Emprical Methods HW2

Emine Tasci

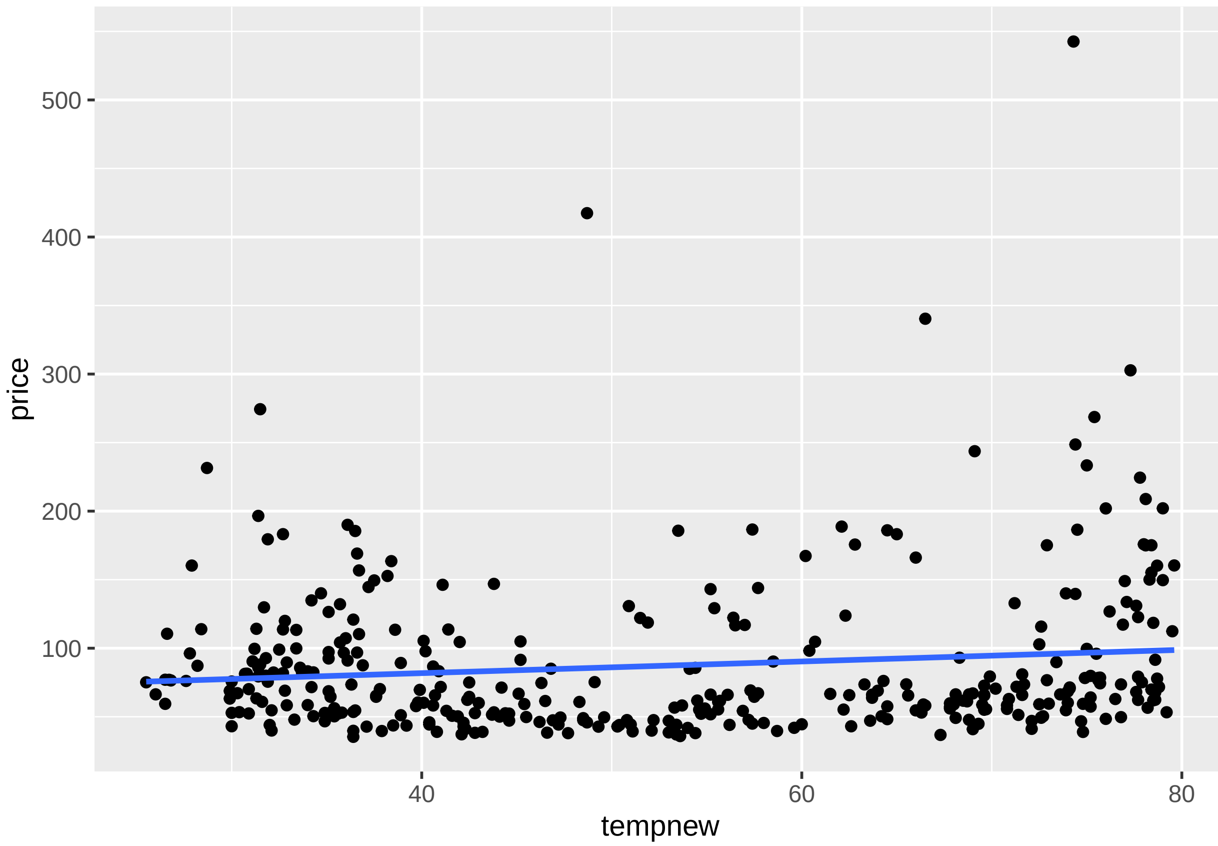
Part1

```
require(plyr)
```

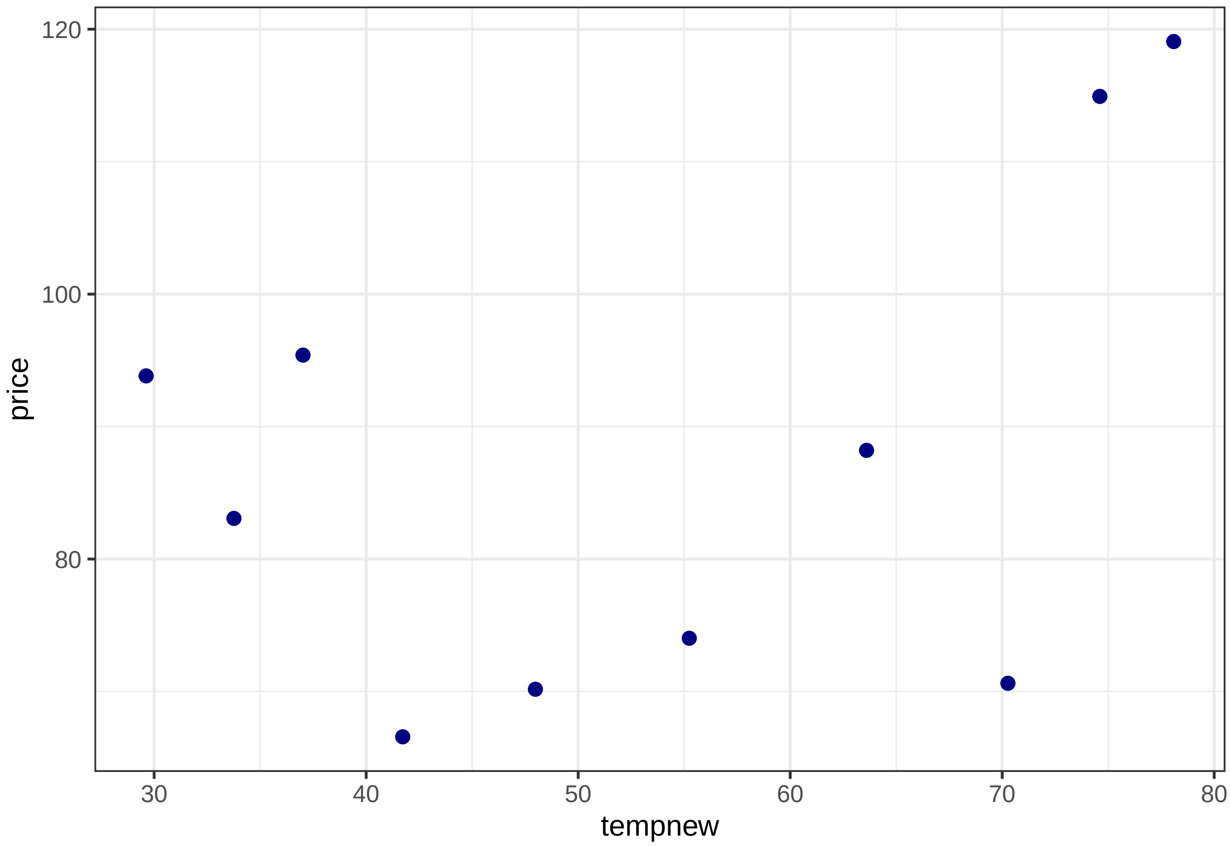
```
## Loading required package: plyr
```

```
library(binsreg)
library(ggplot2)

setwd("~/Desktop/BC/Spring 2021/Emprical Methods/HW2/")
data <- read.csv('nodelmp.csv')
names(data) <- c("hour","lmp","year","month", "day", "temp", "hrank")

hour <-data[,1]
lmp <-data[,2]
year <-data[,3]
month <- data[,4]
day <- data[,5]
temp <- data[,6]
hrank <- data[,7]

dataNew <- data[, c(2,4,5)]

maxPrice <- ddply(dataNew, ~month + day, summarise, max = max(lmp, na.rm = TRUE))
price <-maxPrice$max

tempnew<- c()
for(j in 1:length(price)){
for(i in 1:length(lmp)){
  if(lmp[i]==price[j]){tempnew[j]<- temp[i]}
  }
}

tp <- cbind(tempnew,price)
tp <- as.data.frame(tp)

ggplot(tp,aes(y=price, x=tempnew)) +
  geom_point()+
  geom_smooth(method=lm, se=FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
binsreg(y=price,x=tempnew)
```

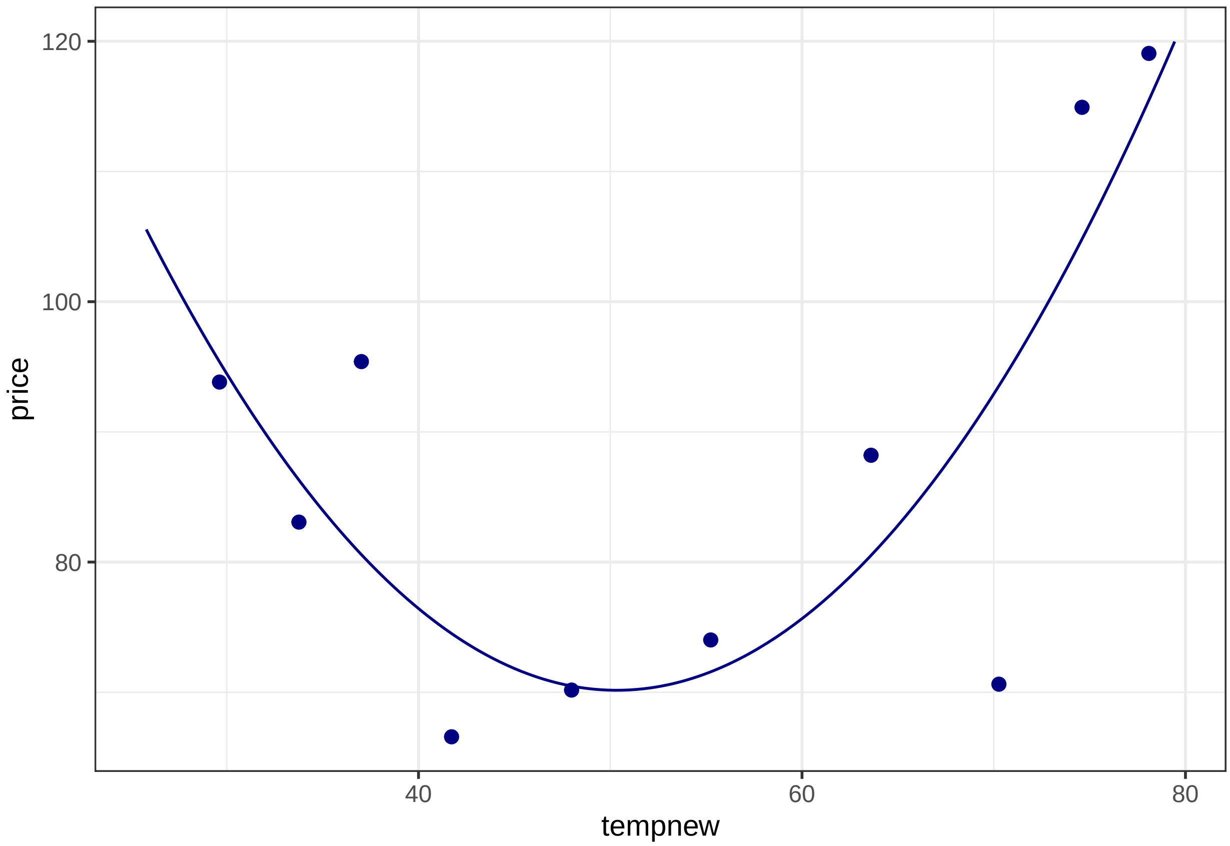



```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

The relationship does not look like linear.

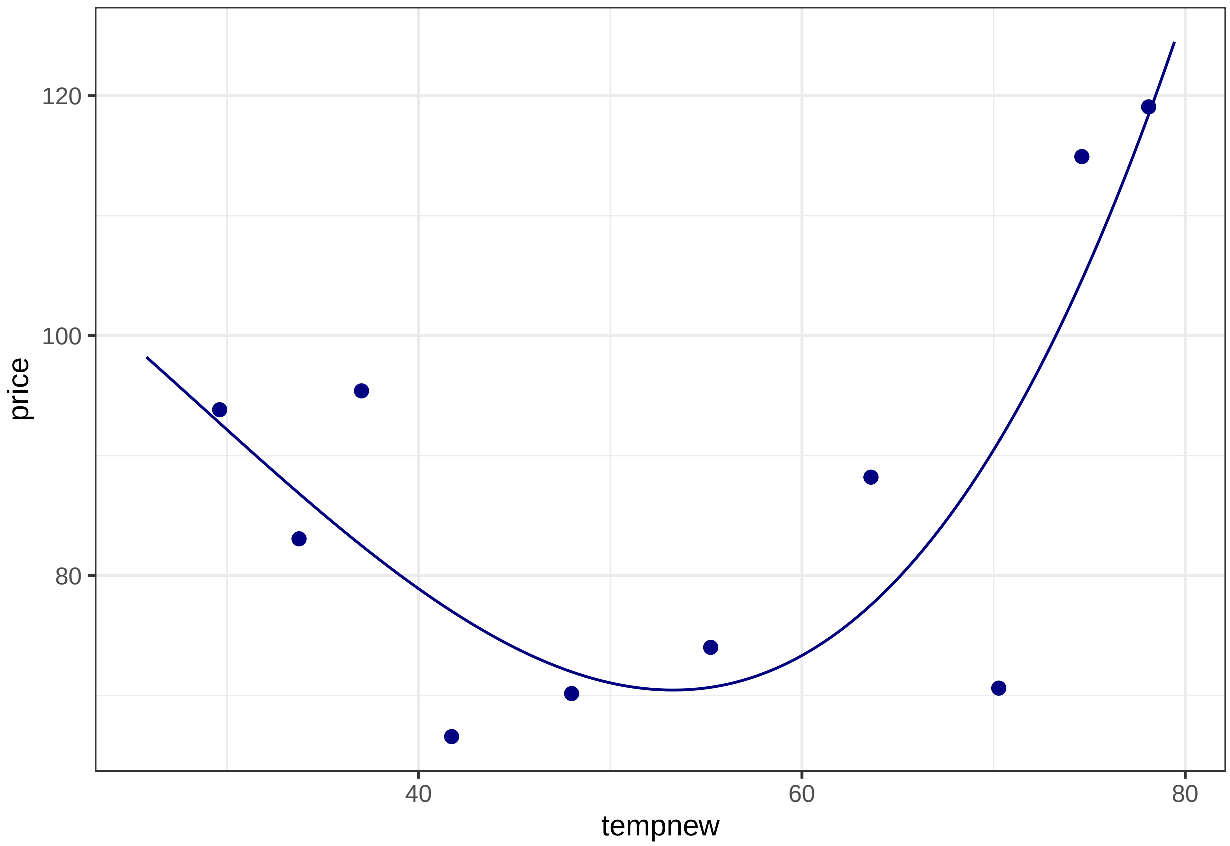
Part2

```
binsreg(y=price,x=tempnew, polyreg = 2)
```



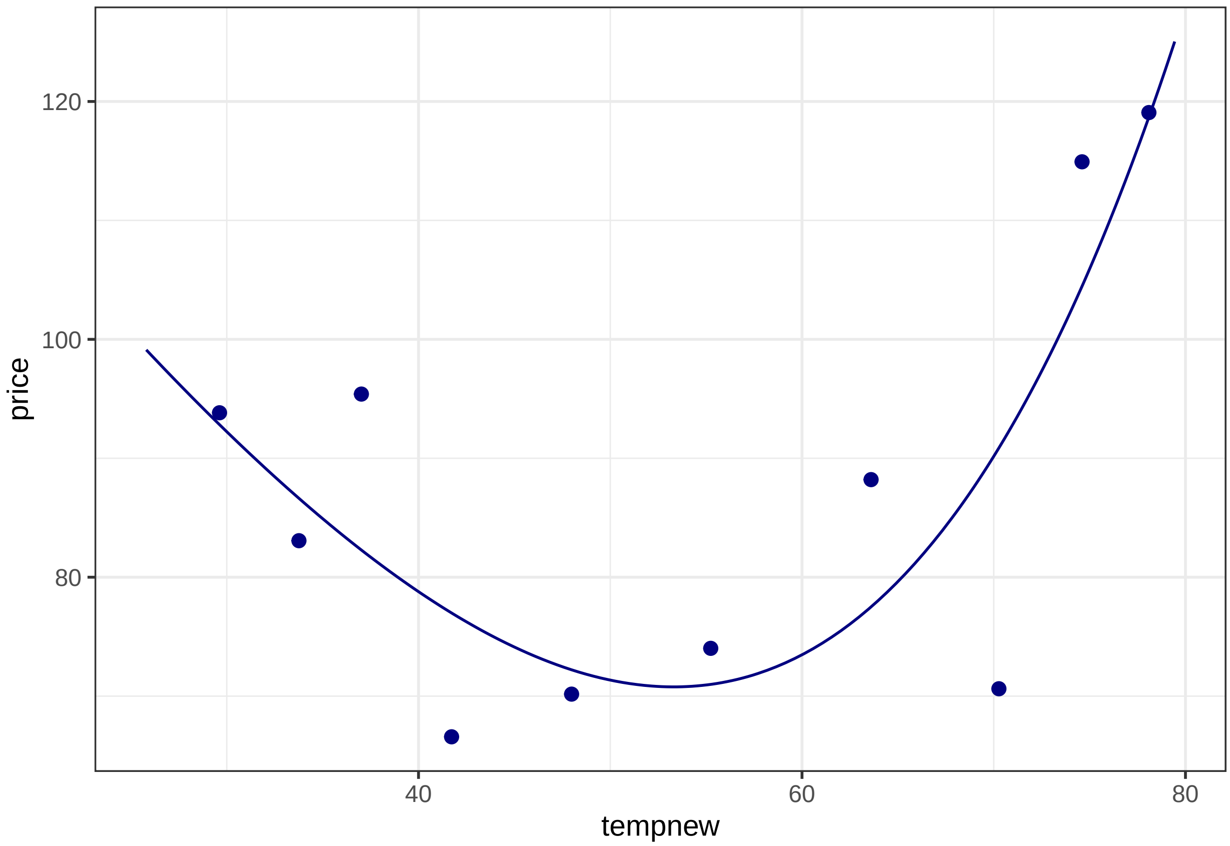

```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```
binsreg(y=price,x=tempnew, polyreg = 3)
```



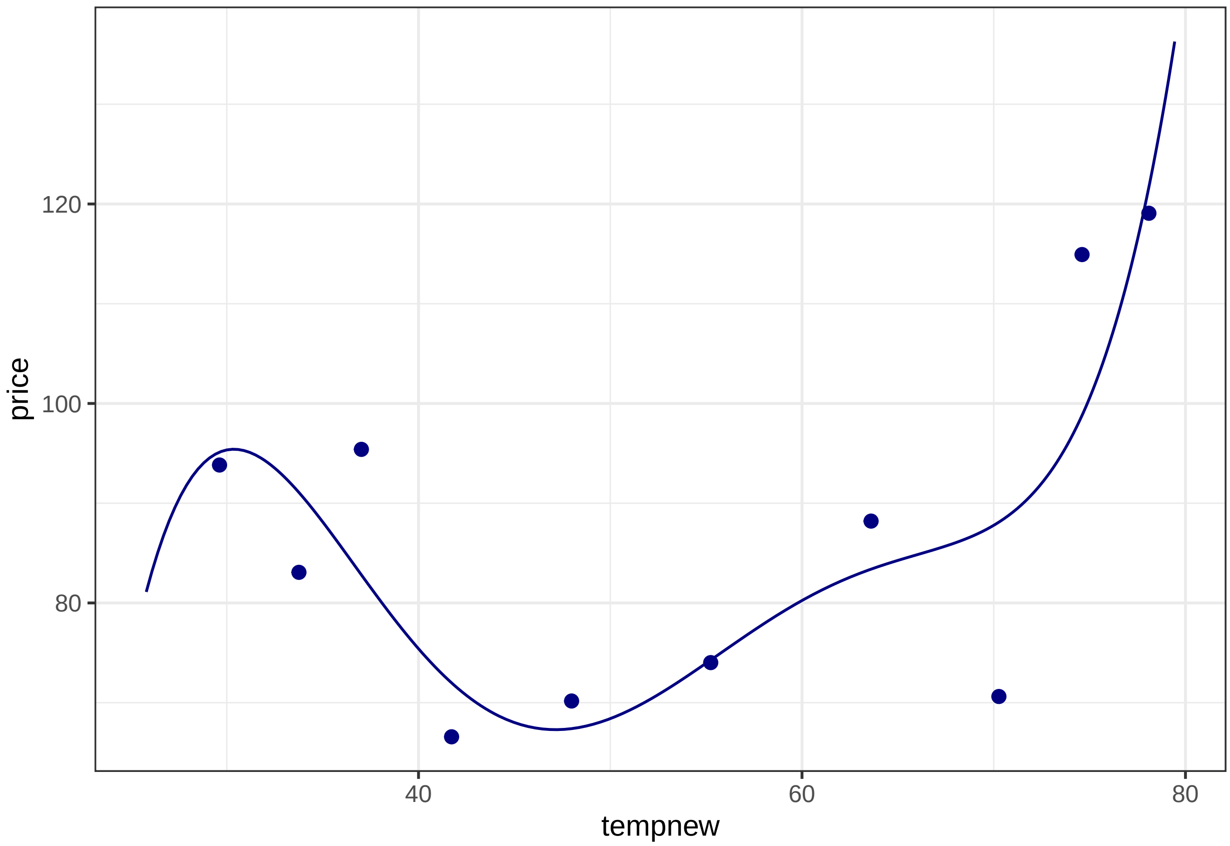

```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```r
binsreg(y=price,x=tempnew, polyreg = 4)
```



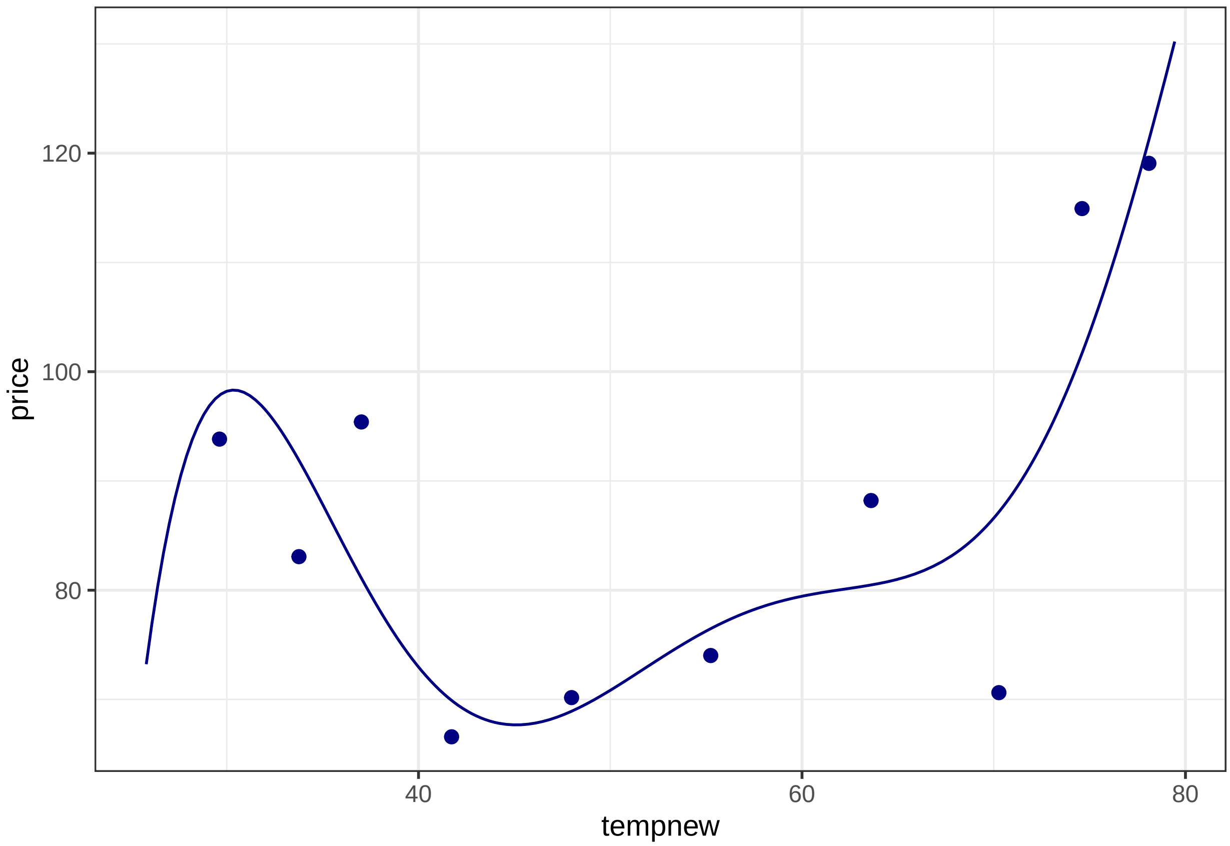

```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```
binsreg(y=price,x=tempnew, polyreg = 5)
```



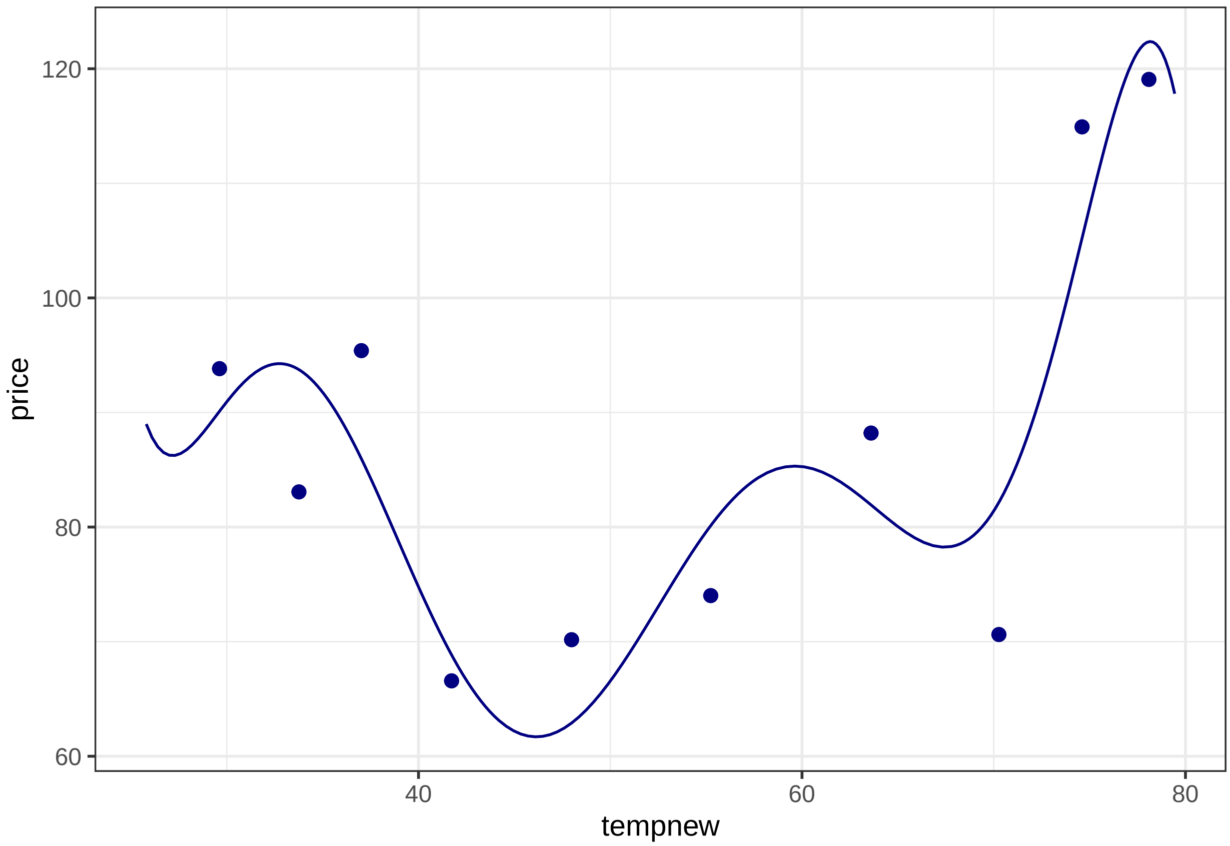

```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```r
binsreg(y=price,x=tempnew, polyreg = 6)
```



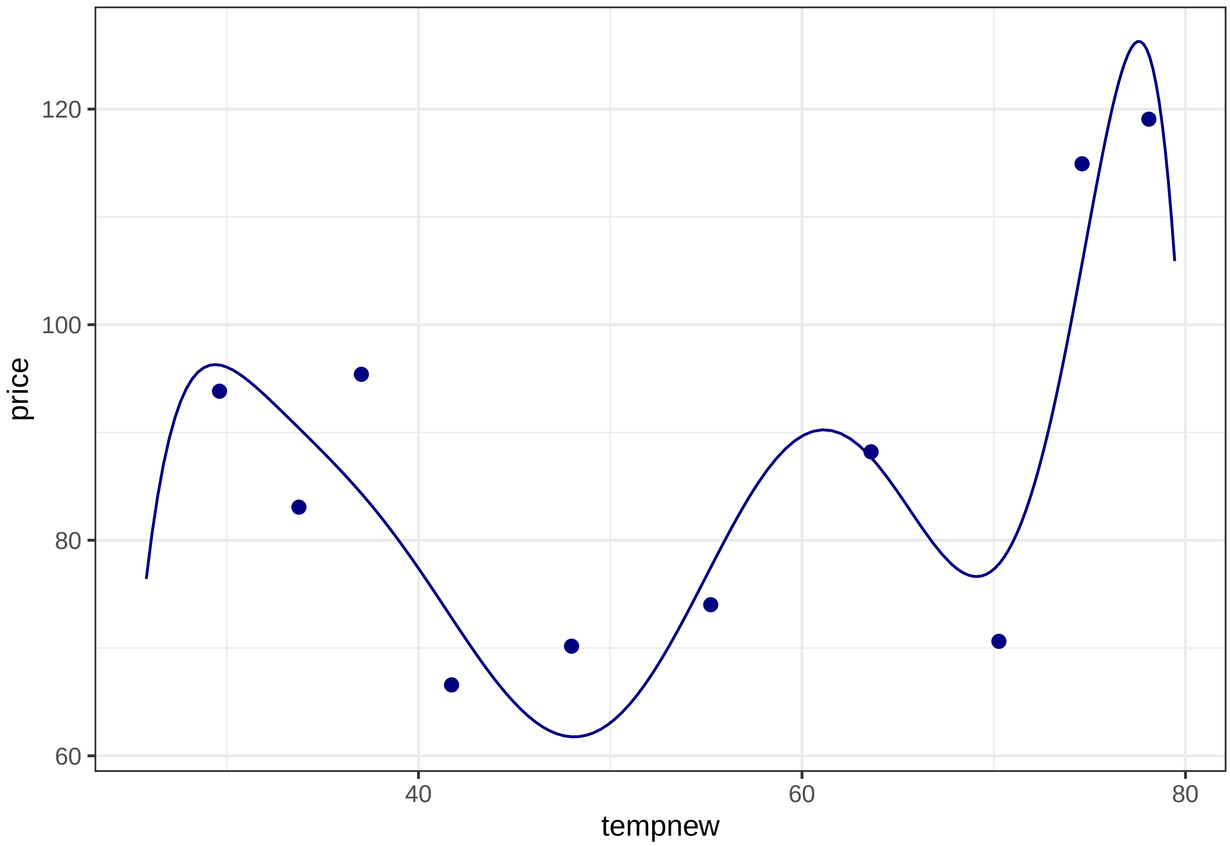

```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```r
binsreg(y=price,x=tempnew, polyreg = 7)
```



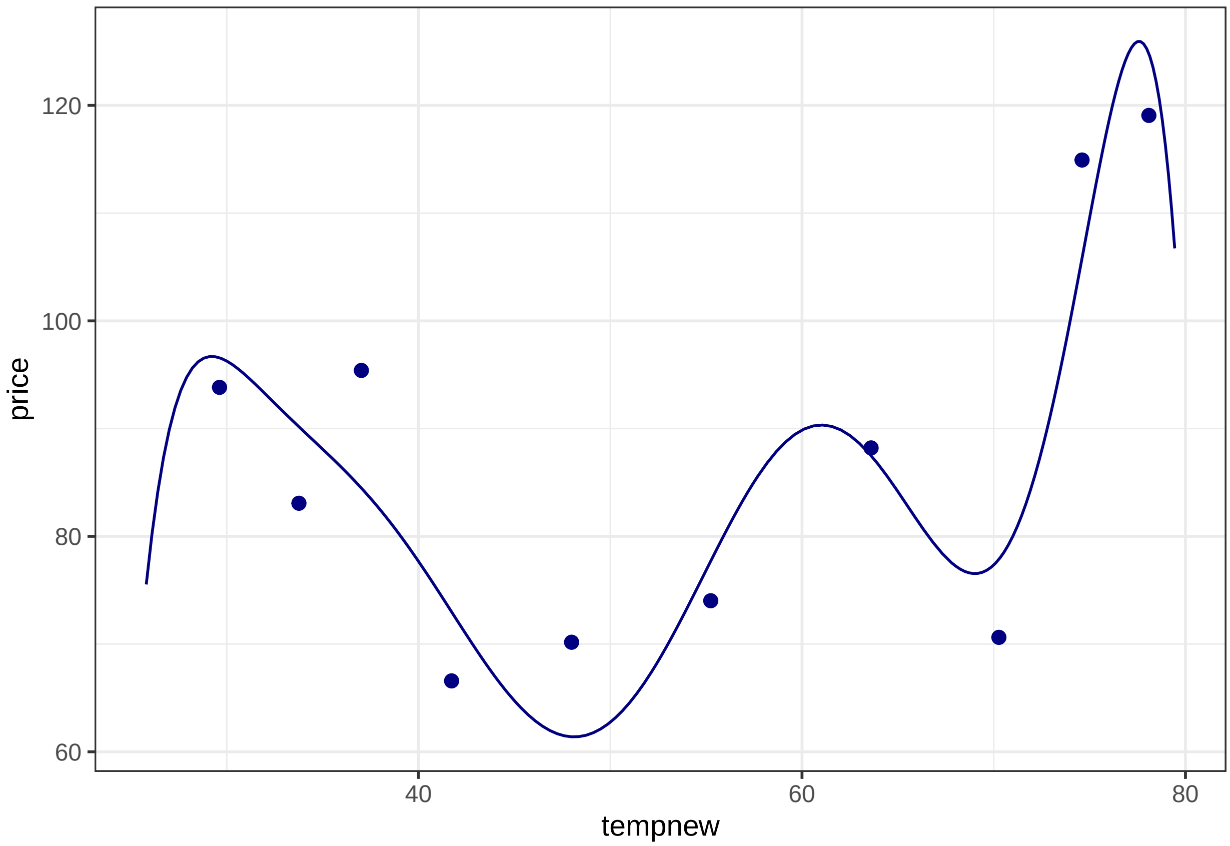

```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```
binsreg(y=price,x=tempnew, polyreg = 8)
```




```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

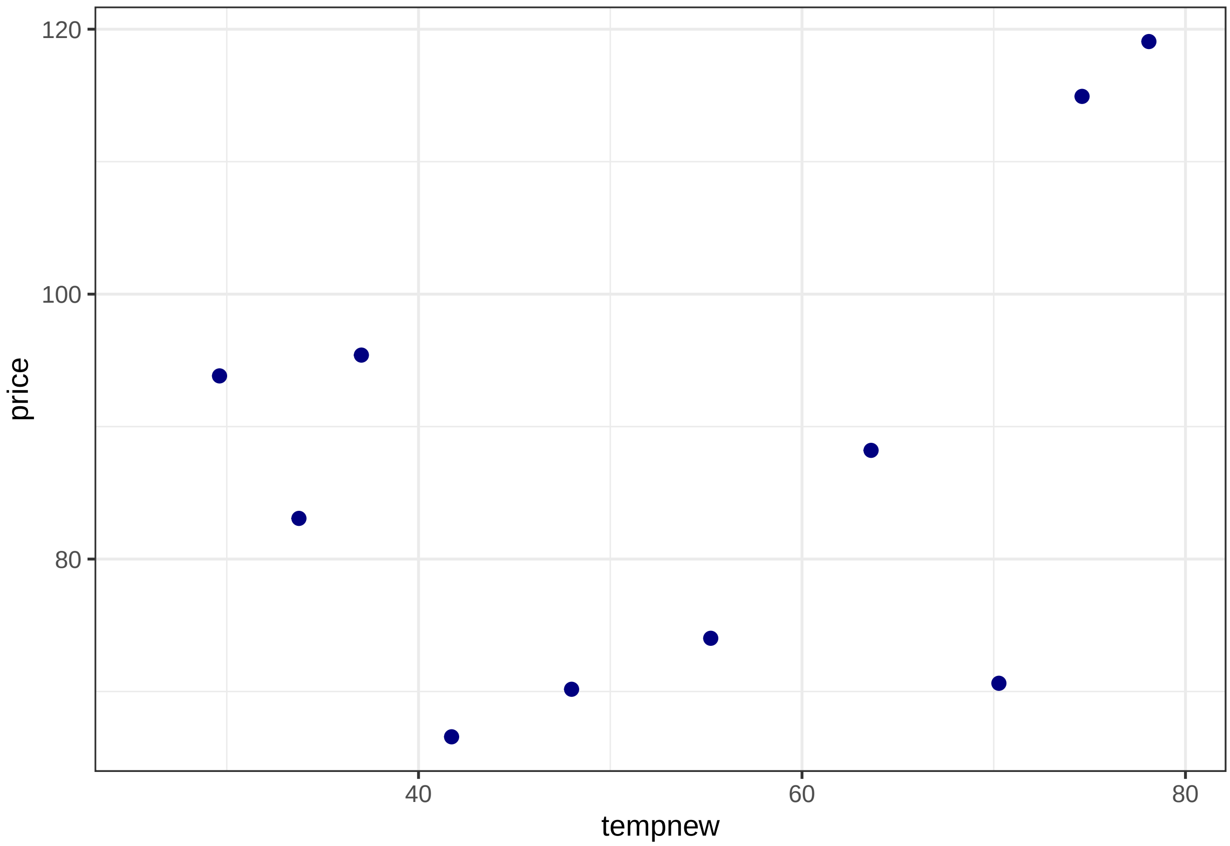
```
binsreg(y=price,x=tempnew, polyreg = 9)
```




```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

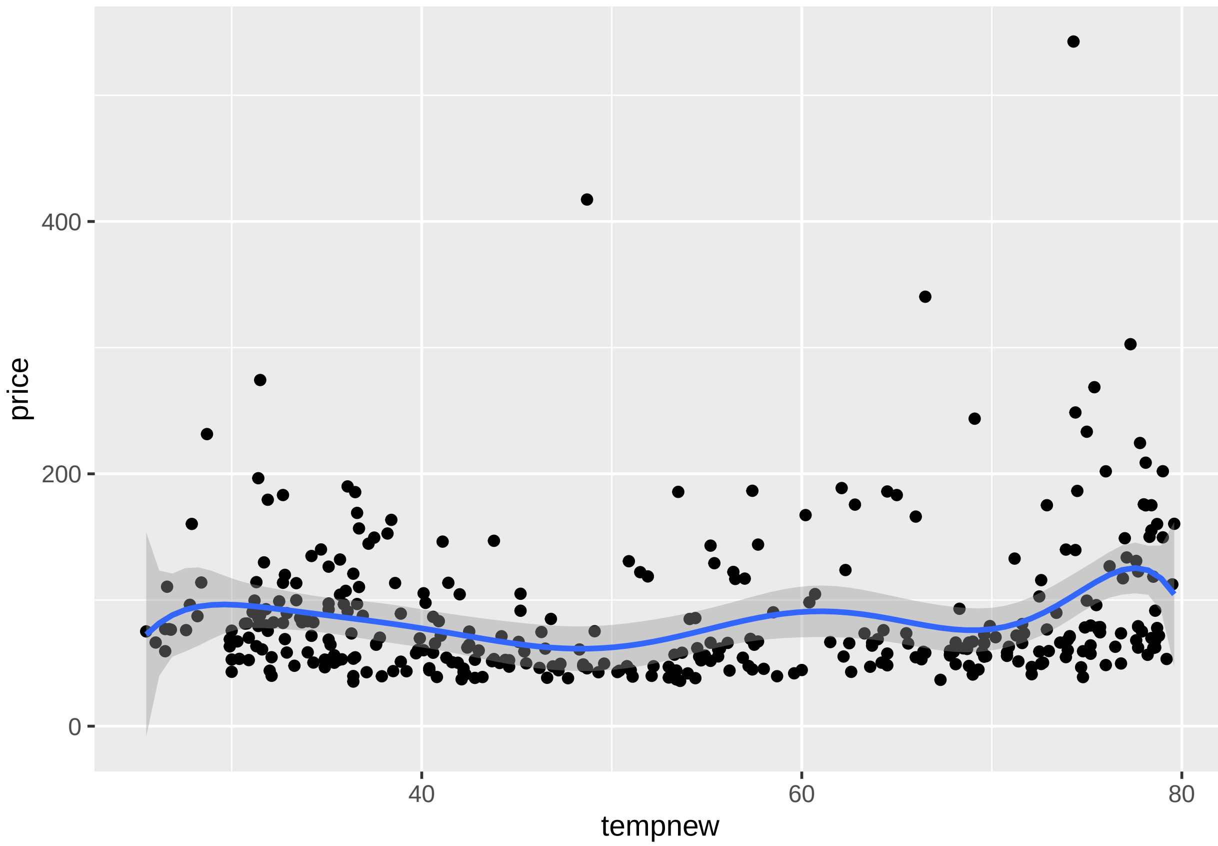
```
binsreg(y=price,x=tempnew, polyreg = 10)
```

```
## Warning: Removed 209 row(s) containing missing values (geom_path).
```




```
## Call: binsreg
##
## Binscatter Plot
## Bin selection method (binsmethod)  =  IMSE direct plug-in
## Placement (binspos)                =  Quantile-spaced
## Derivative (deriv)                 =  0
##
## Group (by)                         =  Full Sample
## Sample size (n)                    =  362
## # of distinct values (Ndist)       =  259
## # of clusters (Nclust)             =  NA
## dots, degree (p)                   =  0
## dots, smooth (s)                   =  0
## # of bins (nbins)                  =  10
```

```
ggplot(tp, aes(x=tempnew, y=price)) +
          geom_point() +
          stat_smooth(method='lm', formula = y ~ poly(x,10), size = 1) +
         xlab('tempnew') +
          ylab('price')
```



Part3

```
#randomly shuffle data
tp.shuffled <- tp[sample(nrow(tp)),]

#define number of folds to use for k-fold cross-validation
K <- 10

#define degree of polynomials to fit
degree <- 10

#create k equal-sized folds
folds <- cut(seq(1,nrow(tp.shuffled)),breaks=K,labels=FALSE)

#create object to hold MSE's of models
mse = matrix(data=NA,nrow=K,ncol=degree)

#Perform K-fold cross validation
for(i in 1:K){

    #define training and testing data
    testIndexes <- which(folds==i,arr.ind=TRUE)
    testData <- tp.shuffled[testIndexes, ]
    trainData <- tp.shuffled[-testIndexes, ]

    #use k-fold cv to evaluate models
    for (j in 1:degree){
```

```r
        fit.train = lm(price ~ poly(tempnew,j), data=trainData)
        fit.test = predict(fit.train, newdata=testData)
        mse[i,j] = mean((fit.test-testData$price)^2)
    }
}

#find MSE for each degree
colMeans(mse)
```

```
##  [1] 3224.666 3110.161 3115.080 3135.072 3121.200 3131.431 3126.363 3148.030
##  [9] 3170.313 3205.820
```

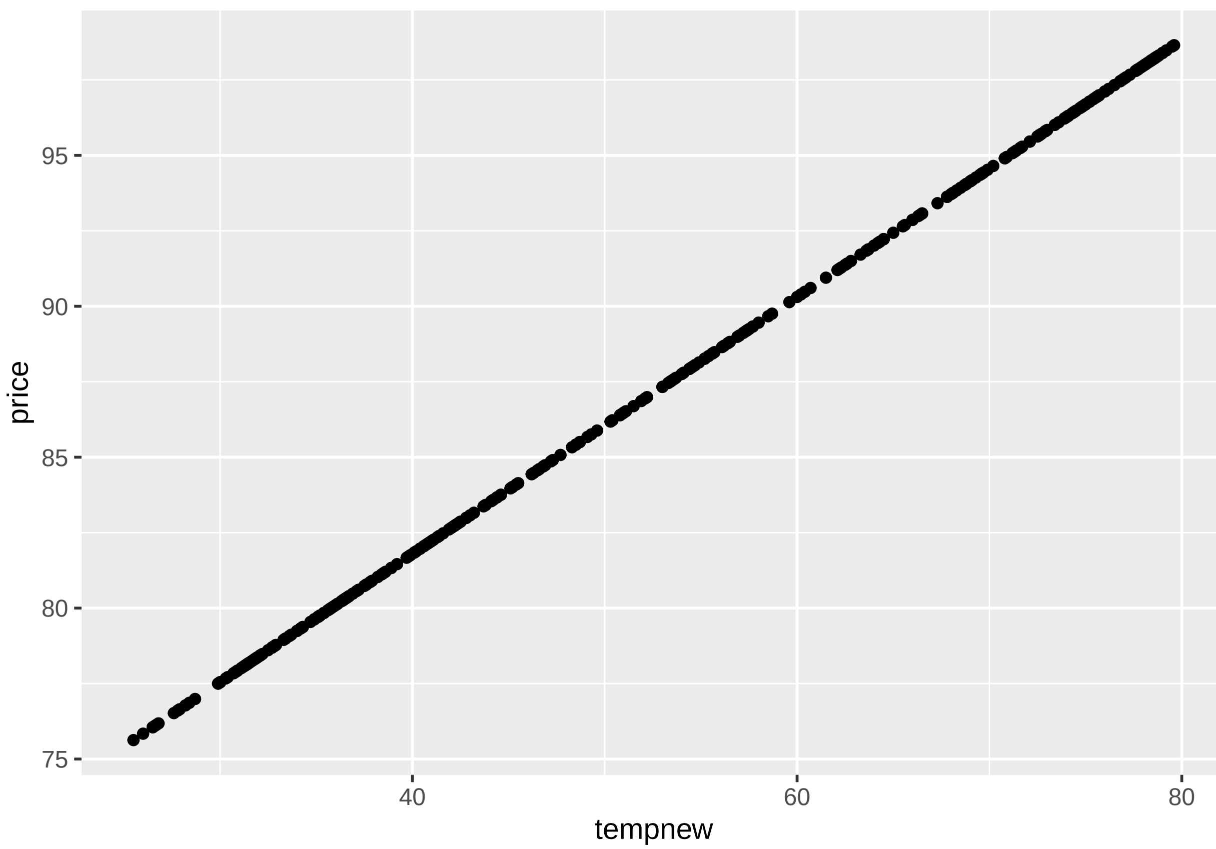
```r
min( colMeans(mse))
```

```
## [1] 3110.161
```

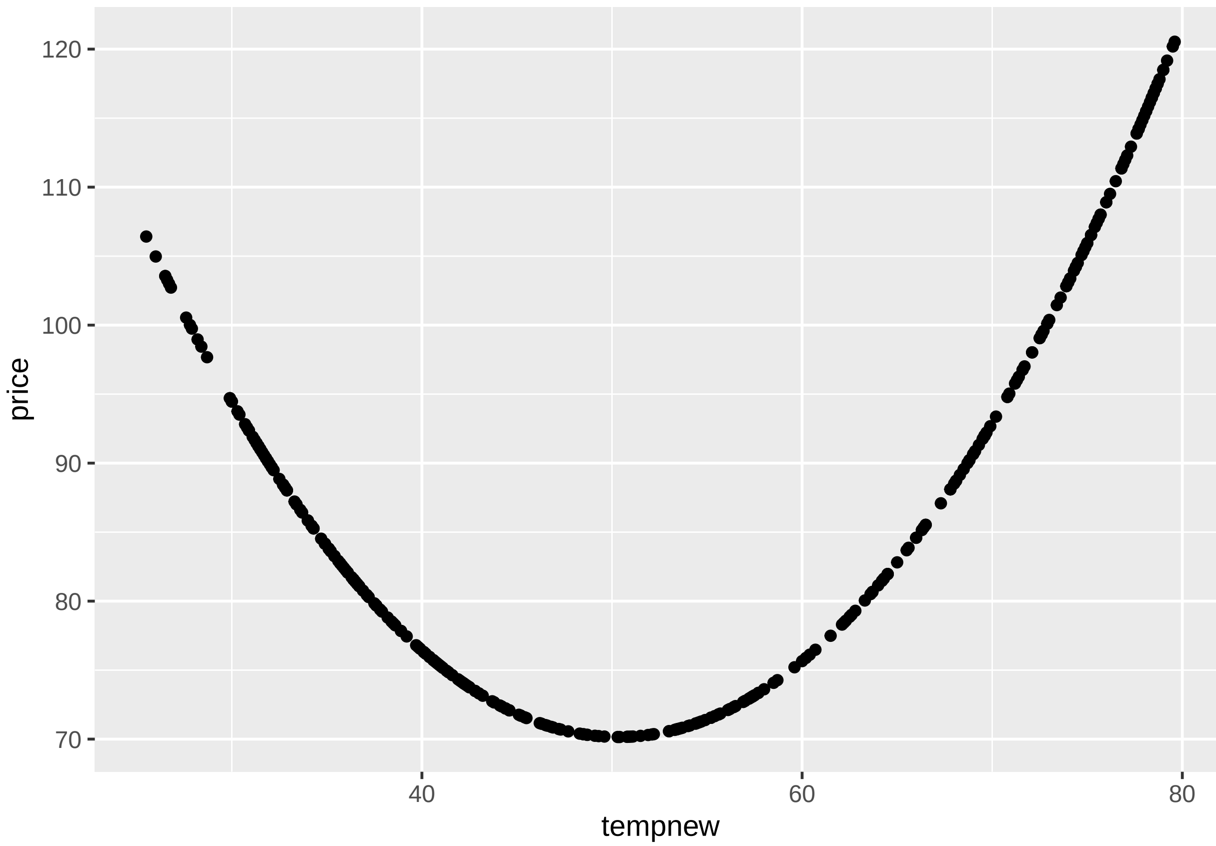
p=2 fits best now.

Part4

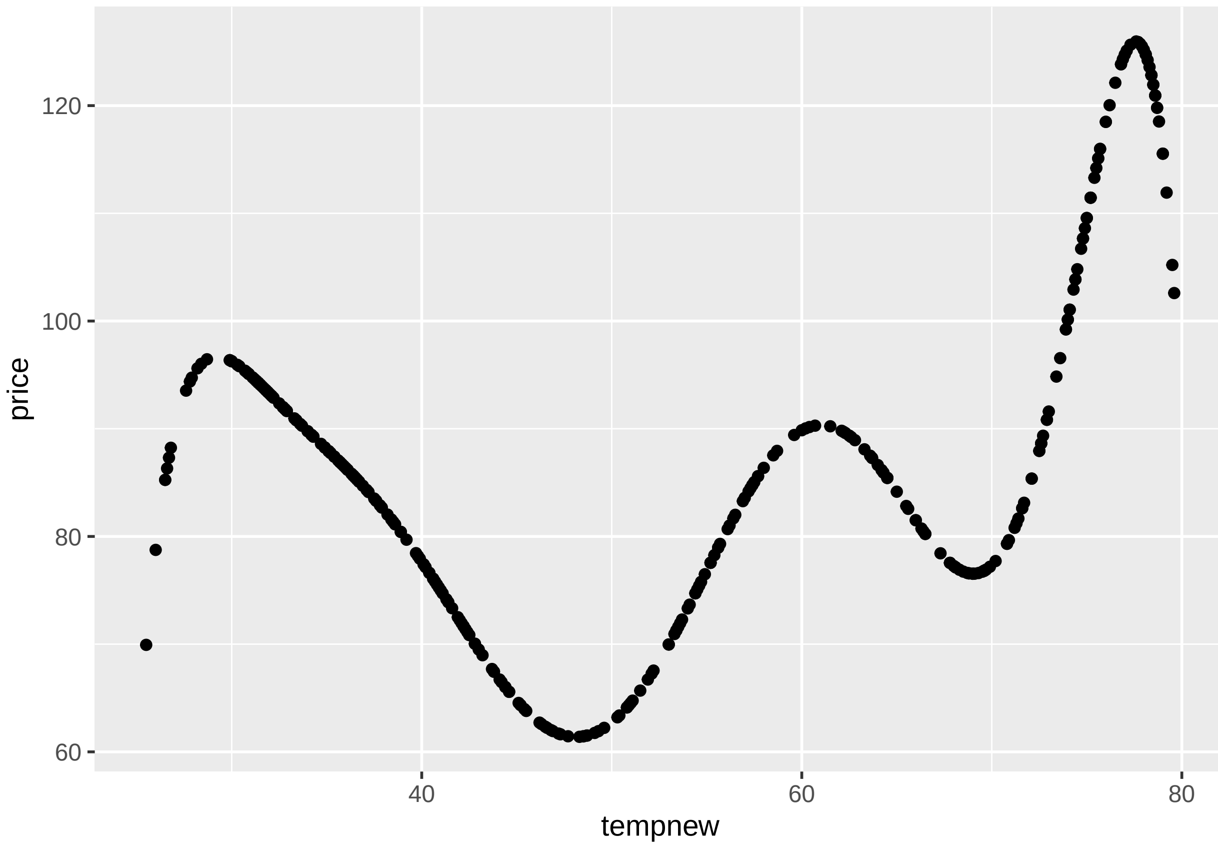
```r
first = lm(price ~ poly(tempnew,1, raw=T), data=tp)
price_1  <- predict(first)

ggplot(tp, aes(x=tempnew, y=price_1)) +
          geom_point() +
          xlab('tempnew') +
          ylab('price')
```



```r
second =lm(formula = price ~ poly(tempnew, 2, raw = T), data = tp)
price_2 <- predict(second)


ggplot(tp, aes(x=tempnew, y=price_2)) +
          geom_point() +
          xlab('tempnew') +
          ylab('price')
```




```r
tenth <- lm(formula = price ~ poly(tempnew, 10, raw = T), data = tp)
price_10 <- predict(tenth)

ggplot(tp, aes(x=tempnew, y=price_10)) +
          geom_point() +
          xlab('tempnew') +
          ylab('price')
```



Part5

```
library(splines)
tp.shuffled <- tp[sample(nrow(tp)),]

#define number of folds to use for k-fold cross-validation
K <- 10

#define degree of polynomials to fit
degree <- 10

#create k equal-sized folds
folds <- cut(seq(1,nrow(tp.shuffled)),breaks=K,labels=FALSE)

#create object to hold MSE's of models
mse = matrix(data=NA,nrow=K,ncol=degree)

#Perform K-fold cross validation
for(i in 1:K){
     #define training and testing data
    testIndexes <- which(folds==i,arr.ind=TRUE)
    testData <- tp.shuffled[testIndexes, ]
    trainData <- tp.shuffled[-testIndexes, ]

    for (j in 1:degree){
        fit.train = lm(price ~ bs(tempnew, df=j), data = trainData)
        fit.test = predict(fit.train, newdata=testData)
```

```
        mse[i,j] = mean((fit.test-testData$price)^2)
    }
}


colMeans(mse)
```

```
##  [1] 3067.082 3067.082 3067.082 3075.654 3056.068 3080.199 3057.789 3051.746
##  [9] 3086.671 3086.054
```

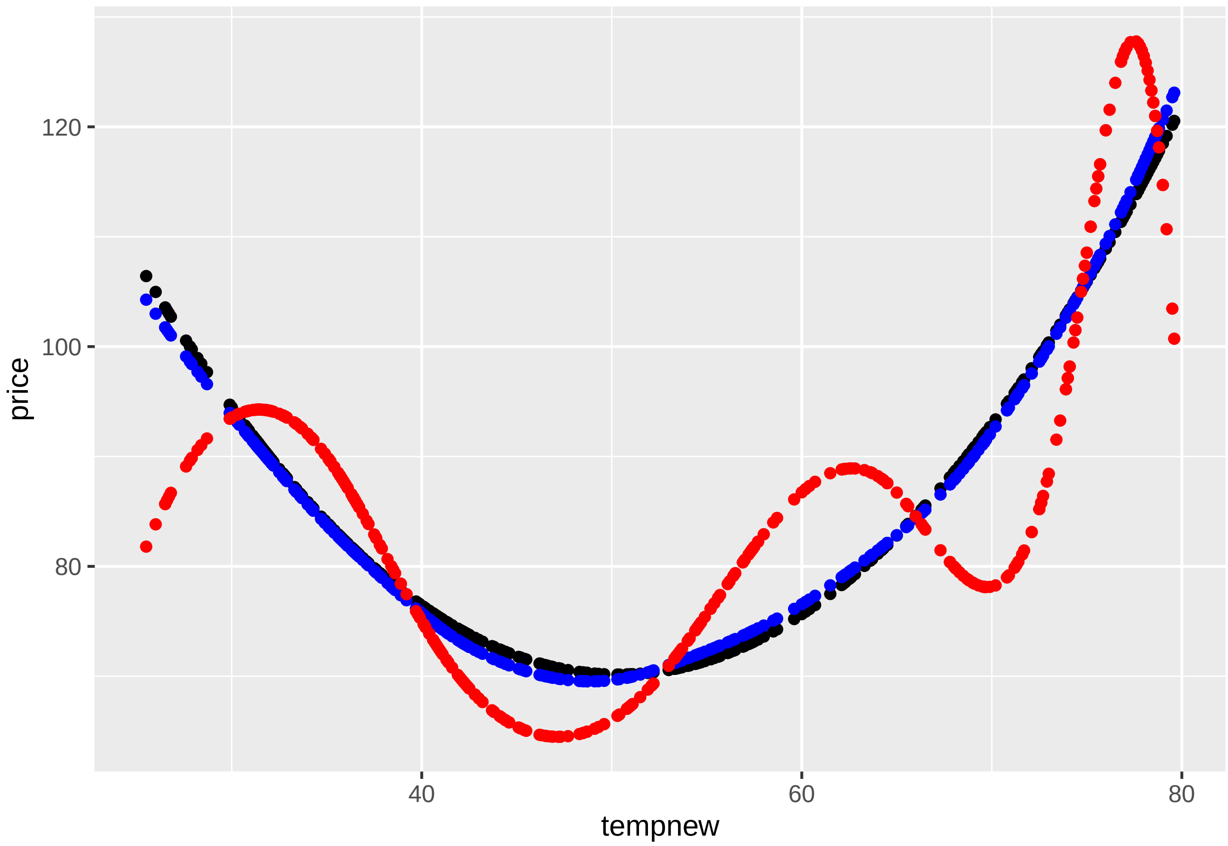
```
min(colMeans(mse))
```

```
## [1] 3051.746
```

Optimal number of knots is 8.

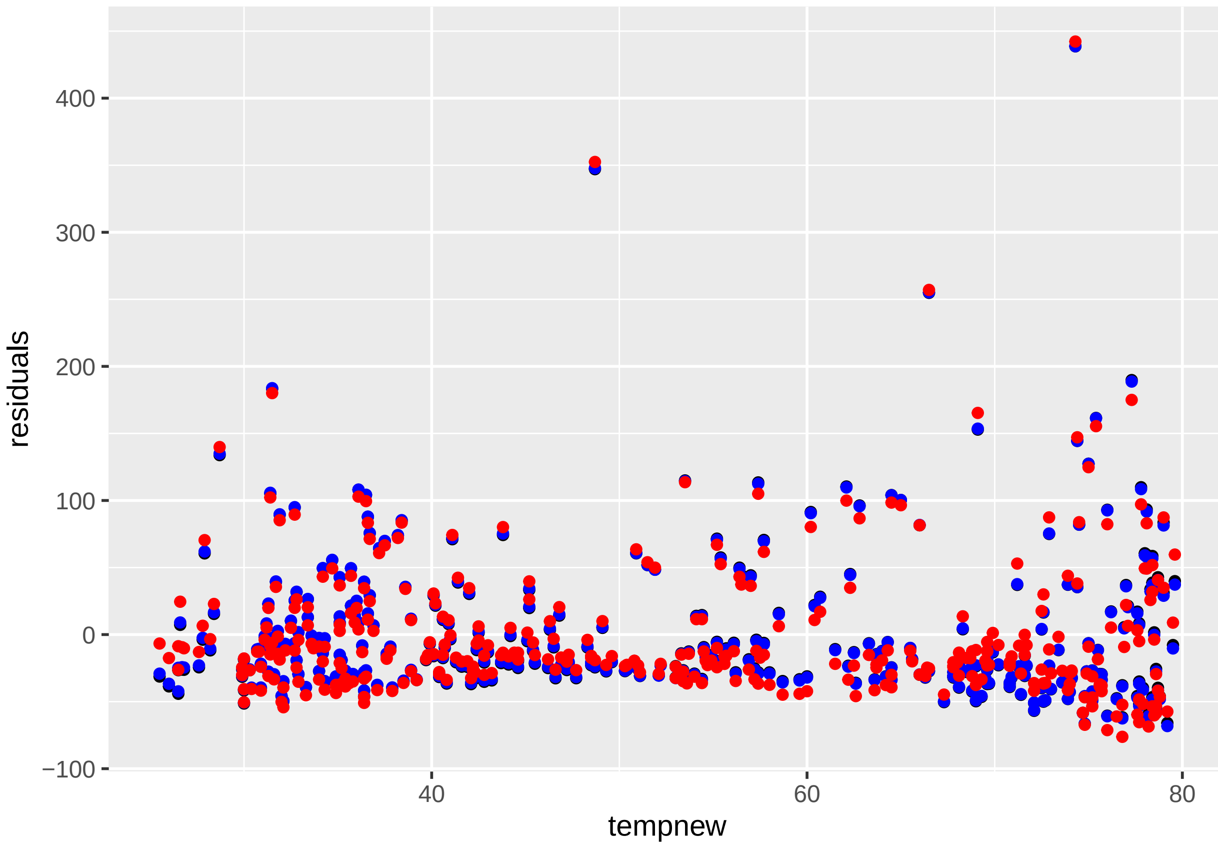
Part6

```
price_losess <- loess(price ~ tempnew, data=tp)
pre_price_losess <- predict(price_losess)

price_spline <- lm(price ~ bs(tempnew, df=8), data = tp)
pre_price_spline <- predict(price_spline)

ggplot(tp) +
  geom_point(aes(x=tempnew, y=price_2)) +
  geom_point(aes(x=tempnew, y=pre_price_losess), colour = 'blue') +
  geom_point(aes(x=tempnew, y=pre_price_spline), colour = 'red')+
  xlab('tempnew') +
  ylab('price')
```



Part7

```r
ggplot(tp) +
  geom_point(aes(x=tempnew, y=price-price_2)) +
  geom_point(aes(x=tempnew, y=price-pre_price_losess), colour = 'blue') +
  geom_point(aes(x=tempnew, y=price-pre_price_spline), colour = 'red') +
  xlab('tempnew') +
  ylab('residuals')
```



Medium part is better for all.