Hwk #1: Regression

For this homework we will use NHANES data that exists in a package for R.

NHANES consists of survey data collected by the US National Center for Health Statistics (NCHS) which has conducted a series of health and nutrition surveys since the early 1960's. Since 1999 approximately 5,000 individuals of all ages are interviewed in their homes every year and complete the health examination component of the survey. The health examination is conducted in a mobile examination center (MEC).

Note that there is the following warning on the NHANES website: "For NHANES datasets, the use of sampling weights and sample design variables is recommended for all analyses because the sample design is a clustered design and incorporates differential probabilities of selection. If you fail to account for the sampling parameters, you may obtain biased estimates and overstate significance levels."

For this homework, please ignore this warning and just apply our analyses to the data as if they were randomly sampled! We will be using the data called `NHANESraw`.

For questions that ask for your comments, it suffices to answer with one or two sentences in each case.

Data Preparation

1. Install the package `NHANES` into R, load the `NHANES` package, and then run the command `data(NHANES)` which will load the NHANES data. Type `?NHANES` and read about the dataset.

```
library(NHANES)
NHANESraw <- data("NHANES")
?NHANES
```

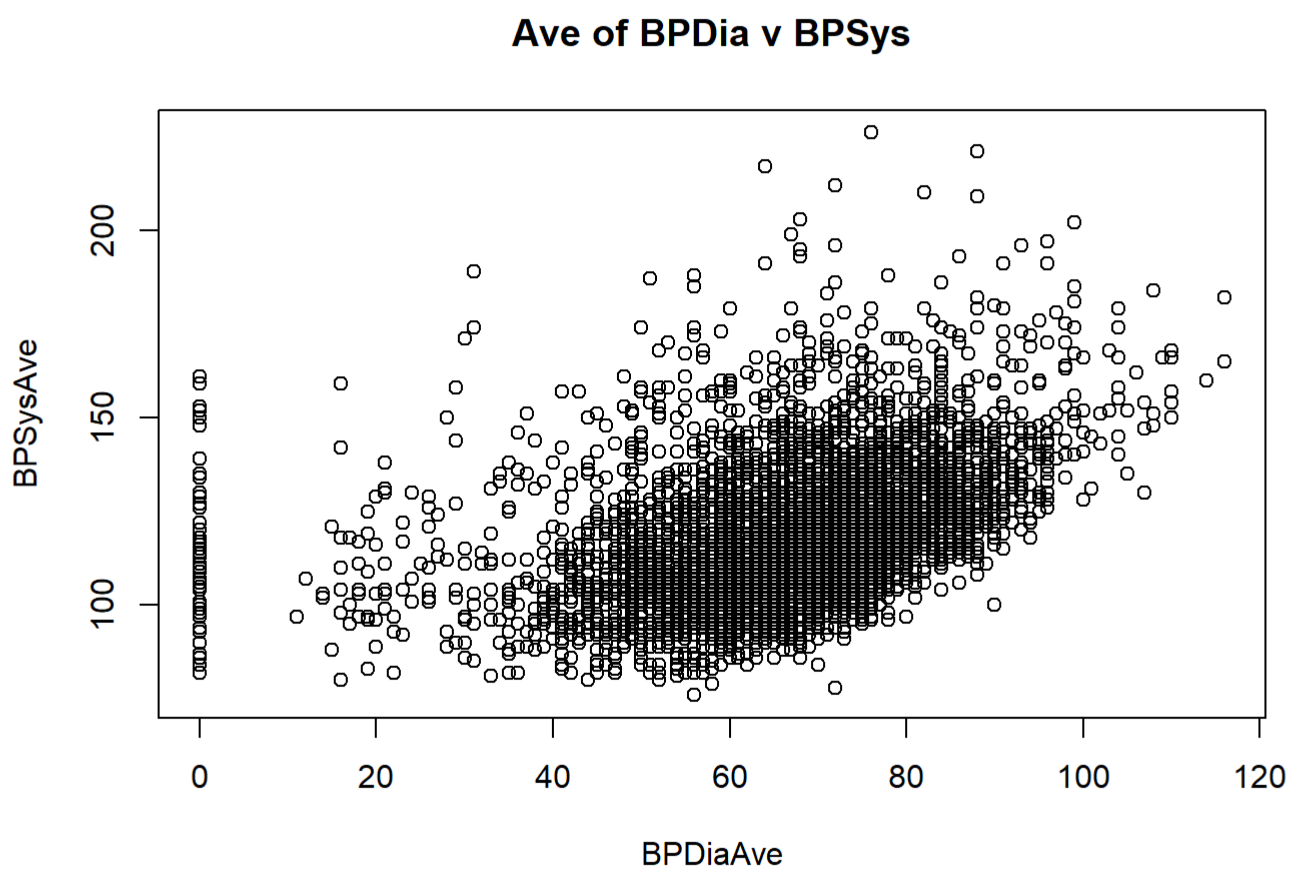
```
## starting httpd help server ... done
```

2. Make an object `nhanes` that is a subset version `NHANESraw` that does not include any missing data for `Diabetes`, `BPSysAve`, `BPDiaAve`

```
nhanes <- subset(NHANES, !is.na(NHANES$Diabetes)& !is.na(NHANES$BPSysAve)& !is.na(NHANES$BPDiaAv
e))
```

3. (1 point) Plot `BPSysAve` against `BPDiaAve`. Comment on what is going on. Further subset the data such the observations with `BPDiaAve` equal to zero are removed.

```
plot(nhanes$BPDiaAve, nhanes$BPSysAve, xlab="BPDiaAve", ylab="BPSysAve", main="Ave of BPDia v BP
Sys")
```



```
nhanes <- subset(nhanes, nhanes$BPDiaAve !=0)
```

Comment: Higher BPDiaAVe is higher BPSysAve

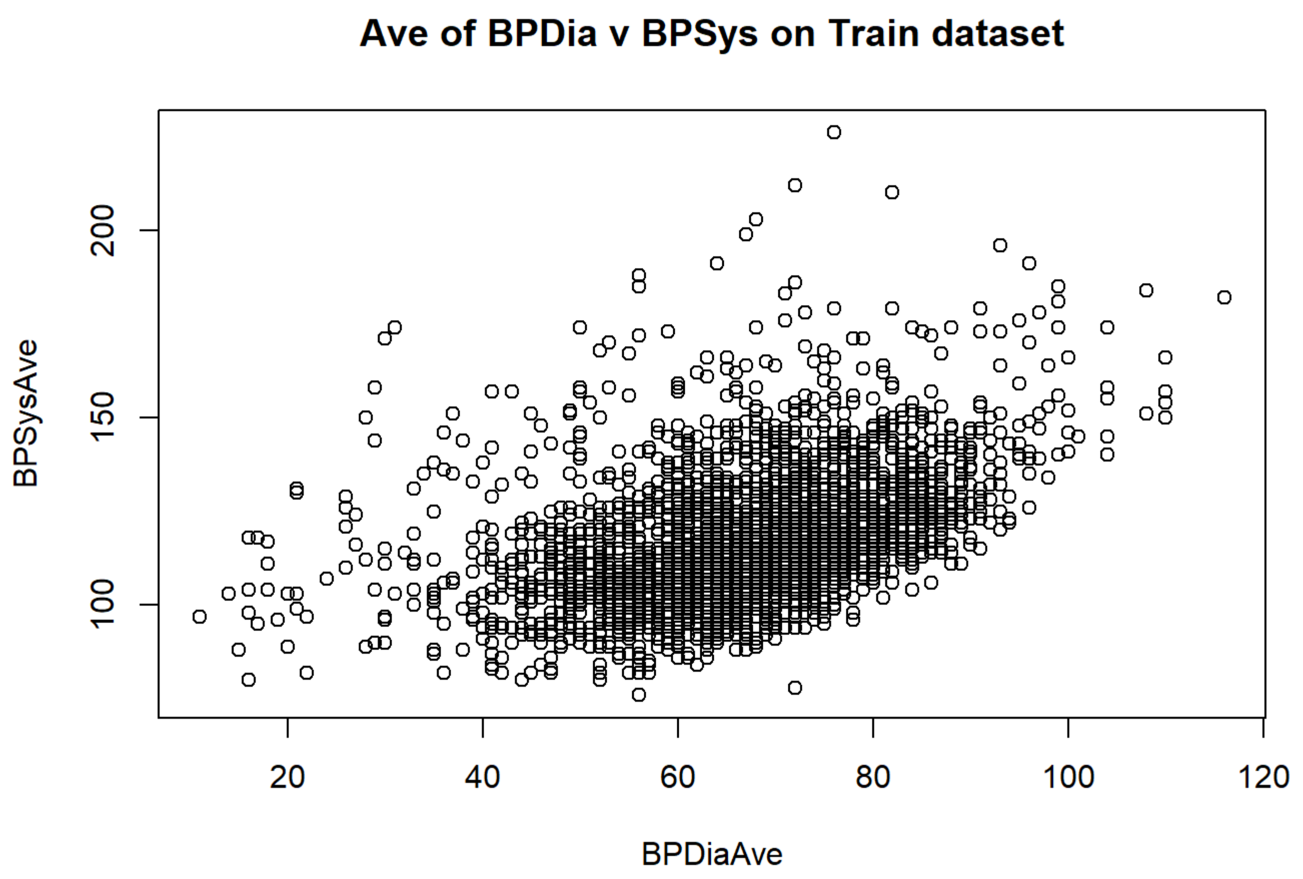
4. (1 point) Make an object `nhanes09` that is a subset of `nhanes` to only the 2009_10 data. This will be your training dataset. Also make an object `nhanes11` that is a subset of `nhanes` to only the 2011_12 data. This will be your test dataset.

```
nhanes09 <- subset(nhanes, nhanes$SurveyYr=='2009_10')
nhanes11 <- subset(nhanes, nhanes$SurveyYr=='2011_12')
```

Linear regression

5. (1 point) Plot `BPSysAve` against `BPDiaAve` for the training data.

```
plot(nhanes09$BPDiaAve, nhanes09$BPSysAve,xlab="BPDiaAve", ylab="BPSysAve", main="Ave of BPDia v
BPSys on Train dataset")
```



6. (2 points) Fit a linear model using the training data with `BPSysAve` as outcome and `BPDiaAve` as the single predictor.

```
mdl1 = lm(BPSysAve ~ BPDiaAve, data = nhanes09)
```

7. (1 point) Use the `summary` command to examine the resulting fitted model.

```
summary(mdl1)
```

```
##
## Call:
## lm(formula = BPSysAve ~ BPDiaAve, data = nhanes09)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -42.223 -10.757  -2.610   7.398 103.625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 81.49840    1.25745   64.81   <2e-16 ***
## BPDiaAve     0.53785    0.01838   29.27   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.54 on 4233 degrees of freedom
## Multiple R-squared:  0.1683, Adjusted R-squared:  0.1681
## F-statistic: 856.4 on 1 and 4233 DF,  p-value: < 2.2e-16
```

8. (1 point) Generate 95% confidence intervals for the parameters of the fitted model.

```
confint(mdl1, level=0.95)
```

```
##                  2.5 %     97.5 %
## (Intercept) 79.0331444 83.9636490
## BPDiaAve     0.5018154  0.5738794
```

9. (1 point) Also, generate 99% confidence intervals for the parameters of the fitted model.

```
confint(mdl1, level=0.99)
```

```
##                  0.5 %     99.5 %
## (Intercept) 78.2579713 84.7388221
## BPDiaAve     0.4904855  0.5852093
```

10. (2 points) Comment on the difference between the 95% and 99% confidence intervals and whether or not the difference is what you would expect.

Comment: 99% confidence intervals(CI) are wider than 95% CI. This is not difference from my expectation because the possibility of 99% CI that include true mean values is higher than 95% and it means that the range is wider.

11. (3 points) Now fit models with quadratic and cubic terms in the predictor `BPDiaAve` in addition to the linear term. Look at the output of each model with summary and generate 95% confidence intervals for each.

```
# Quadratic terms
mdl_quadratic <- lm(BPSysAve ~ BPDiaAve + I(BPDiaAve^2), data = nhanes09)
summary(mdl_quadratic)
```

```
##
## Call:
## lm(formula = BPSysAve ~ BPDiaAve + I(BPDiaAve^2), data = nhanes09)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.595 -10.152  -2.476   7.310 104.181
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   136.184661   3.313326   41.10   <2e-16 ***
## BPDiaAve       -1.239112   0.101741  -12.18   <2e-16 ***
## I(BPDiaAve^2)   0.013817   0.000779   17.74   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.99 on 4232 degrees of freedom
## Multiple R-squared:  0.2258, Adjusted R-squared:  0.2255
## F-statistic: 617.2 on 2 and 4232 DF,  p-value: < 2.2e-16
```

```
confint(mdl_quadratic, level=0.95)
```

```
##                       2.5 %       97.5 %
## (Intercept)   129.68880318 142.68051851
## BPDiaAve       -1.43857837  -1.03964508
## I(BPDiaAve^2)   0.01228968   0.01534414
```

```
# Cubic terms
mdl_cubic <- lm(BPSysAve ~ BPDiaAve + I(BPDiaAve^2)+I(BPDiaAve^3), data= nhanes09)
summary(mdl_cubic)
```

```
##
## Call:
## lm(formula = BPSysAve ~ BPDiaAve + I(BPDiaAve^2) + I(BPDiaAve^3),
##     data = nhanes09)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.385 -10.152  -2.401   7.327 104.459
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.256e+02  6.500e+00  19.324   <2e-16 ***
## BPDiaAve      -6.534e-01  3.256e-01  -2.007   0.0449 *
## I(BPDiaAve^2)  3.878e-03  5.307e-03   0.731   0.4650
## I(BPDiaAve^3)  5.287e-05  2.792e-05   1.893   0.0584 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.99 on 4231 degrees of freedom
## Multiple R-squared:  0.2265, Adjusted R-squared:  0.2259
## F-statistic: 412.9 on 3 and 4231 DF,  p-value: < 2.2e-16
```

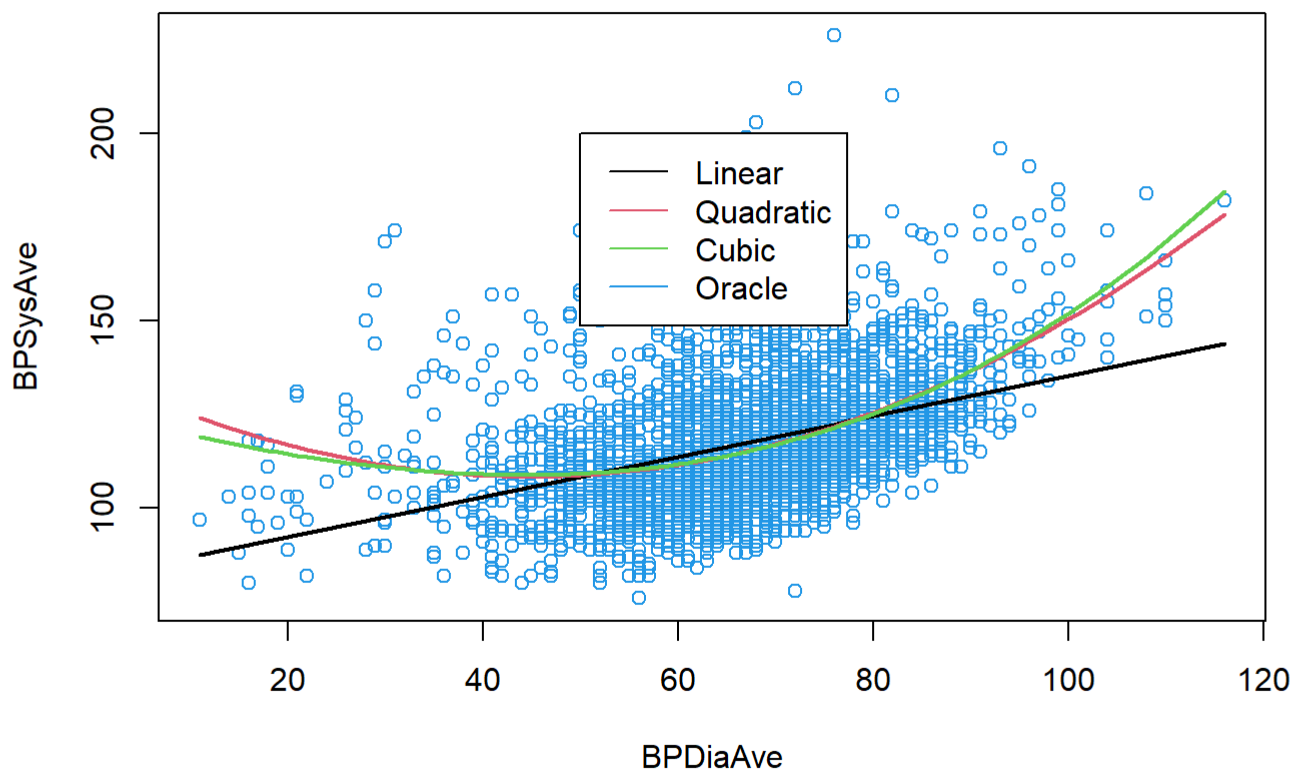
```
confint(mdl_cubic)
```

```
##                       2.5 %        97.5 %
## (Intercept)    1.128534e+02  1.383388e+02
## BPDiaAve      -1.291810e+00 -1.500903e-02
## I(BPDiaAve^2) -6.526290e-03  1.428177e-02
## I(BPDiaAve^3) -1.873471e-06  1.076045e-04
```

12. (2 points) Plot the training data along with the linear, quadratic and cubic fit lines in different colors.

```
new_dat <- data.frame(BPDiaAve= nhanes09$BPDiaAve)
lm_pred_dat <- data.frame(
  new_dat,
  lm1 = predict(mdl1, newdata=nhanes09),
  lm2 = predict(mdl_quadratic, newdata=nhanes09),
  lm3 = predict(mdl_cubic, newdata=nhanes09),
  oracle= nhanes09$BPSysAve
)
lm_pred_dat <- lm_pred_dat[order(lm_pred_dat$BPDiaAve),]

plot(lm_pred_dat$BPDiaAve,lm_pred_dat$oracle,col=4,type= 'p', xlab="BPDiaAve",ylab="BPSysAve")
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm1,type="l",col=1, lwd=2)
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm2,type="l",col=2, lwd=2)
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm3,type="l",col=3, lwd=2)
legend(50,200,c("Linear","Quadratic","Cubic","Oracle"),lty=rep(1,4),col=(1:4))
```



13. (1 point) Which would be your preferred model based on the visual fits? *I think that the Quadratic model is a better model than the other models. Because the Quadratic model is more clearly fitted than the Linear model. The line of Quadratic and Cubic models are similar but the Quadratic model is simpler than the Cubic model. Therefore, I will choose the Quadratic model.*

14. (3 points) Perform an anova test comparing the 3 models. Does the result seem in line with what you were expecting from the visual fits? Why/why not?

```
anova(mdl1, mdl_quadratic, mdl_cubic)
```

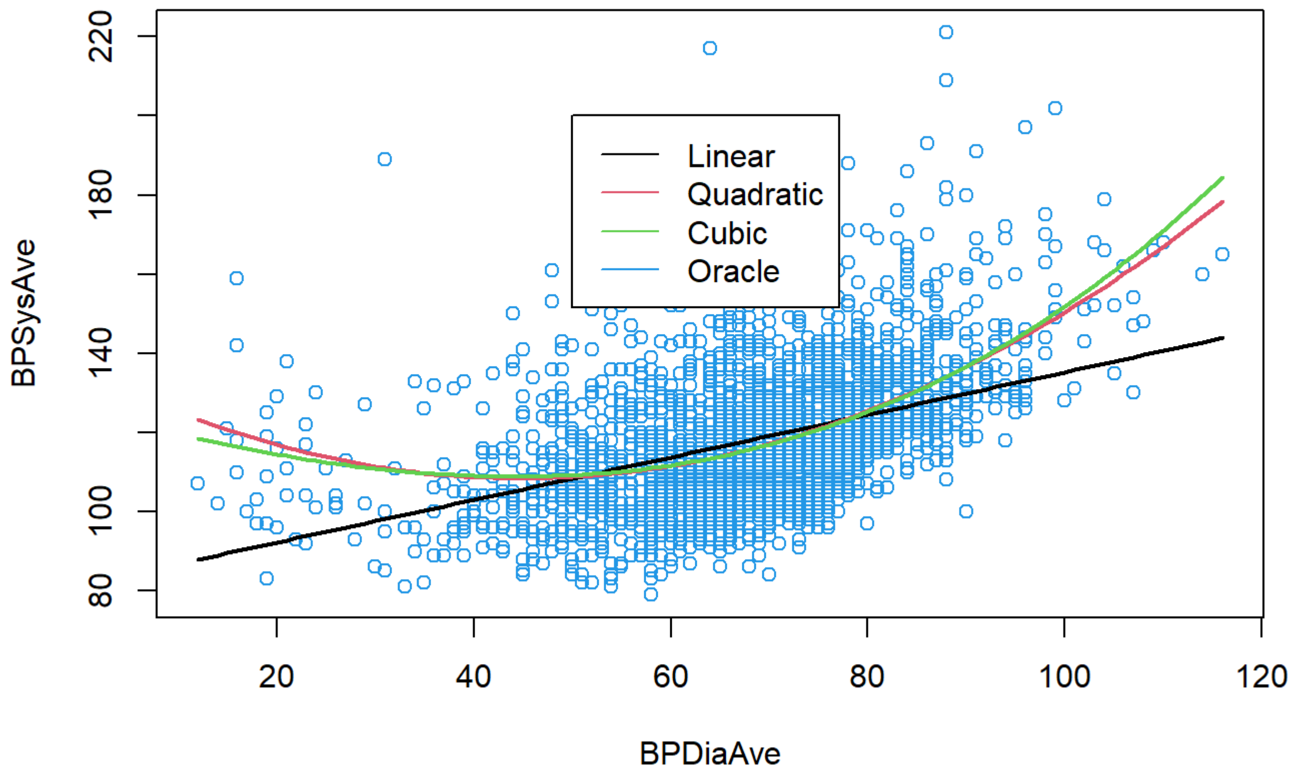
```
## Analysis of Variance Table
##
## Model 1: BPSysAve ~ BPDiaAve
## Model 2: BPSysAve ~ BPDiaAve + I(BPDiaAve^2)
## Model 3: BPSysAve ~ BPDiaAve + I(BPDiaAve^2) + I(BPDiaAve^3)
##   Res.Df     RSS Df Sum of Sq        F  Pr(>F)
## 1   4233 1022253
## 2   4232  951518  1     70734 314.7914 < 2e-16 ***
## 3   4231  950713  1       806   3.5851 0.05837 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This result is similar to the visual fits. Because the Quadratic model is significantly better than the Linear model. However, the Cubic model is not significantly different from the quadratic model. This result indicates that the Quadratic model is better than the other models.

15. (1 point) Now plot `BPSysAve` against `BPDiaAve` for the test data and overlay the fitted linear, quadratic, and cubic models.

```
new_dat <- data.frame(BPDiaAve= nhanes11$BPDiaAve)
lm_pred_dat <- data.frame(
  new_dat,
  lm1 = predict(mdl1, newdata=nhanes11),
  lm2 = predict(mdl_quadratic, newdata=nhanes11),
  lm3 = predict(mdl_cubic, newdata=nhanes11),
  oracle= nhanes11$BPSysAve
)
lm_pred_dat <- lm_pred_dat[order(lm_pred_dat$BPDiaAve),]


plot(lm_pred_dat$BPDiaAve,lm_pred_dat$oracle,col=4,xlab="BPDiaAve",ylab="BPSysAve")
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm1,type="l",col=1,lwd=2)
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm2,type="l",col=2,lwd=2)
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm3,type="l",col=3,lwd=2)
legend(50,200,c("Linear","Quadratic","Cubic","Oracle"),lty=rep(1,4),col=(1:4))
```



16. (3 points) Does this change your opinion at all about which is the best fit? Why/why not? *I do not change my opinion. I think the Quadratic model is the best fitted model because the model fits the test data as well as train data and the Quadratic model is significantly different from the Linear model and simpler than the Cubic model.*

Smoothing kernels:

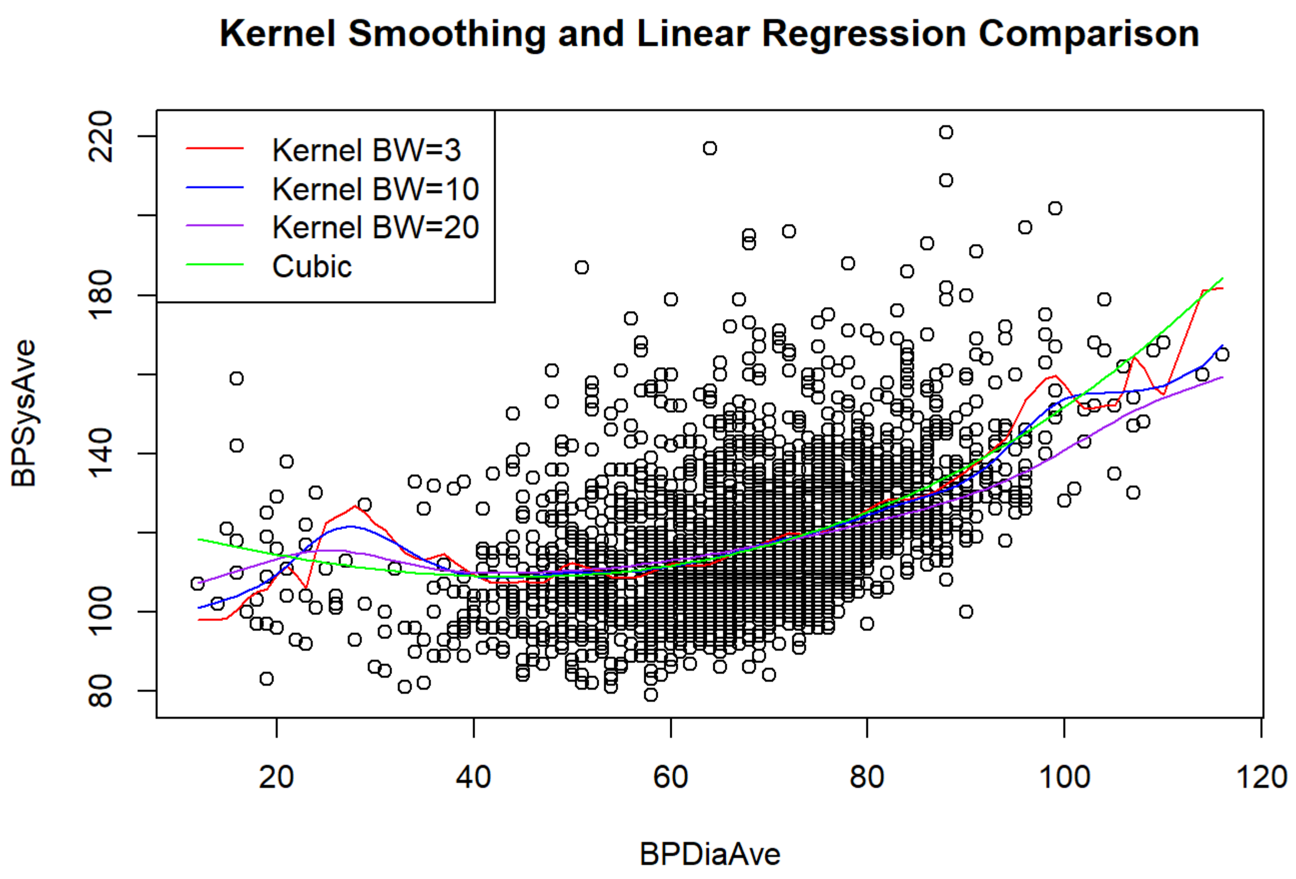
17. (3 points) Fit a nearest neighbors curve with `ksmooth` using the "normal" kernel to the `nhanes09` data with bandwidths of 3, 10, and 20.

```
ksmooth_3 <- ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 3)
ksmooth_10 <- ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 10)
ksmooth_20 <- ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 20)
```

18. (2 points) Plot the test data as well as the fitted curves from kernel smoothing and the best model fitted in the previous section using linear regression. Use different colors for each model.

```
ksmooth_3 = ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 3, x.po
ints = nhanes11$BPDiaAve)
ksmooth_10 = ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 10, x.
points = nhanes11$BPDiaAve)
ksmooth_20 = ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 20, x.
points = nhanes11$BPDiaAve)

plot(nhanes11$BPDiaAve, nhanes11$BPSysAve, col='black', xlab="BPDiaAve", ylab="BPSysAve", main
="Kernel Smoothing and Linear Regression Comparison")
lines(ksmooth_3$x, ksmooth_3$y, col='red', type='l')
lines(ksmooth_10$x, ksmooth_10$y, col='blue', type='l')
lines(ksmooth_20$x, ksmooth_20$y, col='purple', type='l')
lines(lm_pred_dat$BPDiaAve,lm_pred_dat$lm3,type="l",col='green')
legend("topleft",
       legend=c("Kernel BW=3", "Kernel BW=10", "Kernel BW=20", "Cubic"),
       col=c('red', 'blue', 'purple', 'green'),
       lty=1)
```



19. (1 point) Based on the above results, which model would you pick? In the next section, we'll evaluate the model based on its test error.

I will pick the green model (bandwidth=20).

Evaluating error on a test set

20. (5 points) Evaluate the mean squared error of the fitted models from the "Linear Regression" section on the test data. Also provide standard errors.

```
# MSE function
mse <- function(actual, estimated){
  mean((actual-estimated)^2)
}

# Calculate each MSE
actual <- nhanes11$BPSysAve

mse_linear <- mse(actual,predict(mdl1, newdata=nhanes11))
mse_quadratic <- mse(actual, predict(mdl_quadratic, newdata=nhanes11))
mse_cubic <- mse(actual, predict(mdl_cubic, newdata=nhanes11))



# Calculate SE
se_linear <- sqrt(mse_linear)
se_quadratic <- sqrt(mse_quadratic)
se_cubic <- sqrt(mse_cubic)

# Results
cat("Linear model MSE:",mse_linear,"SE:",se_linear,"\n")
```

```
## Linear model MSE: 242.2397 SE: 15.56405
```

```
cat("Quadratic Model MSE:", mse_quadratic, "SE:", se_quadratic, "\n")
```

```
## Quadratic Model MSE: 229.3377 SE: 15.1439
```

```
cat("Cubic Model MSE:", mse_cubic, "SE:", se_cubic, "\n")
```

```
## Cubic Model MSE: 229.1492 SE: 15.13767
```

21. (2 points) Using ANOVA, did we pick the model with the lowest mean squared error on the test data? Why do you think this happened? *No, we chose the different model based on ANOVA. This difference might be cased by the thing that the different trend between the training data and the test data. This is because ANOVA used only the training data, however, we use the training data and the test data to calculate MSE.*
22. (5 points) Evaluate the mean squared error of the fitted models from kernel smoothing on the test data.

```
ksmooth_3 = ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 3, x.po
ints = nhanes11$BPDiaAve)
ksmooth_10 = ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 10, x.
points = nhanes11$BPDiaAve)
ksmooth_20 = ksmooth(nhanes09$BPDiaAve, nhanes09$BPSysAve, kernel = 'normal', bandwidth = 20, x.
points = nhanes11$BPDiaAve)


# MSE
mse_bd3 <- mse(actual, ksmooth_3$y)
mse_bd10 <- mse(actual, ksmooth_10$y)
mse_bd20 <- mse(actual, ksmooth_20$y)
# SE
se_bd3 <- sqrt(mse_bd3)
se_bd10 <- sqrt(mse_bd10)
se_bd20 <- sqrt(mse_bd20)
# Results
cat("Bandwidth=3 is","MSE:",mse_bd3,"SE:",se_bd3,"\n")
```

```
## Bandwidth=3 is MSE: 379.7302 SE: 19.48667
```

```
cat("Bandwidth=10 is","MSE:",mse_bd10,"SE:",se_bd10,"\n")
```

```
## Bandwidth=10 is MSE: 364.2058 SE: 19.08418
```

```
cat("Bandwidth=20 is","MSE:",mse_bd20,"SE:",se_bd20,"\n")
```

```
## Bandwidth=20 is MSE: 337.2069 SE: 18.36319
```

23. (1 point) Among all the methods we tried, which one had the lowest test error?

```
mse_values <- c(linear = mse_linear, quadratic = mse_quadratic, cubic = mse_cubic,
                ksmooth_3 = mse_bd3, ksmooth_10 = mse_bd10, ksmooth_20 = mse_bd20)
cat('The best model is', names(which.min(mse_values)),'(MSE:', min(mse_values),')')
```

```
## The best model is cubic (MSE: 229.1492 )
```