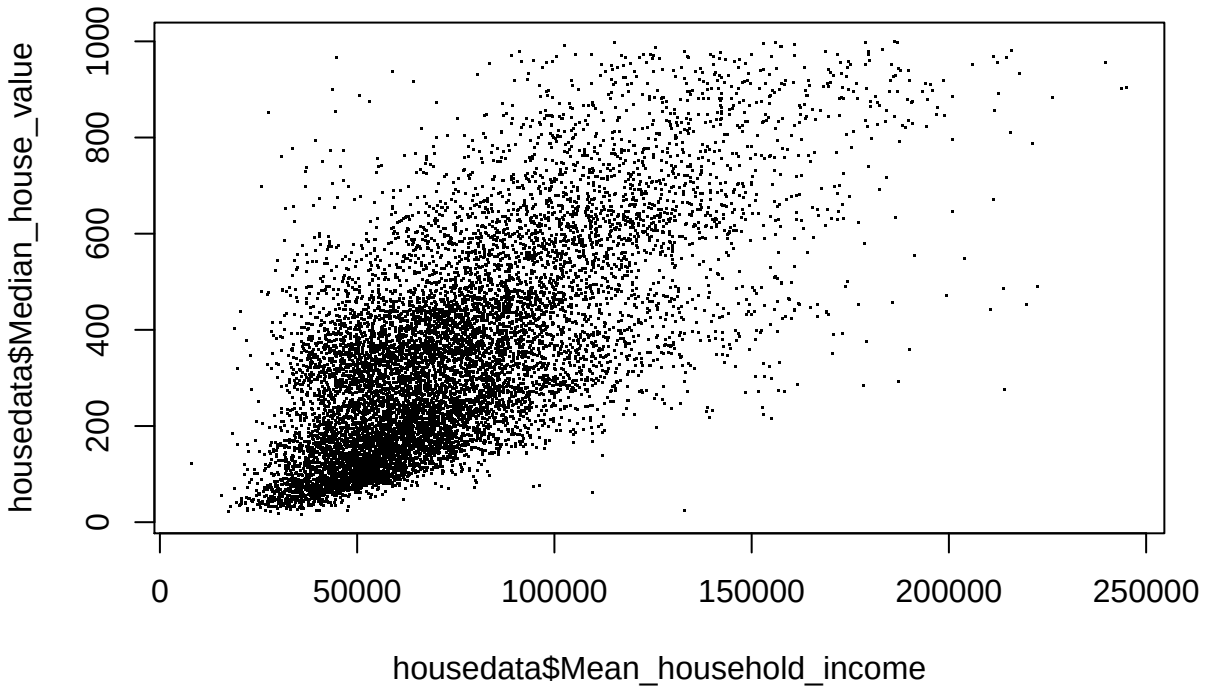
---
title: "36-402 Homework 6"
author: "Jacky Liu"
date: "2/12/21"
output: pdf_document

Problem 1

Problem 1 (a)

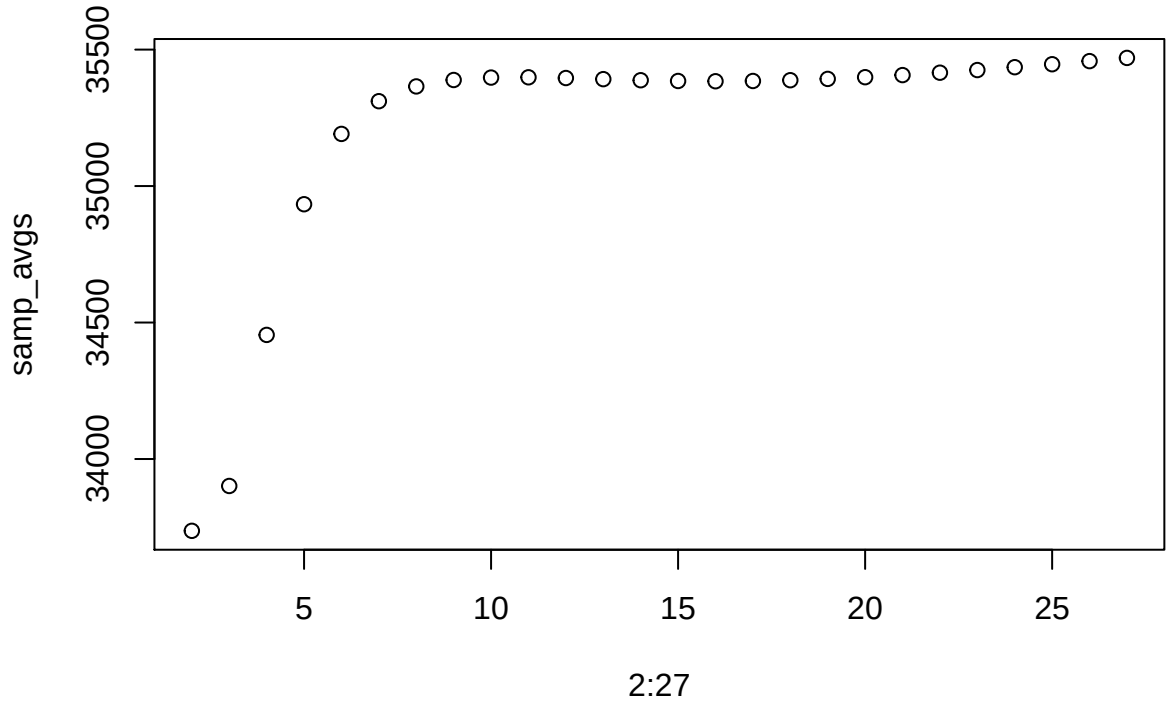
```
housetrain=read.csv("housetrain.csv")
housetest=read.csv("housetest.csv")
housedata=rbind(housetrain, housetest)
plot(housedata$Mean_household_income, housedata$Median_house_value, pch=".")
```



Problem 1 (b)

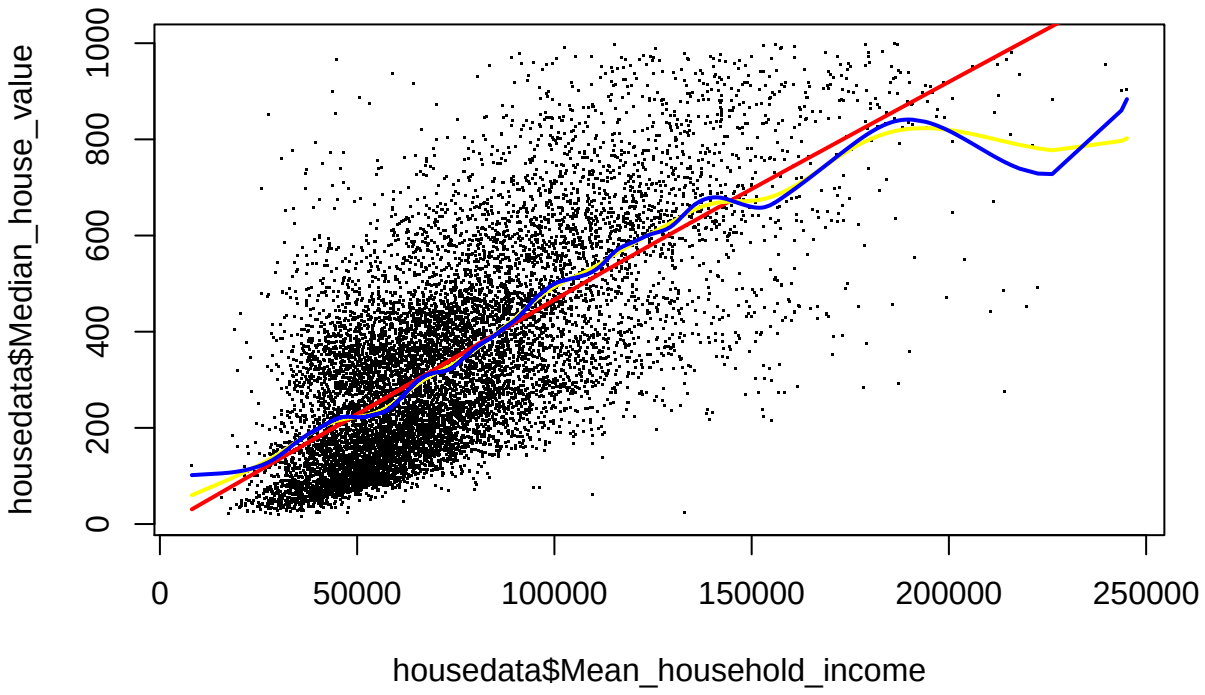
```
prederr = matrix(nrow=5, ncol=26)
for(i in 2:27){
  set.seed(69)
  samp <- sample(rep(1:5, 2121), replace = FALSE)
```

```
  for (k in 1:5) {
    testd <- housedata[samp == k, ]
    traind <- housedata[!(samp == k), ]
    splinemod <- smooth.spline(traind$Mean_household_income, traind$Median_house_value, df=i)
    splinemodpreds <- smooth.spline(testd$Mean_household_income, testd$Median_house_value, df=i)
    prederr[k,i-1] = mean((splinemod$y - splinemodpreds$y)^2)
  }
}
samp_avgs = apply(prederr, 2, mean)
std_errs = apply(prederr, 2, sd)/sqrt(5)
plot(2:27, samp_avgs)
```



DF of 2 gives the lowest average error.

```
mod2 <- smooth.spline(traind$Mean_household_income, traind$Median_house_value, df=2)
mod15 <- smooth.spline(traind$Mean_household_income, traind$Median_house_value, df=15)
mod25 <- smooth.spline(traind$Mean_household_income, traind$Median_house_value, df=25)
plot(housedata$Mean_household_income, housedata$Median_house_value, pch=".")
lines(mod2$x, mod2$y, col="red", lwd=2)
lines(mod15$x, mod15$y, col="yellow", lwd=2)
lines(mod25$x, mod25$y, col="blue", lwd=2)
```



Looking at the plot we made manually above showing lines of df=2,15,25, we can see that df=2 is in fact significantly better fitting th epoints than 15 and 25, both of which seem non-resistant to noise and overfitted. Therefore, this comparison inspires confidence that we have found the best df.

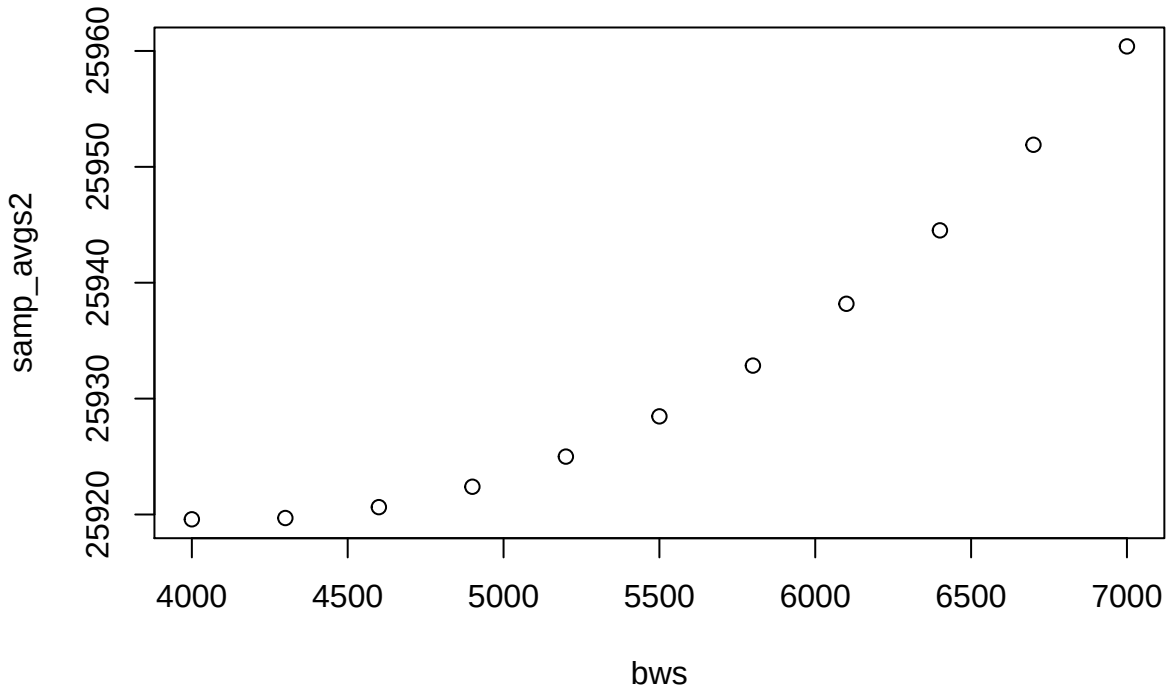
Problem 1 (c)

```
library(np)
```

```
## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-10)
## [vignette("np_faq",package="np") provides answers to frequently asked questions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]
```

```
bws = seq(from=4000,to=7000, by=300)
prederr2 = matrix(nrow=5, ncol=length(bws))
count = 1
for(i in bws){
  set.seed(69)
  samp <- sample(rep(1:5, 2121), replace = FALSE)
  for (k in 1:5) {
    testd <- housedata[samp == k, ]
    traind <- housedata[!(samp == k), ]
    model = npreg(Median_house_value ~ Median_household_income,
                  data=traind, newdata=testd, bws = i)
```

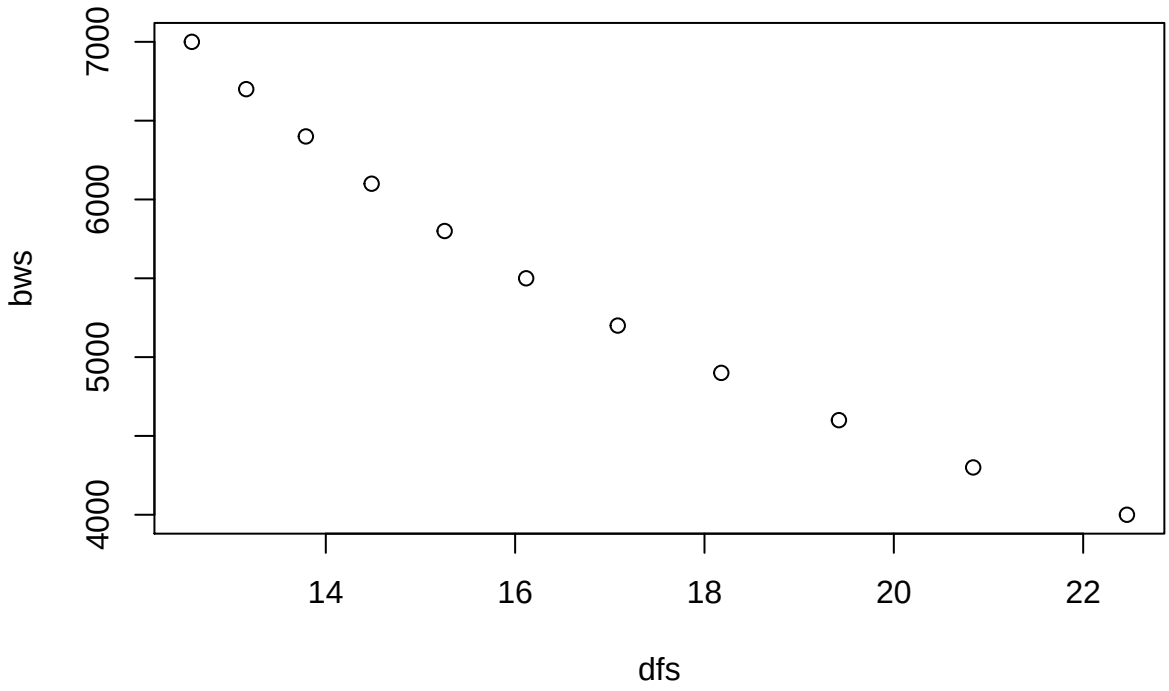
```
    prederr2[k,count] = mean((model$mean - testd$Median_house_value)^2)
  }
  count = count + 1
}
samp_avgs2 = apply(prederr2, 2, mean)
std_errs2 = apply(prederr2, 2, sd)/sqrt(5)
plot(bws, samp_avgs2)
```



Problem 1 (d)

```
df = function(h, x){
  total = 0
  for(i in 1:length(x)){
    t = 0
    for(j in 1:length(x)){
      t = t + dnorm((x[i]-x[j])/h)
    }
    total = total + 1/t
  }
  return(dnorm(0)*total)
}
count = 1
dfs = vector(length = length(bws))
for(h in bws){
  dfs[count] = df(h, housedata$Mean_household_income)
  count = count + 1
```

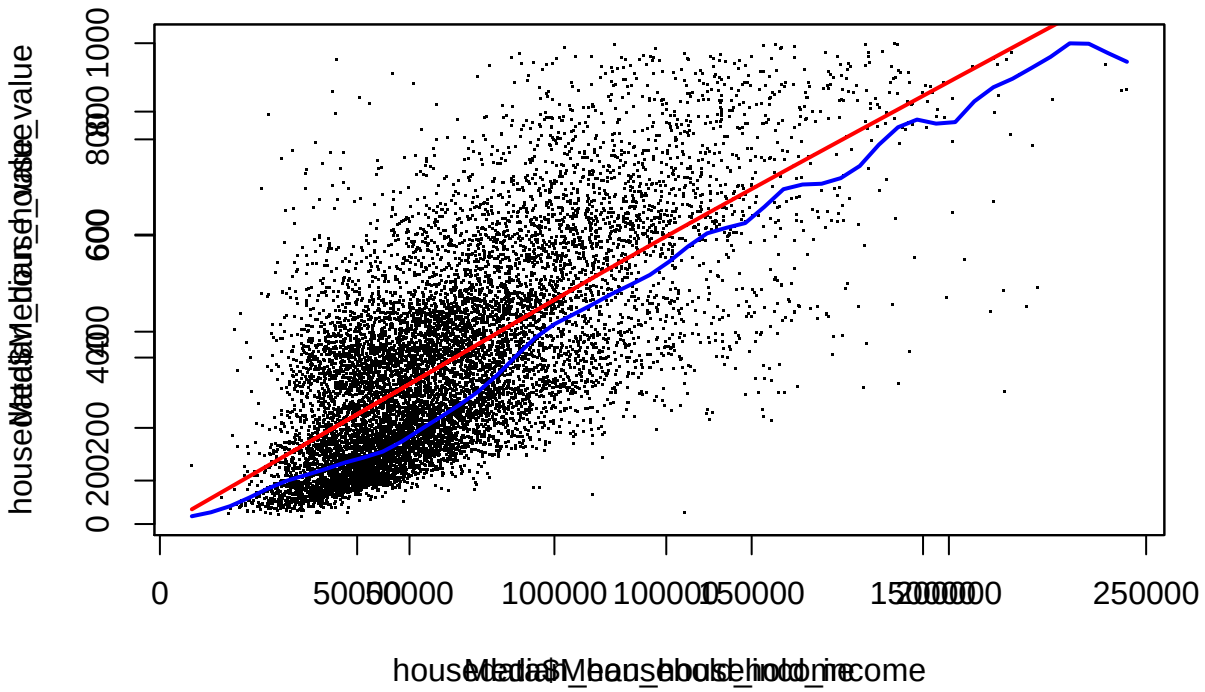
```
}
plot(dfs, bws)
```



This does not inspire confidence that we can tell which method provides better out-of-sample prediction. The dfs shown here only ranges from 14-22 whereas we used a df=2 value with a bws=4000 value. We can see that as dfs increases, the value of bws decreases as the only trend in this plot but it doesn't provide evidence that one method is better than another.

Problem 1 (e)

```
plot(housedata$Mean_household_income, housedata$Median_house_value, pch=".")
bestspline <- smooth.spline(traind$Mean_household_income, traind$Median_house_value, df=2)
lines(bestspline$x, bestspline$y, col="red", lwd=2)
bestkernel = npreg(Median_house_value ~ Median_household_income,
                   data=traind, newdata=testd, bws = 4000)
par(new=TRUE)
plot(bestkernel, col="blue", lwd=2)
```



From looking at the plot above, spline seems to fit the data better than nonparametric. While npreg seems to go towards the denser areas more as evident by the dip near 5000, it also seems less resistant to noise as we can see from the rugged tail at the end. A reason these two fitted curves behave so differently is because these two types of regression are implemented differently; spline yields a piecewise continuous function composed of many polynomials to model the data while np gives a single polynomial that models the entire dataset.

Problem 2

Problem 2 (a)

I have no known conflicts.

Problem 2 (b)

I successfully knit Rmd to PDF.

Problem 2 (c)

I've read the sample reports.