Statistics 452: Statistical Learning and Prediction

Chapter 3, Part 4: Linear Regression vs K-Nearest Neighbors

Brad McNeney

2018-09-16

Parametric vs Non-Parametric

- Linear regression is parametric.
  - We assume a functional (parametrized) form for f, and then fitting f amounts to fitting parameters.
- Non-parametric regression does not assume a parametric form for f.
  - More flexible.
  - A simple example is K-nearest neighbors (KNN) regression.

KNN Regression

- Define a neighborhood size K.
- For each x_i, take $\hat{f}(x_i)$ to be the average of the y_j's for x_j's in the neighborhood of x_i.

Example Neighborhood

```
Xdat <- advert[,c("TV","radio")]
dm <- as.matrix(dist(Xdat))
dm[1:3,1:3]
```

```
##          1         2         3
## 1   0.0000 185.60606 213.05403
## 2 185.6061   0.00000  28.08647
## 3 213.0540  28.08647   0.00000
```

```
dd <- dm[,1] # distances from first x
nbrThresh <- sort(dd)[9] # find 9th smallest distance
nbrThresh
```

```
##       69
## 12.62458
```

```
nn <- (dd <= nbrThresh)
nn[1:3]
```

```
##     1     2     3
##  TRUE FALSE FALSE
```

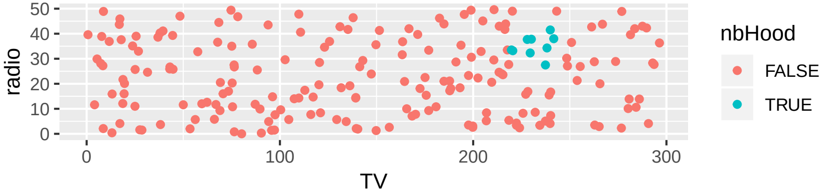
- Encapsulate computation in a function.

```r
nbhd<-function(index,dat,K){
  dd <- as.matrix(dist(dat))[,index]
  nbrThresh <- sort(dd)[K]
  return(factor(dd <= nbrThresh))
}
advert <- mutate(advert,nbHood = nbhd(1,Xdat,K=9))
```

```r
advert[1,]
```

```
##      TV radio newspaper sales     cTV cRadio nbHood
## 1 230.1  37.8      69.2  22.1 83.0575 14.536   TRUE
```

```r
ggplot(advert,aes(x=TV,y=radio,color=nbHood)) + geom_point()
```



KNN Prediction for First City

```
advert[1,]
```

```
##      TV radio newspaper sales     cTV cRadio nbHood
## 1 230.1  37.8      69.2  22.1 83.0575 14.536   TRUE
```

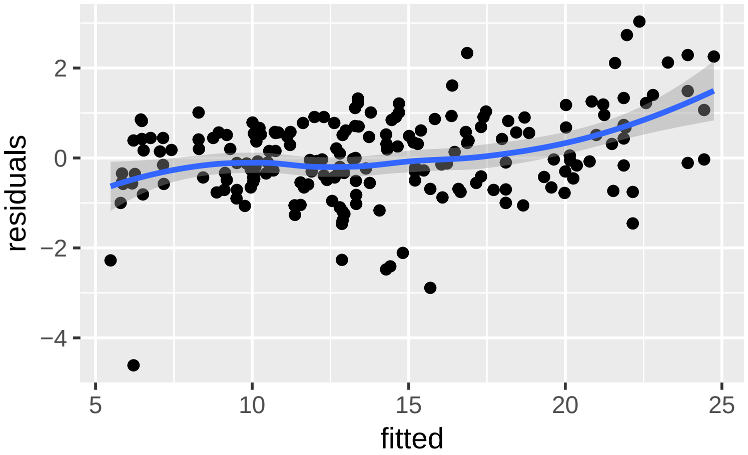
```
with(advert,mean(sales[nbHood==TRUE]))
```

```
## [1] 20.84444
```

KNN Predictions for Advertising Data

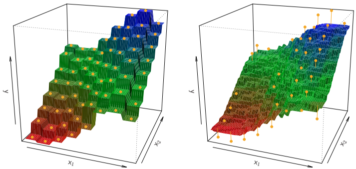
```
n <- nrow(advert)
K <- 9
KNNpred <- rep(NA,n)
for(i in 1:n) {
  advert <- mutate(advert,nbHood=nbhd(i,Xdat,K))
  KNNpred[i] <- with(advert,mean(sales[nbHood==TRUE]))
}
```

```r
mutate(advert,fitted = KNNpred, residuals=sales-KNNpred) %>%
  ggplot(aes(x=fitted,y=residuals)) + geom_point() +
  geom_smooth()
```



Example from Text

- Plots of $\hat{f}(X)$ using KNN on 64 observations with $K = 1$ (left panel) and $K = 9$ (right panel).



- For $K = 1$ the KNN interpolates and for $K = 9$ it smooths.
  - Which is best? The one that gives the best test set error rates
  - We will discuss methods for esimating the test set error rate, but for now we simply break the advertising data into a training and test set.

Test Set Predictions from KNN

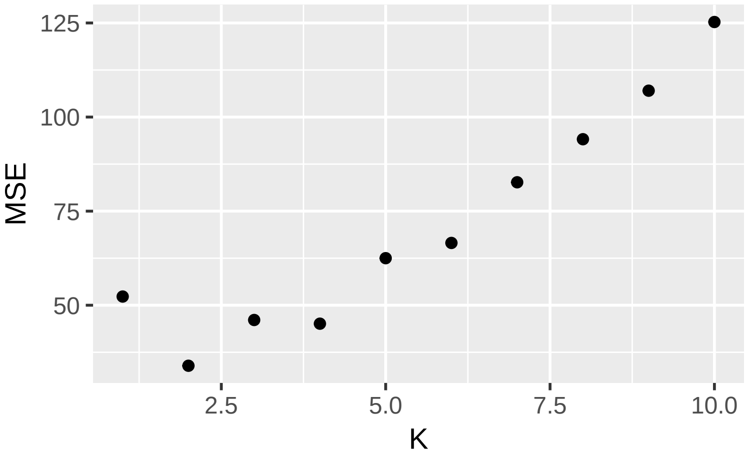
- For a prediction point x_0, find the neighborhood $\mathcal{N}_0$ of the K closest points in the training set, and take

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in \mathcal{N}_0} y_i$$

KNN on Advertising Data

```
library(caret) # install.packages("caret") if not already done
Y <- advert$sales
trainset <- (1:(.8*n))
trainX <- Xdat[trainset,]
trainY <- Y[trainset]
testset <- ((.8*n+1):n)
testX <- Xdat[testset,]
testY <- Y[testset]
maxK <- 10
testMSE <- rep(NA,maxK)
for(k in 1:maxK) {
  fit <- knnreg(trainX, trainY, k)
  testMSE[k] <- sum((testY - predict(fit, testX))^2)
}
```

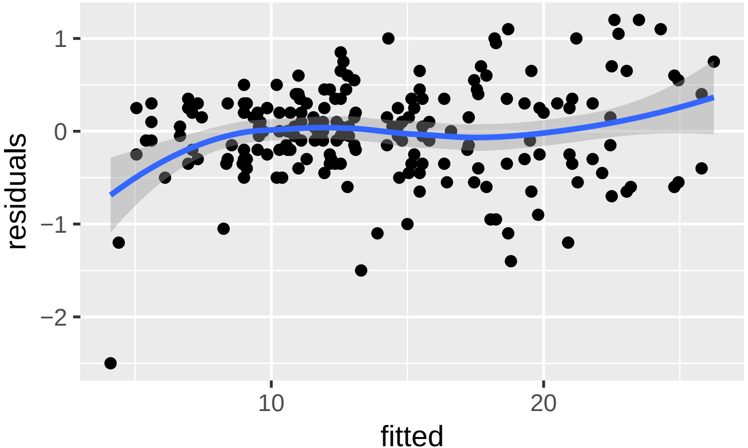
```r
data.frame(MSE = testMSE, K=1:maxK) %>%
  ggplot(aes(x=K,y=MSE)) + geom_point()
```



- According to the current test set, $K = 2$ gives the best test set MSE.

KNN with $K = 2$ on Advertising Data

```
K <- 2
for(i in 1:n) {
  advert <- mutate(advert,nbHood=nbhd(i,Xdat,K))
  KNNpred[i] <- with(advert,mean(sales[nbHood==TRUE]))
}
mutate(advert,fitted=KNNpred,residuals=sales-KNNpred) %>%
  ggplot(aes(x=fitted,y=residuals)) + geom_point() + geom_smooth()
```

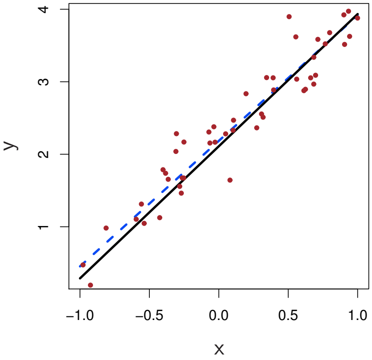


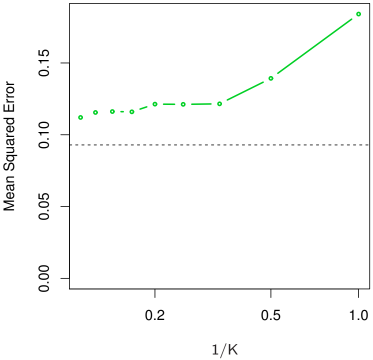
Comparison of KNN to Linear Regression

- The text compares KNN to linear regression (with main effects) under different relationships (linear or non-linear) and different numbers of predictors p.

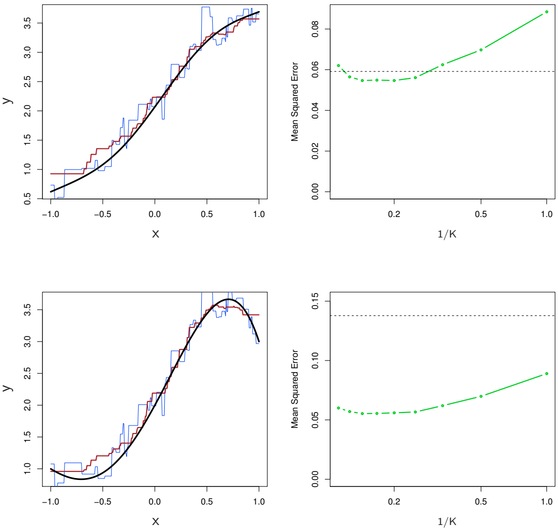
Linear f, $p = 1$.

- When the true f is linear, linear regression MSE (dotted line) is slightly better than KNN MSE (green).



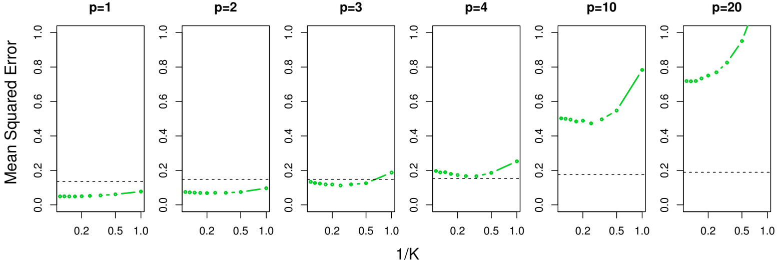


Non-Linear f, $p = 1$



Non-Linear f, Varying p

- KNN is better for small p but worse for large p



- When $p = 20$, for example, the "nearest" neighbors are not very near, and so do a poor job of predicting f.
  - This phenomenon is known as the "curse of dimensionality".