <- c(2,5,8,10,20,50)
par(mfrow=c(2,3))
MSEs.xs <- matrix(nrow = 6, ncol=1)
for(i in 1:length(sd.xs)){
  sd.x <- sd.xs[i]
  MSEs <- simulate.knn(100, 8, sd.x = sd.x)
  plot(1:100, MSEs, type="l", main=paste("MSE for sd.x = ",sd.x), col="blue", ylim =c(0.7,2.0))
  abline(h=mean(MSEs), col="red")
  MSEs.xs[i] <- mean(MSEs)
}
```



```
par(mfrow=c(1,1))
plot(sd.xs, MSEs.xs, type="l", main = "MSE vs sd.x", col="blue", lwd=2)
points(sd.xs, MSEs.xs, pch="x", cex=2, col="red")
```


MSE vs sd.x
MSEs.xs
1.20
1.21
1.22
1.23
1.24
10
20
30
40
50
sd.xs