

Decision Support System for Managers
Prof. Sujoy Bhattacharya
Vinod Gupta School of Management
Indian Institute of Technology, Kharagpur

Lecture – 40
Pricing: Model Selection Using Cross – Validation

Hello and welcome to the “Decision Support System for Managers”! I will just recap what we did in the last class. We had talked about cross validation at a theory level, we had talked about MSE as an error definition which was used to select amongst models; then, we explained the basic ideas of the R-software: how to implement the basic cross validation at a toy data level with the R-software. Today what we are doing is, we are using the glm-based cross validation for a simple model of pricing.

(Refer Slide Time: 01:05)

The slide features a background with various icons including gears, a tree, and a brain. The text on the slide is as follows:

Concepts Covered

$Price = f(B, E)$

- Model selection using Cross-Validation using a Simple Pricing Model
- Implementation in R

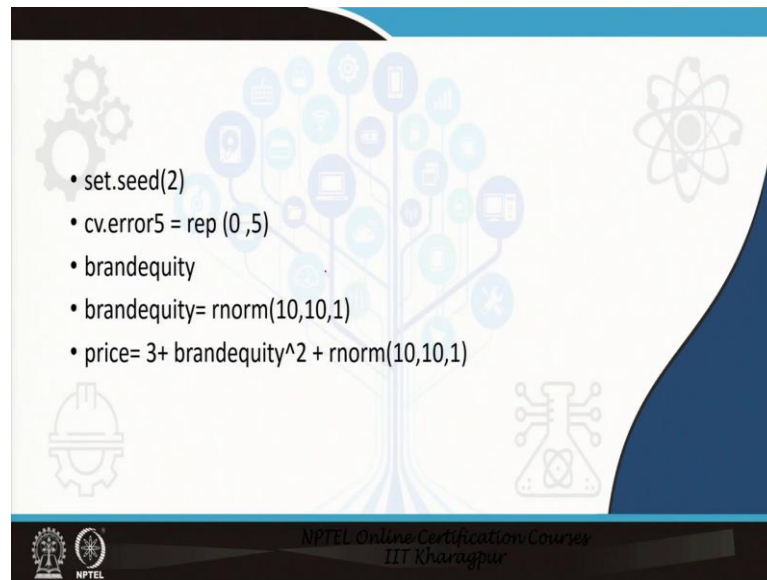
At the bottom of the slide, there are logos for NPTEL and IIT Kharagpur, along with the text "NPTEL Online Certification Courses IIT Kharagpur".

So, we start with today’s lecture. So, today’s concept which will be covered would be Model Selection Using Cross Validation using a simple pricing model. So, we are taking a very simple pricing model, we are saying that our price is a function of brand equity this is what we have been consistently talking about.

So, more brand equity more price less brand equity less price, but whether the relationship is a linear relationship or the relationship is a quadratic or a cubic relationship or a higher order relationship we will test it out. We will implement this in R. So, I have the codes I will explain the codes then we do an implementation of this,

then we move on to our data set called as the Boston housing data set. That Boston housing data set we are going to talk about and we are going to deal with the variable definitions of the Boston housing data set.

(Refer Slide Time: 02:33)



The slide displays the following R code:

- `set.seed(2)`
- `cv.error5 = rep(0,5)`
- `brandequity`
- `brandequity = rnorm(10,10,1)`
- `price = 3 + brandequity^2 + rnorm(10,10,1)`

The slide also features the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur" at the bottom.

So, I will explain the code one by one. So, the first line of the code is set dot seed in bracket. We use this set dot seed command because whenever there is a randomization in our model, we want to believe, we want to ensure that we have replicable results. So, if you work with this code, I work with this code or any third party works with this code our results should tally. So, as to ensure replicability, we have set dot seed and I give an argument to you should also give an argument to should you like to get the same results.

So, this is as far as the set dot seed is concerned. Now, once you have set dot seed this command is done, we define a variable called cv dot error5. So, this is a user defined definition you can call it anything you want as long as you understand it, I am calling it cv error5 because I am creating a variable which stores 5 of the cross validation errors for the 5 models I am going to test and I am going to initialize this with 0s.

So, I am creating a vector of length 5 populated by 0s. So, 0 0 0 0 0 this is the initialization of the variable cv dot error5. So, I am going to define a variable called brand equity and the brand equity I am creating I am simulating as a R norm; this I have already done in the previous class. So, I am simulating 10 points from a normal

distribution with the parameters as 10 and 1 the two parameters of the normal distribution.

So, thereby I am creating a brand equity vector of length 10. Now, I am defining another variable called price. So, price is I am creating this equation price is 3 plus brand equity to the power 2 plus another randomization R norm, again a vector of 10 is added as an additive noise and the vector of 10 comes from another normal distribution with 10 and 1.

So, mu sigma square is 10 1 right. So, I have created a data set which has a quadratic relationship between price and brand equity fine. So, the thing to know is, whenever I do a R norm 10 here and an R norm 10 here; that means, a vector of length 10 here and a vector of length 10 again the second time these two are going to give two different results because this is simulating 10 points from a normal distribution of a given mu sigma square.

So, I have two definitions ready for the two variables and I have a relationship between price and a brand equity known to me. The next step would be to see which model does better. So, I have created a series of models.

(Refer Slide Time: 06:30)

```
• data1=data.frame(price, brandequity)
• for (i in 1:5){
• model.glm =glm(price~poly(brandequity,2),data=data1)
• cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]
• Boston Housing Price Data Set:
• https://www.rdocumentation.org/packages/mlbench/version/1/topics/BostonHousing
```

So, if you look at the previous lecture which I had; I had talked about nested models. So, nested models was the one bigger model is containing the smaller model with less

parameters that is how we looked at the nested model, you can look at the previous lecture and you will see how I created the nested model.

So, now what do I do? I create a data set called data 1 and I am converting this to a data frame using the definition which I have created price and the data which we populated with this equation price equal to 3 plus brand equity square plus R norm with a vector size of 10 and a second variable brand equity. So, in a sense I have created data 1 containing two columns the first column is price and the second column is brand equity.

Now, I have the data one available with me ok. So, now, I have a data set available with me of the type data dot frame. In our previous class when I introduced R I talked about data dot frame, this is the type of data in R which allows me to take heterogeneous column types. So, one column could be numeric, the second column could be date, the third column could be a character and so on.

So, in this case I convert price and brand equity all these; all of these two variable definitions to a data dot frame right. Now, I am going to talk about what do I do next. I create a for loop in R and the for loop in R goes like for i in 1 to 5; that means, the value of i will move from 1 to 5 and I am creating an object called model dot glm and this fits a glm model because I have not given a family argument in this, this will fit a linear model a linear regression.

First when I give this as 1 when I give this is 2; it will fit a quadratic model and so, on fine. So, what do I do? I will try and move on. So, I do a model dot glm, then I do a price and then I do a poly. So, when I do a price and when I do a poly this part brand equity to this means I am fitting a quadratic equation; fine.

Now, if were to replace this with i, this would take the values 1, 2, 3, 4, 5. So, the first time I fed this it will be a linear model, second time it will be a quadratic model, third time it will be a cubic model, then degree 4 and then degree 5 right and the data is coming from the data dot frame which we have defined here fine. So, what I have done? I am fitting an glm type object and I am varying the equations by using a poly argument and I am passing the argument i here.

If I were to do it 2 this is a quadratic equation. So, I am passing i for a general sense and then I am calling it data is coming from the data right. So, now, look at the now look at

this line which I am underlining right. Now, I am saying right. So, `cv dot glm` I am doing a cross validation on `glm` right. So, then I am doing a `data 1` that is the data is coming from the data frame I have created, then the `model dot glm` which is here the object which stores all the information about the model whether it is linear or quadratic or cubic that is brought here.

And then this `dollar delta 1` is the cv error the cross validation error this is what I am trying to extract. So, once I run this for `i equal to 1` let me see what happens when I do it for `i equal to 1` I do a `poly brand equity 1`; that means, price is regressed on brand equity and its a linear regression equation because `i goes 1`. The linear regression results are brought here in `model dot glm` here.

And a cross validation is achieved and the cross validation error is with me right ok. Now, this argument `K` controls what should be the `K` of the `K` fold cross validation remember we had done a `K` fold cross validation right; I can keep `K` equal to 5, I can keep `K` equal to 7, I can keep `K` equal to 10. So, I have just for the demonstration `K` equal to 2, but generally we keep `K` higher than 5; 5 6 7 or even we go for `1 o cv` which I explained in the previous lecture.

So, now `model dot glm` will run in the first iteration a linear model with a `K 4` cross validation and gives me a `delta`. If I were to do this for `i equal to 2` this will do this cross validation for the quadratic model, if I were to do this for 3 this will do it for the cubic model and so, on. So, I will continue to get the cross validation errors how so, ever number of times I have put the value of `i` as the end value that is in this case `i` ends at 5.

So, `cv error indexed i` would mean `i` goes from 1 to 5 or so, it will be `cv error5` for the first time second time third time fourth time and fifth time and every time I am going to get the cross validation errors right. I can later on see what are the cross validation errors. Now, the other thing. So, this is a basic explanation of the code I will be using to do cross validation for model selection and `glm` has a built in cross validation in it and hence, I am using even for a linear model `glm` rather than I am fine; ok.

So, I move on and I look at the implementation of this code.

(Refer Slide Time: 15:15)

```

> set.seed(2)
> cv.error5 = rep(0,5)
> cv.error5
[1] 0 0 0 0 0
> brandequity= rnorm(10,10,1)
> price= 3+ brandequity^2 + rnorm(10,10,1)
> data=data.frame(price, brandequity)
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,2),data=data1)
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]
+ }
> cv.error5
[1] 141.998266  3.338274  9.058143
>

```

So, let us look at the implementation of this code. And I am doing it for 2 here fine and let us see how it goes. So, first of all set dot seed 2 I have set a seed 2 this is ensures that the results are replicable, then I am initializing the cv error5.

So, I have created a initialized vector of length 5, then I have done a brand equity, then I have created the relationship between price and brand equity, then I have created a data frame, then I have run the loop right. So, this is with me ok. So, if this is ok. So, let us see what happens next; right.

(Refer Slide Time: 16:27)

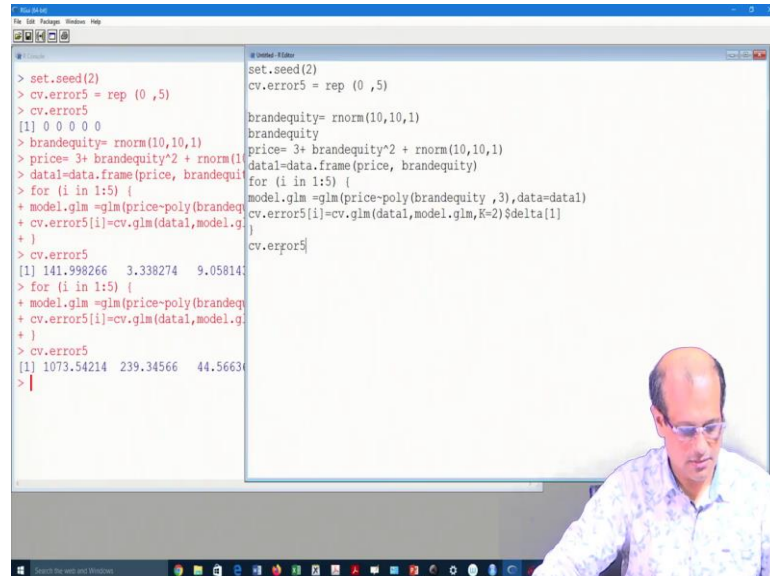
```

> set.seed(2)
> cv.error5 = rep(0,5)
> cv.error5
[1] 0 0 0 0 0
> brandequity= rnorm(10,10,1)
> price= 3+ brandequity^2 + rnorm(10,10,1)
> data=data.frame(price, brandequity)
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,2),data=data1)
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]
+ }
> cv.error5
[1] 141.998266  3.338274  9.058143  4.739509  3.770051
>

```

So, I want to see what is the cross validation error I could run this and see what is the cross validation error, I could see the cross validation error for i goes 1 2 3 4 5; fine.

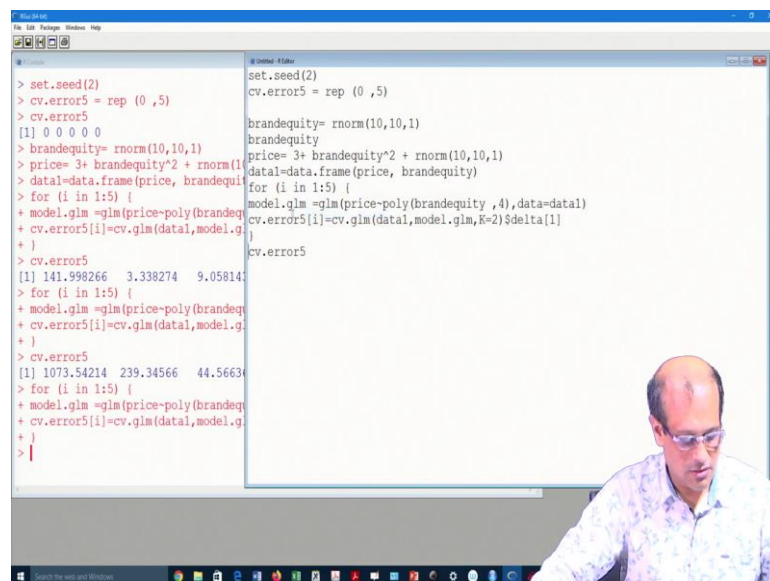
(Refer Slide Time: 16:51)



```
> set.seed(2)
> cv.error5 = rep(0,5)
> cv.error5
[1] 0 0 0 0 0
> brandequity= rnorm(10,10,1)
> price= 3+ brandequity^2 + rnorm(10,10,1)
> data=data.frame(price, brandequity)
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity,2),data=data)
+ cv.error5[i]=cv.glm(data,model.glm,K=2)$delta[1]
+ }
> cv.error5
[1] 141.998266  3.338274  9.058144  1.000000  1.000000
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity,3),data=data)
+ cv.error5[i]=cv.glm(data,model.glm,K=2)$delta[1]
+ }
> cv.error5
[1] 1073.54214 239.34566 44.56634 1.000000 1.000000
> |
```

Now, if this is the case what would happen if I were to make this as 3 and the cross validation error? Right; fine. So, I can again keep on changing the regressions order and I can keep on seeing the cross validation error, fine. So, should you set the seed to 2; you will get the same results; right.

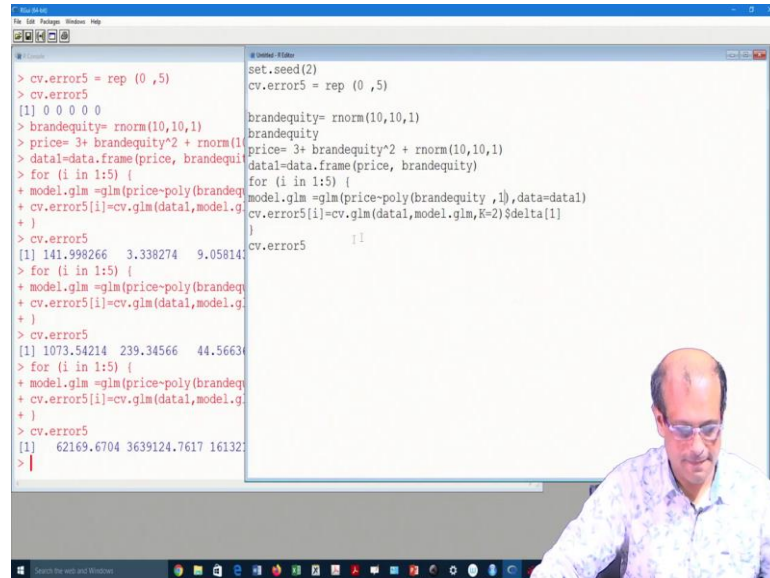
(Refer Slide Time: 17:35)



```
> set.seed(2)
> cv.error5 = rep(0,5)
> cv.error5
[1] 0 0 0 0 0
> brandequity= rnorm(10,10,1)
> price= 3+ brandequity^2 + rnorm(10,10,1)
> data=data.frame(price, brandequity)
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity,4),data=data)
+ cv.error5[i]=cv.glm(data,model.glm,K=4)$delta[1]
+ }
> cv.error5
[1] 141.998266  3.338274  9.058144  1.000000  1.000000
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity,3),data=data)
+ cv.error5[i]=cv.glm(data,model.glm,K=3)$delta[1]
+ }
> cv.error5
[1] 1073.54214 239.34566 44.56634 1.000000 1.000000
> |
```

So, what if I now? Say this is 4, I will run the same model now with another order and I will get the cross validation right now what if I say that this is say 1.

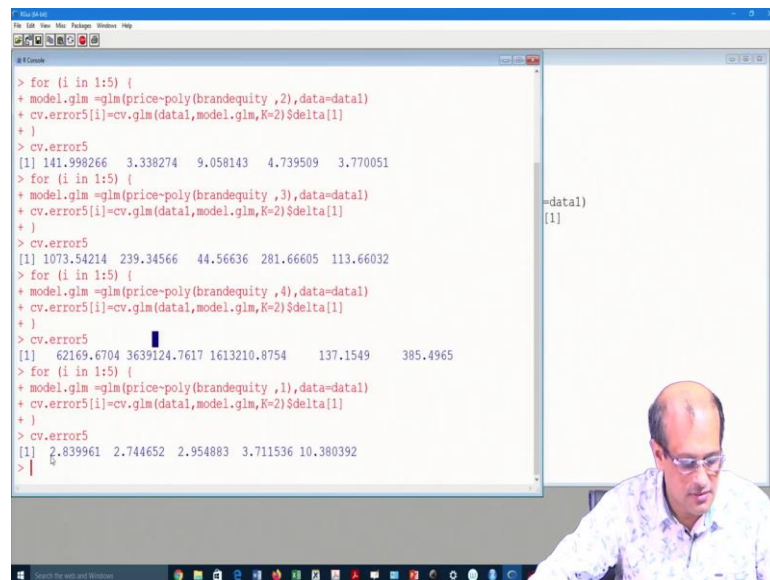
(Refer Slide Time: 17:55)



```
> cv.error5 = rep(0,5)
> cv.error5
[1] 0 0 0 0 0
> brandequity = rnorm(10,10,1)
> price = 3 + brandequity^2 + rnorm(10,10,1)
> data1 = data.frame(price, brandequity)
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,2), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 141.998266 3.338274 9.058143 4.739509 3.770051
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,3), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 1073.54214 239.34566 44.56636 281.66605 113.66032
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,4), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 62169.6704 3639124.7617 1613210.8754 137.1549 385.4965
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,1), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 2.839961 2.744652 2.954883 3.711536 10.380392
>
```

Now, this is a question which I would like you to think about.

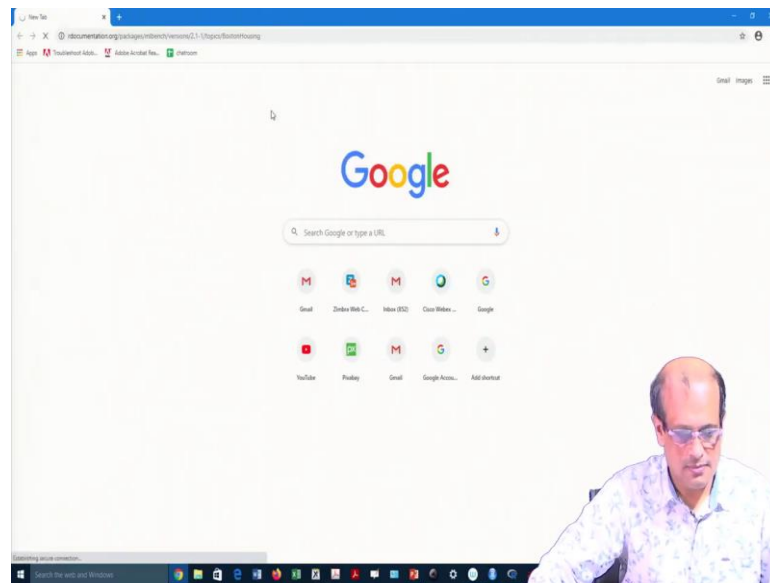
(Refer Slide Time: 18:06)



```
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,2), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 141.998266 3.338274 9.058143 4.739509 3.770051
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,3), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 1073.54214 239.34566 44.56636 281.66605 113.66032
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,4), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 62169.6704 3639124.7617 1613210.8754 137.1549 385.4965
> for(i in 1:5) {
+ model.glm = glm(price~poly(brandequity,1), data=data1)
+ cv.error5[i] = cv.glm(data1, model.glm, K=2)$delta[1]
+ }
> cv.error5
[1] 2.839961 2.744652 2.954883 3.711536 10.380392
>
```

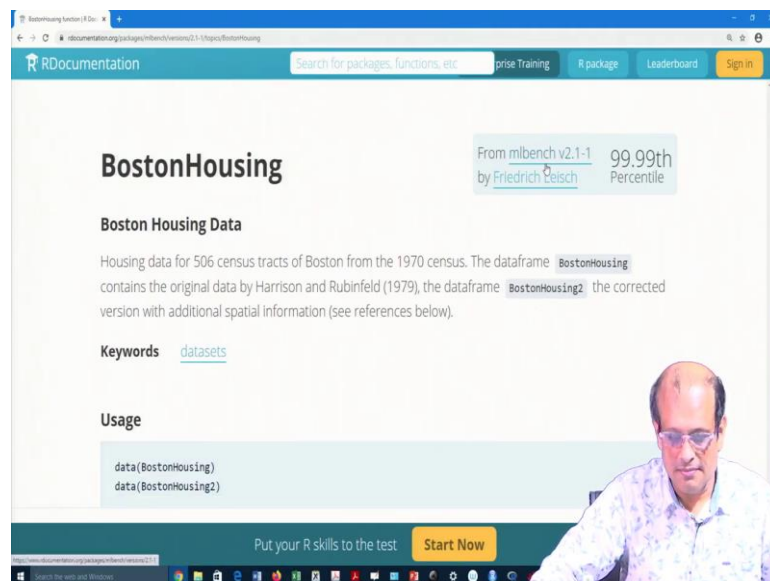
So, what do you think is happening now? So, in this case we are running a linear regression equation with a poly 1, fine right and let me now go to the data set which I talked off.

(Refer Slide Time: 18:42)



And we need to look at this data set; we will be using this data set for the next class.

(Refer Slide Time: 18:50)



So, I will talk about this data set and this data set is coming from mlbench version 2.1-1 and this data set is a very popular data set, this data set is giving 506 census tracts of Boston from the 1970 census.

(Refer Slide Time: 19:16)

Format

The original data are 506 observations on 14 variables, `medv` being the target variable:

`crim`
per capita crime rate by town

`zn`
proportion of residential land zoned for lots over 25,000 sq.ft

`indus`
proportion of non-retail business acres per town

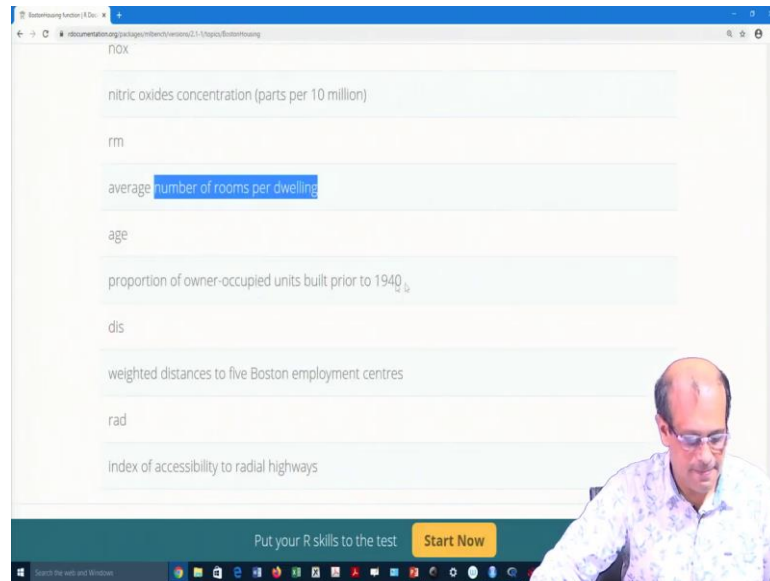
`chas`
Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)

Put your R skills to the test [Start Now](#)

And what is the objective? The objective is to predict the price of the houses in a particular locality in Boston. So, the prices are given by `medv` and this is called as the target variable. So, this is what we will try to predict. So, our agenda is to predict the housing prices.

So, the variables which are available to me to predict the housing prices are `crim` that is a per capita crime rate then the `zn` which is the proportion of residential land zoned for lot size over 25,000 square feet then another variable called `Indus`, then which is the non-retail business acres per town.

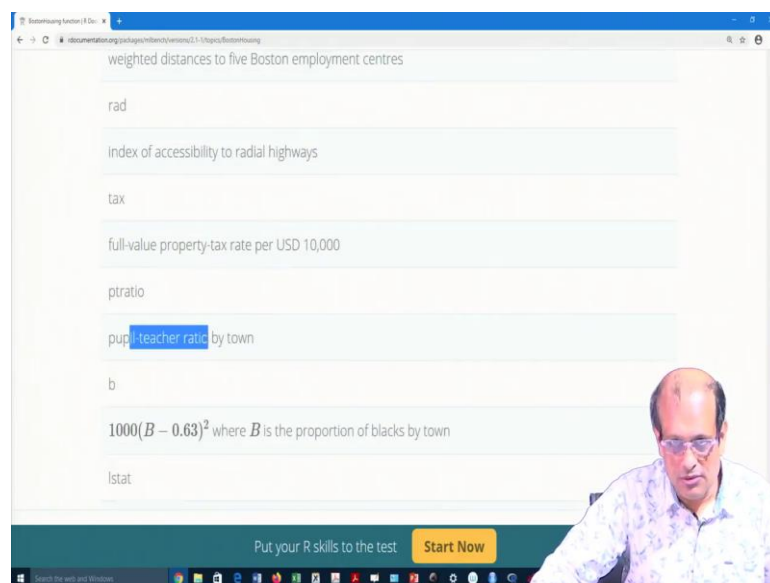
(Refer Slide Time: 19:59)



Then we have a dummy variable here that is a categorical variable in a sense because this talks off whether you are on one side of the Charles river on the other side of the Charles river right then we have nitric oxides concentration.

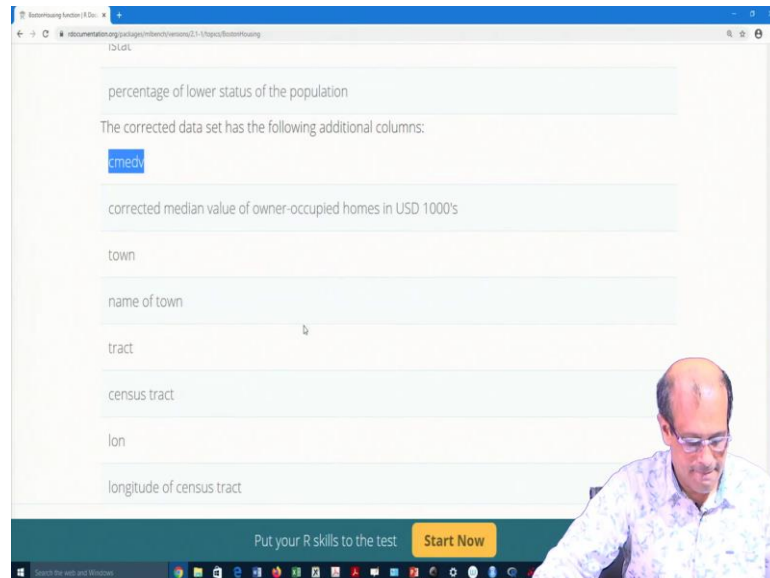
So, we believe that all these variables which are given would either have a positive correlation or a negative correlation with the housing prices then we have average number of rooms per dwelling, then we have age of the owner occupied units.

(Refer Slide Time: 20:44)



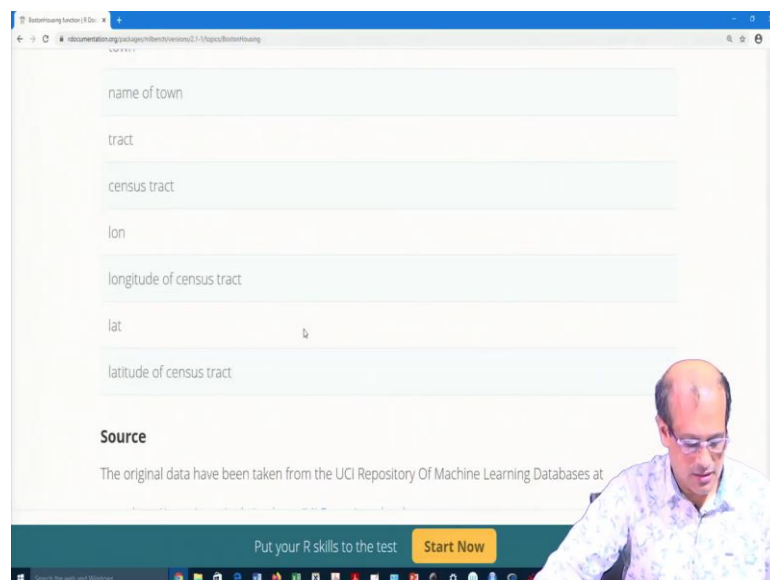
Then weighted distance to 5 Boston employment centers, then index of accessibility to the radial highways, then full property tax rate, then the pupil teacher ratio in the town.

(Refer Slide Time: 20:59)



And percentage of lower status of population and the median value of owner occupied homes this is the medv cmedv variable we which we have; ok.

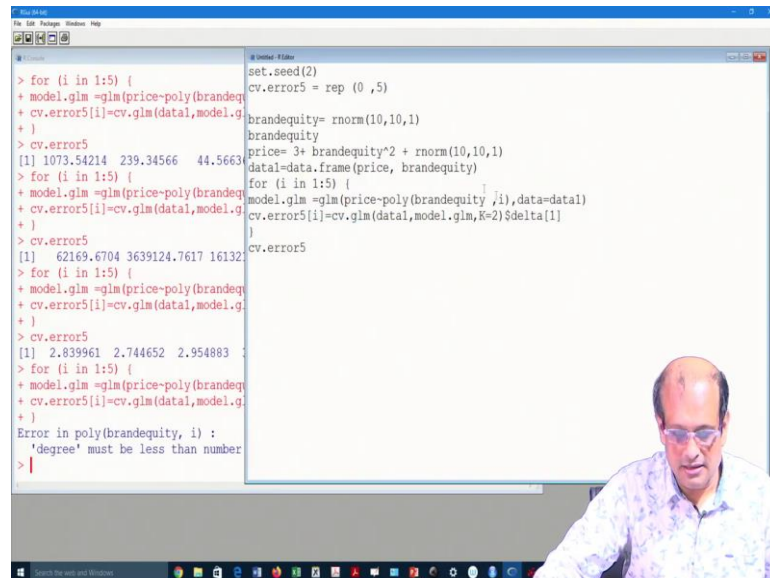
(Refer Slide Time: 21:15)



So, there are many variables I will use in the next class some of these variables to predict prices ok. So, this is my data set which I will be using. So, you can have a look at the

link, the link is also shared in the PPT as you can see the link is shared in the PPT and next class we move on and look at this link; ok.

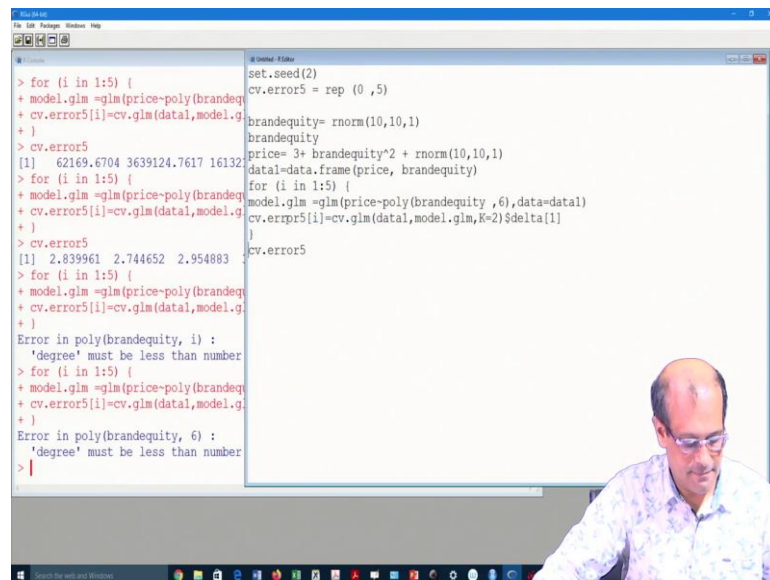
(Refer Slide Time: 21:46)



```
R Console Output:  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 5), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
> cv.error5  
[1] 1073.54214 239.34566 44.56663  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 4), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
> cv.error5  
[1] 62169.6704 3639124.7617 16132.16132  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 3), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
> cv.error5  
[1] 2.839961 2.744652 2.954883  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 2), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
Error in poly(brandequity, i) :  
'degree' must be less than number of observations  
> |
```

So, I will now talk in terms of what happens if I were to make this as, say 6.

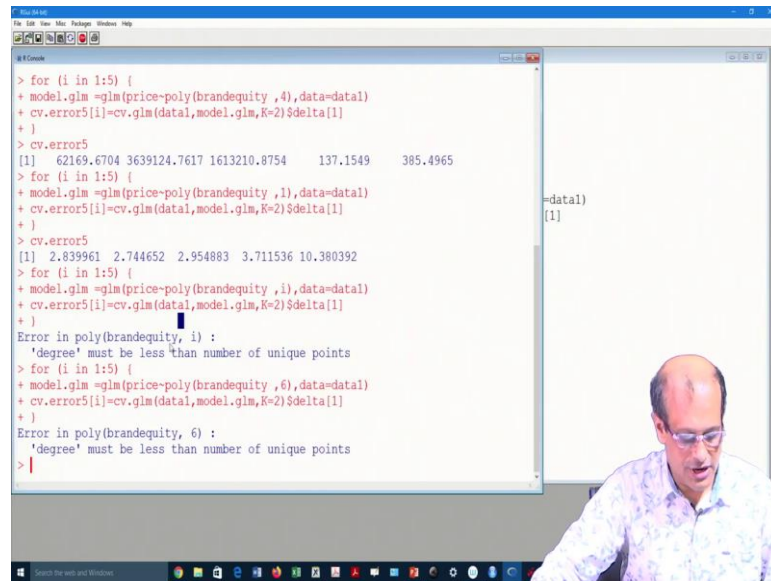
(Refer Slide Time: 21:56)



```
R Console Output:  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 6), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
> cv.error5  
[1] 62169.6704 3639124.7617 16132.16132  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 5), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
> cv.error5  
[1] 2.839961 2.744652 2.954883  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 4), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
Error in poly(brandequity, i) :  
'degree' must be less than number of observations  
> for (i in 1:5) {  
+ model.glm = glm(price~poly(brandequity, 3), data=data1)  
+ cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]  
+ }  
Error in poly(brandequity, 6) :  
'degree' must be less than number of observations  
> |
```

This is what I wanted you to look at.

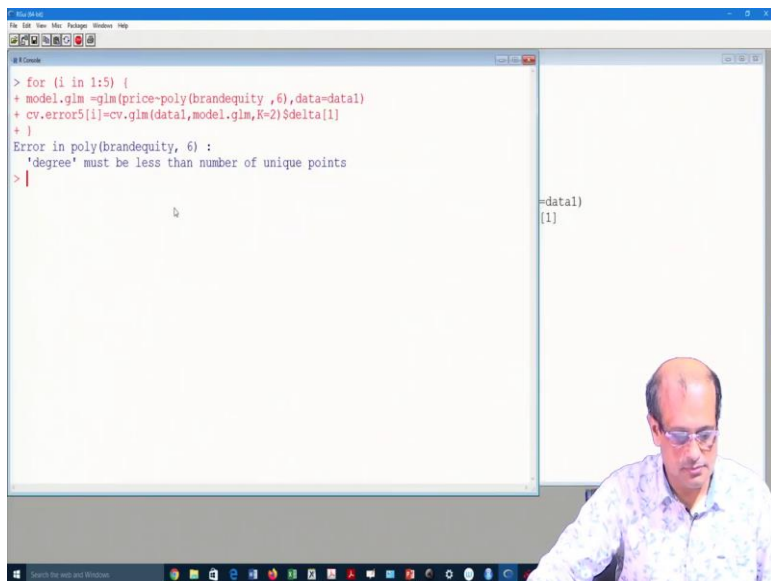
(Refer Slide Time: 22:01)



```
R Console
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,4),data=datal)
+ cv.error5[i]=cv.glm(datal,model.glm,K=2)$delta[1]
+ }
> cv.error5
[1] 62169.6704 3639124.7617 1613210.8754 137.1549 385.4965
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,1),data=datal)
+ cv.error5[i]=cv.glm(datal,model.glm,K=2)$delta[1]
+ }
> cv.error5
[1] 2.839961 2.744652 2.954883 3.711536 10.380392
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,i),data=datal)
+ cv.error5[i]=cv.glm(datal,model.glm,K=2)$delta[1]
+ }
Error in poly(brandequity, i) :
'degree' must be less than number of unique points
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,6),data=datal)
+ cv.error5[i]=cv.glm(datal,model.glm,K=2)$delta[1]
+ }
Error in poly(brandequity, 6) :
'degree' must be less than number of unique points
> |
```

So, I will clear the console and make this as 6 and see what happens.

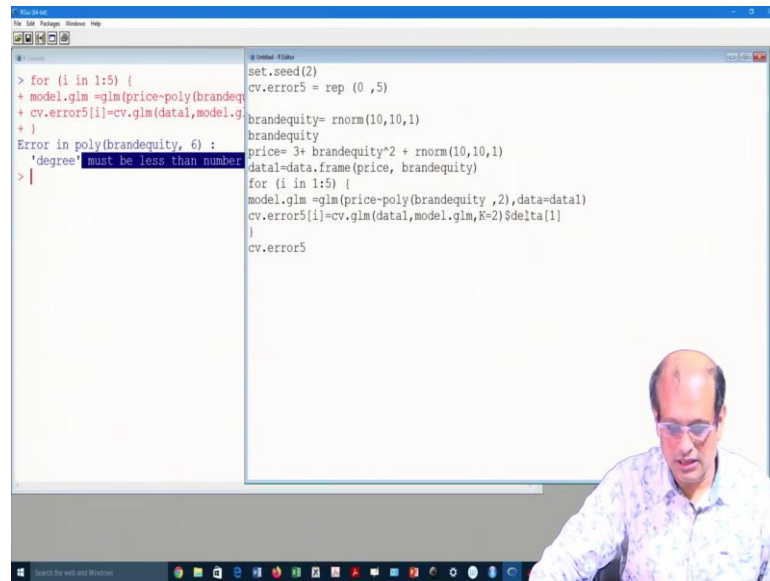
(Refer Slide Time: 22:12)



```
R Console
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandequity ,6),data=datal)
+ cv.error5[i]=cv.glm(datal,model.glm,K=2)$delta[1]
+ }
Error in poly(brandequity, 6) :
'degree' must be less than number of unique points
> |
```

So, this is very important. So, degree must be less than the number of unique points. So, the point we are looking at is your points are 5 which you have generated from the data. So, the degree of the equation in the polynomial regression equation has to be less than the unique points; fine.

(Refer Slide Time: 22:41)



```
> for (i in 1:5) {
+ model.glm =glm(price~poly(brandeq
+ cv.error5[i]=cv.glm(data1,model.g
+ }
Error in poly(brandequity, 6) :
'degree' must be less than number
> |

set.seed(2)
cv.error5 = rep (0 , 5)
brandequity= rnorm(10,10,1)
brandequity
price= 3+ brandequity^2 + rnorm(10,10,1)
data1=data.frame(price, brandequity)
for (i in 1:5) {
model.glm =glm(price~poly(brandequity ,2),data=data1)
cv.error5[i]=cv.glm(data1,model.glm,K=2)$delta[1]
}
cv.error5
```

So, we have done a basic implementation; right; I started with 2; I so, keep this as two and I was able to get the cross validation errors and the cross validation errors which we got initially were listed down; right. So, we were able to get the cross validation error for a model; ok. So, now, in the next class what we will do; we will try to do the implementation of the Boston housing data set and we will also look at some of the other concepts in pricing.

Thank you!