

Predictive Analytics - Regression and Classification
Prof. Sourish Das
Department of Mathematics
Chennai Mathematical Institute

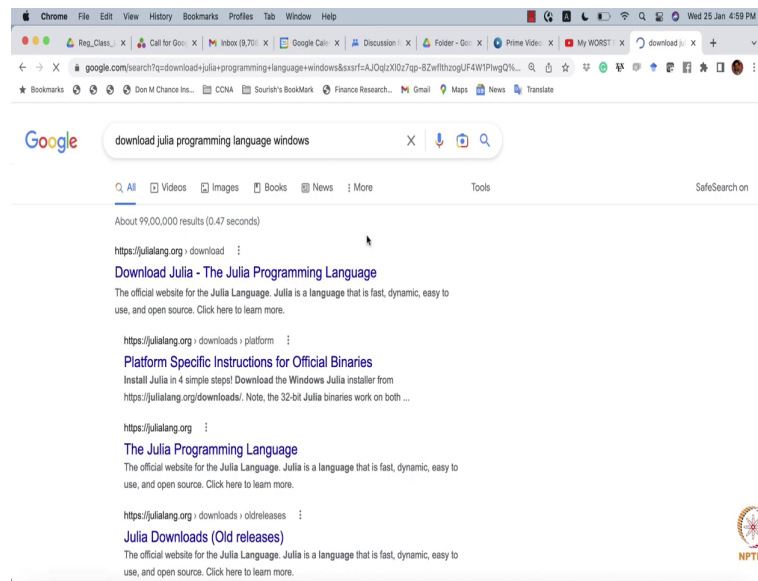
Lecture - 19
Hands-on with Julia

Hello all, in this video I am going to introduce you to how to do Regression Analysis using Julia. Julia Programming Language is a new language and it is comparatively new compared to R or Python R and Python was released the first version was released in 1990s whereas; Julia's first version was released in 2018.

So, Julia is a comparatively new language, the advantage of Julia is I find two fold; one is expressivity that is if you want to write a particular code or something it is much easier in terms of its expression and second advantage I find that Julia programming language tries to take the advantages of new compiler theory new advantages and advancements of the compiler theory.

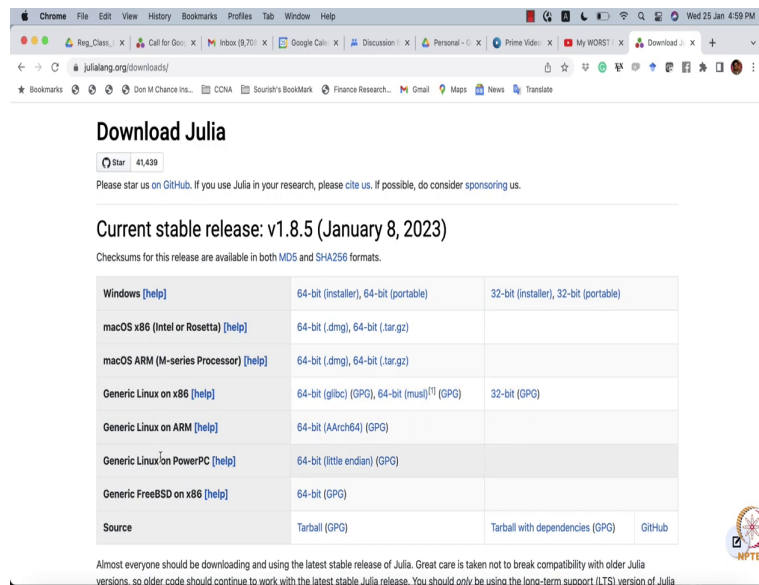
So, it is it tries to implement the all the you know functionality faster than R and Python. So, when we are trying to crunch lot of large big data, the Julia turns out to be very useful. So, I am going to start how to do regression and classification particularly regression in this video using Julia.

(Refer Slide Time: 01:58)



So, first I will start a Google site and in Google if you just say download Julia programming language or Julia for windows you can write Julia programming language or you can say for windows or then you will get like you know it will take you to the latest version of Julia.

(Refer Slide Time: 02:38)



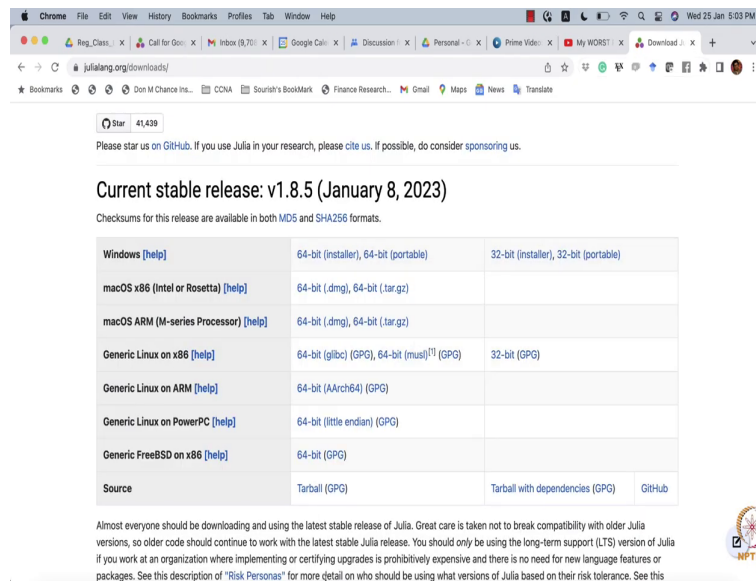
The screenshot shows the JuliaLang website's download page. At the top, it says "Download Julia" with a GitHub star count of 41,499. Below that, it states "Current stable release: v1.8.5 (January 8, 2023)". A table lists download options for various operating systems and architectures. At the bottom, there is a note about using the latest stable release and a logo for NPTEL.

Operating System / Architecture	64-bit (installer), 64-bit (portable)	32-bit (installer), 32-bit (portable)
Windows [help]	64-bit (installer), 64-bit (portable)	32-bit (installer), 32-bit (portable)
macOS x86 (Intel or Rosetta) [help]	64-bit (.dmg), 64-bit (.tar.gz)	
macOS ARM (M-series Processor) [help]	64-bit (.dmg), 64-bit (.tar.gz)	
Generic Linux on x86 [help]	64-bit (glibc) (GPG), 64-bit (musl) ^[1] (GPG)	32-bit (GPG)
Generic Linux on ARM [help]	64-bit (AArch64) (GPG)	
Generic Linux on PowerPC [help]	64-bit (little endian) (GPG)	
Generic FreeBSD on x86 [help]	64-bit (GPG)	
Source	Tarball (GPG)	Tarball with dependencies (GPG) GitHub

Almost everyone should be downloading and using the latest stable release of Julia. Great care is taken not to break compatibility with older Julia versions, so older code should continue to work with the latest stable Julia release. You should only be using the long-term support (LTS) version of Julia.

And here you get windows Mac generic Linux typically if you are using a Ubuntu system this version should work.

(Refer Slide Time: 02:50)



Star 41,438


Please star us on GitHub. If you use Julia in your research, please cite us. If possible, do consider sponsoring us.

Current stable release: v1.8.5 (January 8, 2023)

Checksums for this release are available in both MD5 and SHA256 formats.

Windows [help]	64-bit (installer), 64-bit (portable)	32-bit (installer), 32-bit (portable)
macOS x86 (intel or Rosetta) [help]	64-bit (.dmg), 64-bit (.tar.gz)	
macOS ARM (M-series Processor) [help]	64-bit (.dmg), 64-bit (.tar.gz)	
Generic Linux on x86 [help]	64-bit (glibc) (GPG), 64-bit (musl) ^[1] (GPG)	32-bit (GPG)
Generic Linux on ARM [help]	64-bit (AArch64) (GPG)	
Generic Linux on PowerPC [help]	64-bit (little endian) (GPG)	
Generic FreeBSD on x86 [help]	64-bit (GPG)	
Source	Tarball (GPG)	Tarball with dependencies (GPG) GitHub

Almost everyone should be downloading and using the latest stable release of Julia. Great care is taken not to break compatibility with older Julia versions, so older code should continue to work with the latest stable Julia release. You should only be using the long-term support (LTS) version of Julia if you work at an organization where implementing or certifying upgrades is prohibitively expensive and there is no need for new language features or packages. See this description of "Risk Personas" for more detail on who should be using what versions of Julia based on their risk tolerance. See this



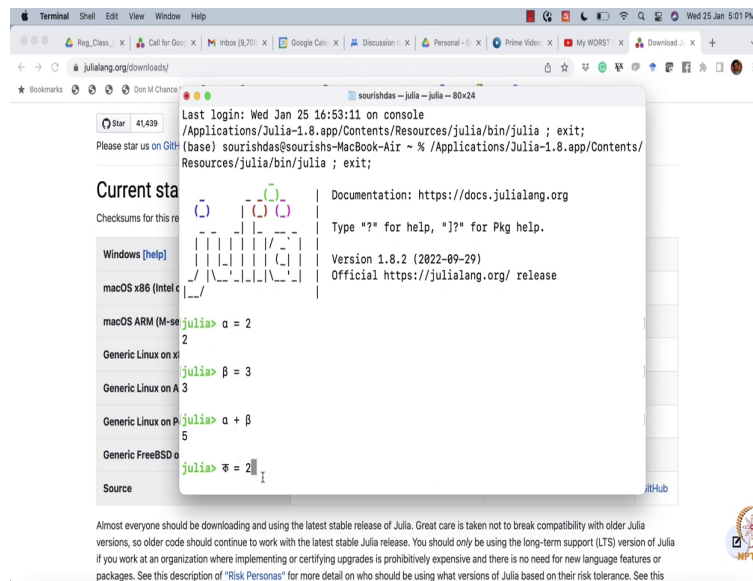
So, I am using Mac Book. So, I with M- series processor so, I my in my system I am going to use this Julia.

(Refer Slide Time: 03:08)



Now, if you open Julia then it will basic Julia console will open in terminal, sorry that.

(Refer Slide Time: 03:15)



```
Last login: Wed Jan 25 16:53:11 on console
/Applications/Julia-1.8.app/Contents/Resources/julia/bin/julia ; exit;
(base) sourishdas@sourishdas-MacBook-Air ~ % /Applications/Julia-1.8.app/Contents/
Resources/julia/bin/julia ; exit;

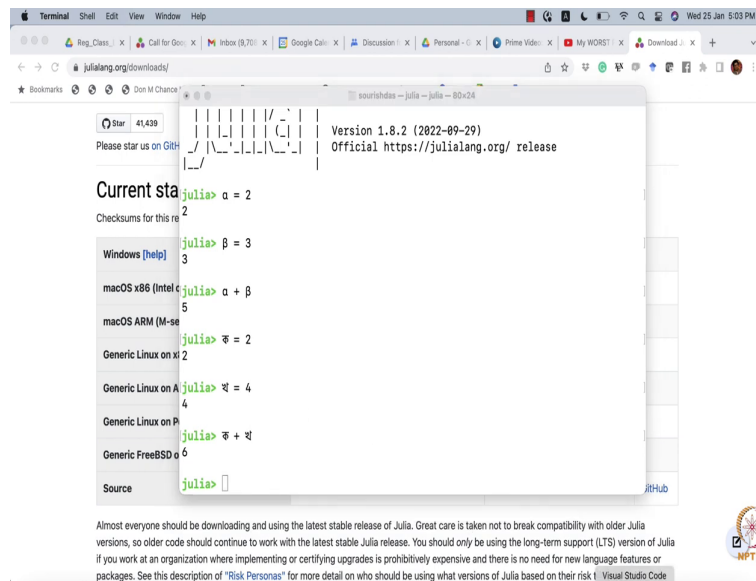
Current status
┌──────────┴──────────┐
┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐
│   │ │   │ │   │ │   │ │   │ │   │ │   │ │   │ │   │ │   │
└───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘
Documentation: https://docs.julialang.org
Type "?" for help, "!" for Pkg help.
Version 1.8.2 (2022-09-29)
Official https://julialang.org/ release

macOS x86 (Intel)  julia> alpha = 2
2
macOS ARM (M-series) julia> beta = 3
3
Generic Linux on x86_64 julia> alpha + beta
5
Generic Linux on ARM64 julia> alpha + beta
5
Generic FreeBSD on amd64 julia> alpha + beta
5
Source
```

So, it will open in terminal. So, that is how it will behave. So, for example, if I want to say alpha and then press tab then it give me the alpha and then I can use assign 2 and then if I say beta and tab and I say 3 then it gives me if alpha and beta 2 variables now we have defined and then. So, let me just increase the size. So, and then if I just say alpha plus beta then it gives me 5.

So, alpha beta you can defined as variable here because Julia do allow Unicode. In fact, certain advantages is that if you can use some you know since it is do Unicode I have Bengali you know script installed here. So, if I just say [FL] in it is a Bengali script and so, say 2.

(Refer Slide Time: 04:41)

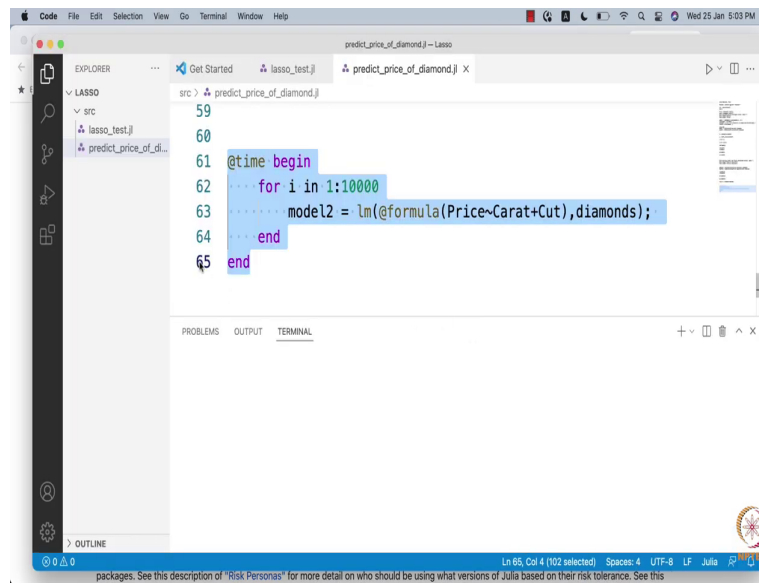


```
Terminal Shell Edit View Window Help
julia.org/downloads/
41,438
Please star us on GitHub
Version 1.8.2 (2022-09-29)
Official https://julia.org/ release
Current stable release: alpha = 2
Checksums for this release:
Windows [help] 3
macOS x86 (Intel) alpha + beta 5
macOS ARM (M-series) alpha + beta 2
Generic Linux on x86_64 alpha + beta 4
Generic Linux on ARM alpha + beta 6
Generic FreeBSD alpha + beta 6
Source [help]
```

And then [FL] equal to say 4 and then [FL] plus [FL] sorry [FL] it is giving me 6. So. In fact, if you are if you have Hindi script or if you have Kannada script or Tamil script you can install it and you can define your variable mean width. So, if you have any you can define your variables in using you know your local regional language and this I find is very helpful in terms of expressing the equations and other things.

So, but you know using writing everything in the console is bit difficult I mean when typical I generally use a VS code.

(Refer Slide Time: 05:51)



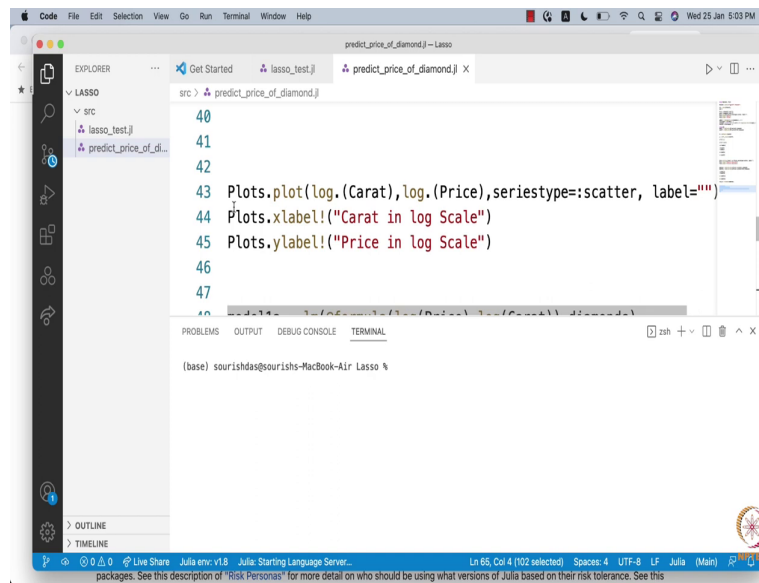
The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project structure with folders 'LASSO' and 'src', and files 'lasso_test.jl' and 'predict_price_of_diamond.jl'. The main editor window displays the following Julia code:

```
59  
60  
61 @time begin  
62     for i in 1:10000  
63         model2 = lm(@formula(Price~Carat+Cut), diamonds);  
64     end  
65 end
```

The status bar at the bottom indicates 'Ln 65, Col 4 (102 selected) Spaces: 4 UTF-8 LF Julia'. A small warning icon is visible in the bottom right corner of the editor area.

And in the VS code you have to go to the VS code is very useful in terms.

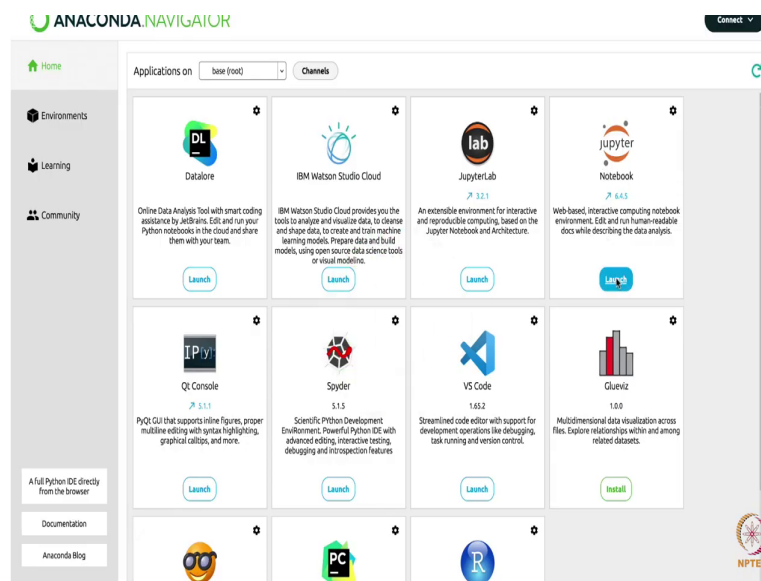
(Refer Slide Time: 05:56)



```
40  
41  
42  
43 Plots.plot(log.(Carat), log.(Price), seriestype=:scatter, label="")  
44 Plots.xlabel!("Carat in log Scale")  
45 Plots.ylabel!("Price in log Scale")  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

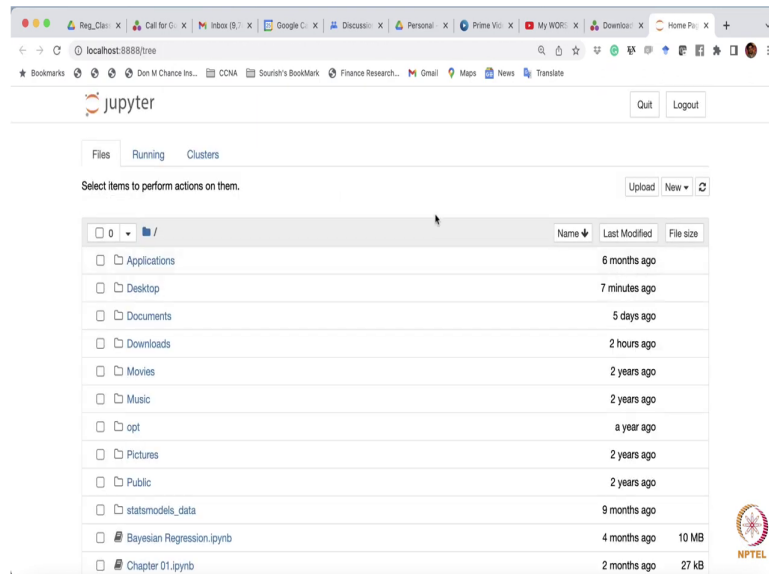
So, when I do some you know project based development I typically rely on VS code. But for teaching I find the you know I find Julia Jupyter Notebook is very useful.

(Refer Slide Time: 06:19)



So, I am going to open Jupyter Notebook, here is my I am opening my Anaconda fast.

(Refer Slide Time: 06:22)

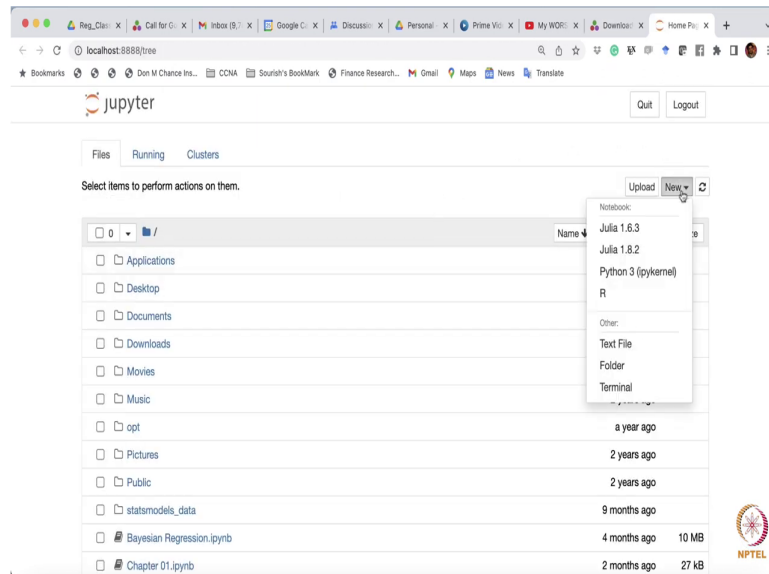


The screenshot shows a web browser window displaying the JupyterLab interface. The browser's address bar shows the URL `localhost:8888/tree`. The JupyterLab header includes the logo and "Quit" and "Logout" buttons. Below the header, there are tabs for "Files", "Running", and "Clusters". The "Files" tab is active, showing a file browser view. The interface prompts the user to "Select items to perform actions on them." and includes "Upload" and "New" buttons. The file browser displays a list of files and folders with columns for "Name", "Last Modified", and "File size".

	Name	Last Modified	File size
<input type="checkbox"/>	/		
<input type="checkbox"/>	Applications	6 months ago	
<input type="checkbox"/>	Desktop	7 minutes ago	
<input type="checkbox"/>	Documents	5 days ago	
<input type="checkbox"/>	Downloads	2 hours ago	
<input type="checkbox"/>	Movies	2 years ago	
<input type="checkbox"/>	Music	2 years ago	
<input type="checkbox"/>	opt	a year ago	
<input type="checkbox"/>	Pictures	2 years ago	
<input type="checkbox"/>	Public	2 years ago	
<input type="checkbox"/>	statsmodels_data	9 months ago	
<input type="checkbox"/>	Bayesian Regression.ipynb	4 months ago	10 MB
<input type="checkbox"/>	Chapter 01.ipynb	2 months ago	27 kB

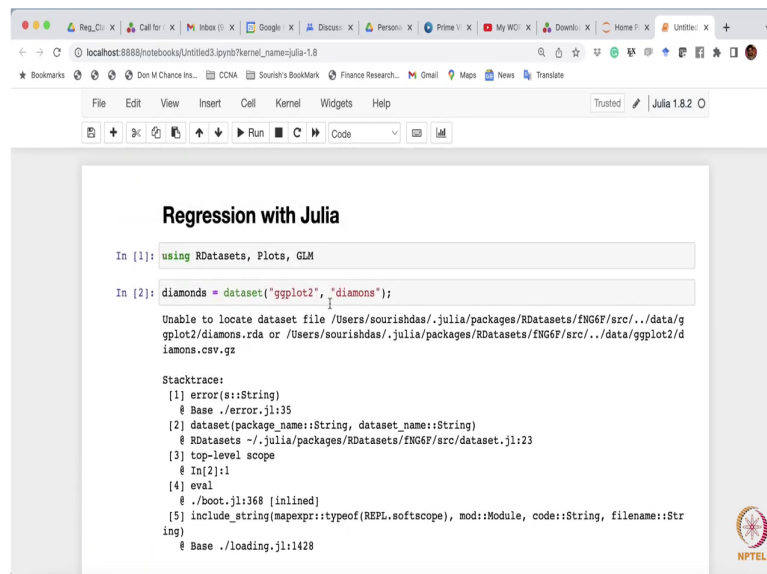
Now I am going to launch my Jupyter Notebook.

(Refer Slide Time: 06:28)



And here you can see Jupyter 1.8.2 version.

(Refer Slide Time: 06:31)



The screenshot shows a Jupyter Notebook interface with a browser window at the top. The notebook title is "Regression with Julia". The code cell contains the following Julia code:

```
In [1]: using RDatasets, Plots, GLM

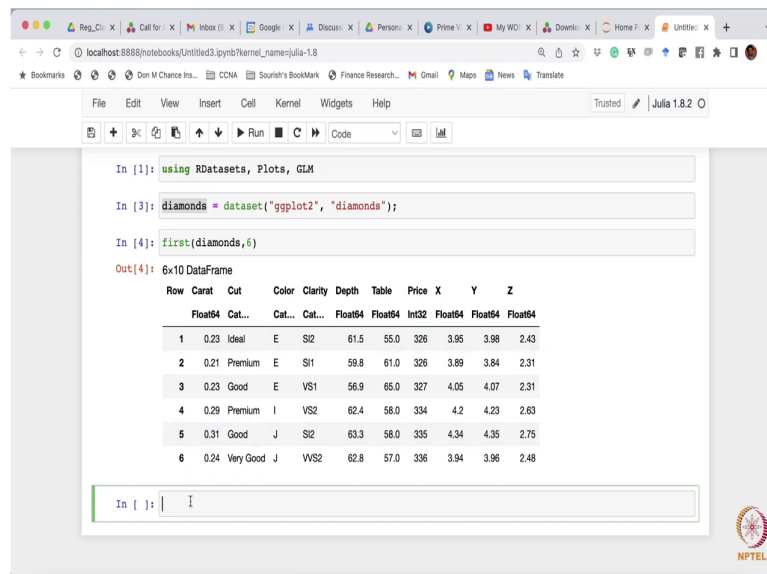
In [2]: diamonds = dataset("ggplot2", "diamonds");
```

The code execution resulted in an error: "Unable to locate dataset file /Users/sourishdas/.julia/packages/RDatasets/ENGGP/src/./data/ggplot2/diamonds.rda or /Users/sourishdas/.julia/packages/RDatasets/ENGGP/src/./data/ggplot2/diamonds.csv.gz". The stack trace shows the error originated from the `dataset` function in the `RDatasets` package.

I have and now I am going to use the Jupyter Notebook in our an learning the regression with Julia. So, first thing I will do I will first code block I am going to create as a markdown regression with Julia. So, if we just done it, it will give me the heading then first thing I will do I will I am just going to you know call some R data sets these are the packages these are the Julia packages that I am going to call R data sets, Plots and GLM.

Now, what R data sets does it is a Julia package which essentially from the different popular R packages it takes the data set and gives you the data here. So, all the popular data sets like empty carts data set and the diamond data set in the “gg plot 2” all those data set is automatically available in Julia. So, this is a very useful I find this is very useful and then what I am going to do, I am going to call the diamond dataset from ggplot2 ggplot2 ‘ggplot2’, I am going to call “diamonds” data set.

(Refer Slide Time: 08:49)



```
In [1]: using RDatasets, Plots, GLM

In [3]: diamonds = dataset("ggplot2", "diamonds");

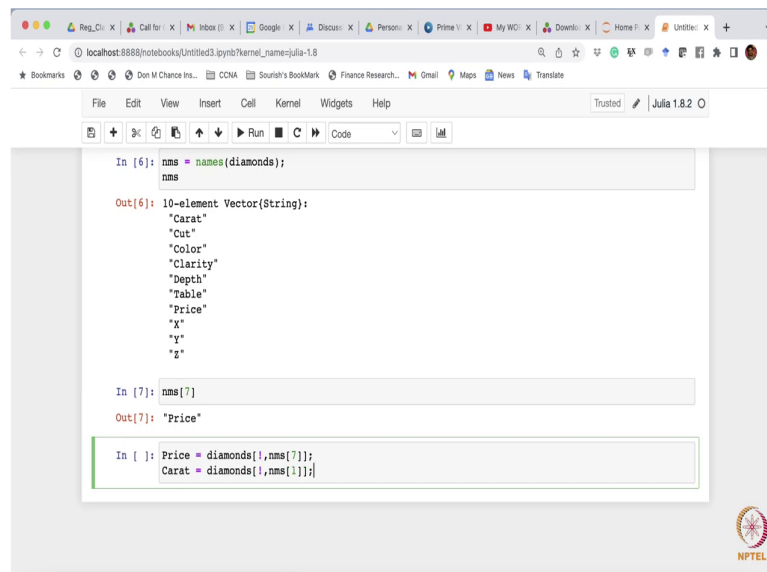
In [4]: first(diamonds, 6)

Out[4]: 6x10 DataFrame
  Row  Carat  Cut      Color Clarity Depth  Table  Price X    Y    Z
  Float64 Cat... Cat... Cat... Float64 Float64 Int32 Float64 Float64 Float64
-----
  1    0.23  Ideal  E      SI2    61.5   55.0  326   3.95   3.98  2.43
  2    0.21  Premium E      SI1    59.8   61.0  326   3.89   3.84  2.31
  3    0.23  Good   E      VS1    56.9   65.0  327   4.05   4.07  2.31
  4    0.29  Premium I      VS2    62.4   58.0  334   4.2    4.23  2.63
  5    0.31  Good   J      SI2    63.3   58.0  335   4.34   4.35  2.75
  6    0.24  Very Good J      VS2    62.8   57.0  336   3.94   3.96  2.48
```

So, let me just run it sorry here is a spelling mistake yeah there is no error now I believe. So, first and if I just say first comma say 6 and run it. So, what it will do it will just plot the first 6 rows of the data set in the diamond of the diamond dataset. Now, what we are seeing that diamond data set has one to about 10 columns 3 4 5 6 7 8 9 10 carat, cut, color, clarity, depth, table, price of the diamond and then X, Y, Z there is a concept of table of the diamond and what is the length, breadth and height of the diamond.

So, those dimensions are given here X, Y, Z and these all have very important effect on the pricing of the diamond. So, first if you just so, and it is read as a data frame. So, like pandas data frame in Python and data frame in art Julia also has a data frame and this data frame is also very useful, this data structure particular data frame is very useful. Next, if I just say names equal to names of diamonds. So, it gives us the names of the all columns ok.

(Refer Slide Time: 10:32)



```
In [6]: nms = names(diamonds);
nms

Out[6]: 10-element Vector{String}:
 "Carat"
 "Cut"
 "Color"
 "Clarity"
 "Depth"
 "Table"
 "Price"
 "X"
 "Y"
 "Z"

In [7]: nms[7]

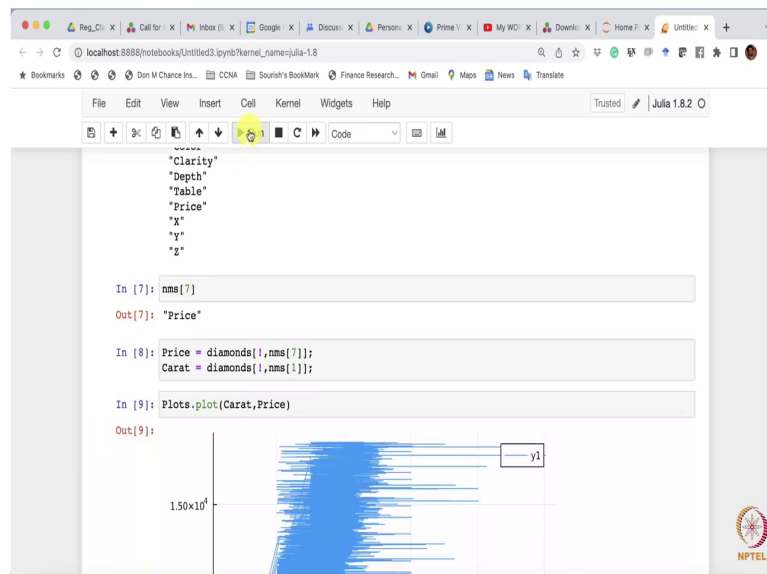
Out[7]: "Price"

In [ ]: Price = diamonds[1,nms[7]];
Carat = diamonds[1,nms[1]];
```

Now, similarly if I just say names the 7th one if I just call the 7th one. So, it is the 7th one element is price. So, now, name is a vector essentially it is a 10 element vector of all strings it is a vector of strings this essentially extracting all the column names of the data frame.

So, the first thing what I am going to do from the data this data frame I am going to take the price and the Carat, Carat is also a continuous variable like float 64 “Price” is integer 32, 32 bits integer, but let me just call these guys. So, Price equal to diamonds 7 and carat equals to diamonds comma. So, this exclamation marks means you take all the rows essentially nms 1 so, no error so far.

(Refer Slide Time: 12:37)



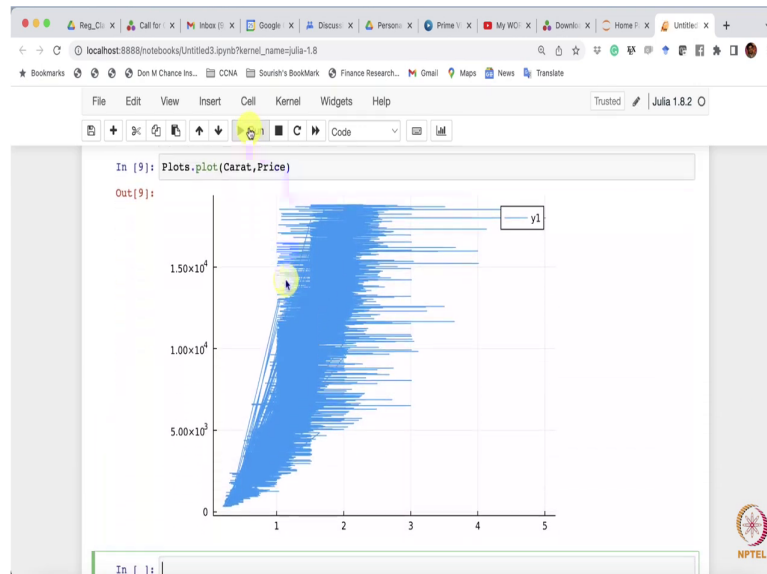
The screenshot shows a web-based Julia REPL interface. The browser address bar indicates the URL is localhost:8888. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with various icons. The code editor contains the following code:

```
.....  
"Clarity"  
"Depth"  
"Table"  
"Price"  
"x"  
"y"  
"z"  
  
In [7]: nms[7]  
Out[7]: "Price"  
  
In [8]: Price = diamonds[1,nms[7]];  
Carat = diamonds[1,nms[1]];  
  
In [9]: Plots.plot(Carat,Price)  
Out[9]:
```

The output of the last command is a scatter plot. The y-axis is labeled with 1.50×10^4 . The plot shows a dense collection of blue data points. A legend in the top right corner of the plot area shows a blue line segment labeled 'y1'. The NPTEL logo is visible in the bottom right corner of the interface.

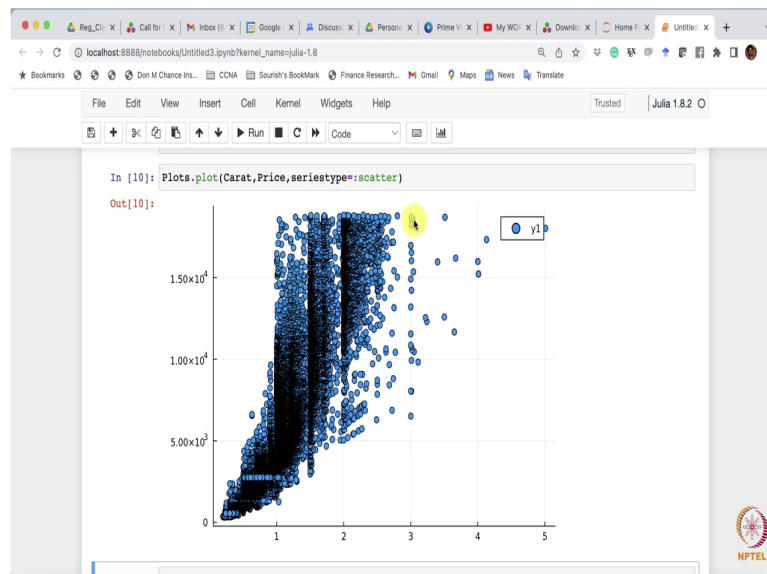
Now, what I am going to do I am going to run the from the Plots I am going to call the plot function ok Carat, comma Price if I just run this.

(Refer Slide Time: 12:54)



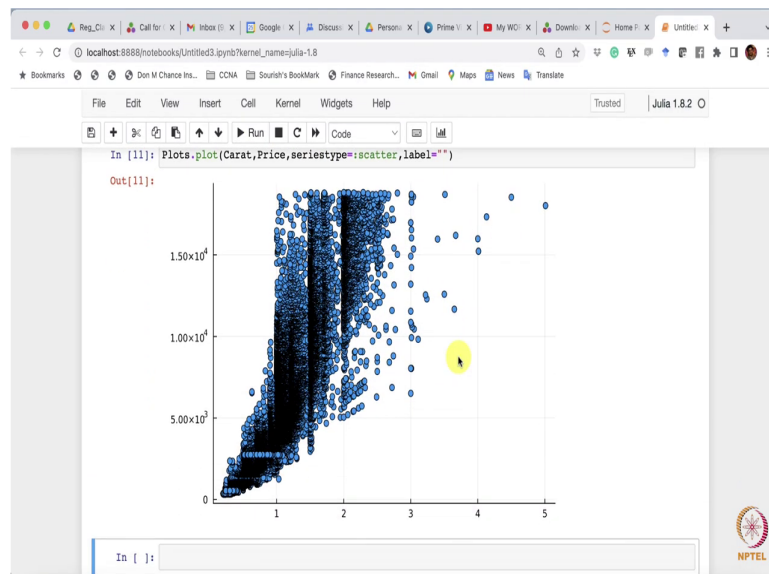
So, you can see it is plotted as sort of a series typically by default plot strives to plot as a you know time series. So, that is why it is slightly weird I rather I would like to see a scatter plot here.

(Refer Slide Time: 13:13)



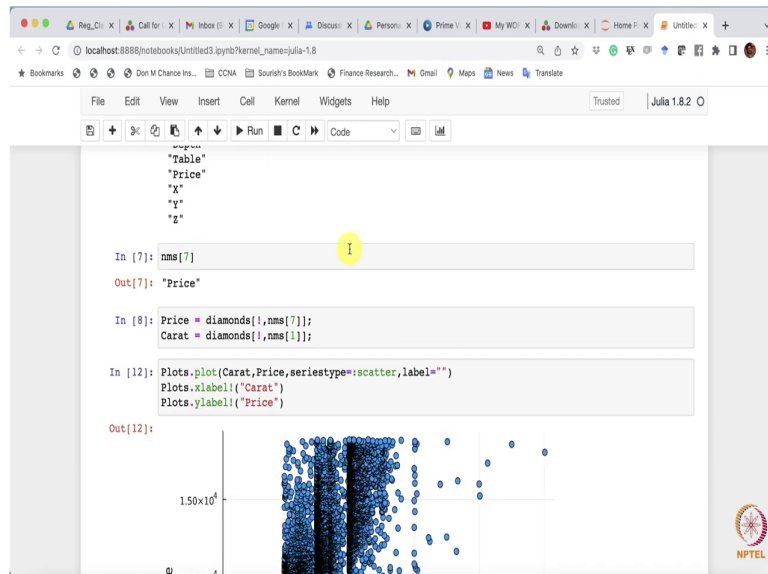
So, what you have to say you have to say a series type equals to colon scatter. So, now, it is giving you the scatter plot now, but it is saying y 1. So, I do not want the y 1 it is still treating it as a sort of a carat as the index here and the price as if y series time series.

(Refer Slide Time: 13:50)



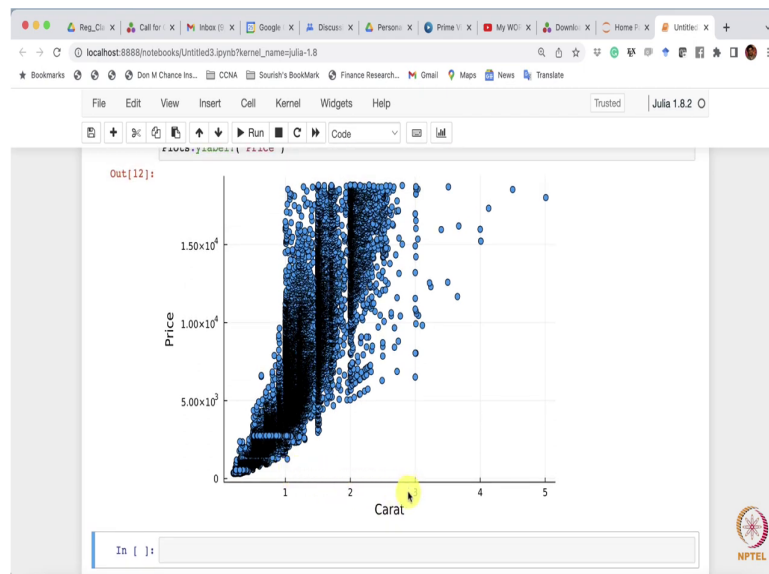
So, what I want I do not want the label. So, what I am going to do I am going to say that do not put me any level. But if you see the there is no labelling of the x axis of y axis.

(Refer Slide Time: 14:12)



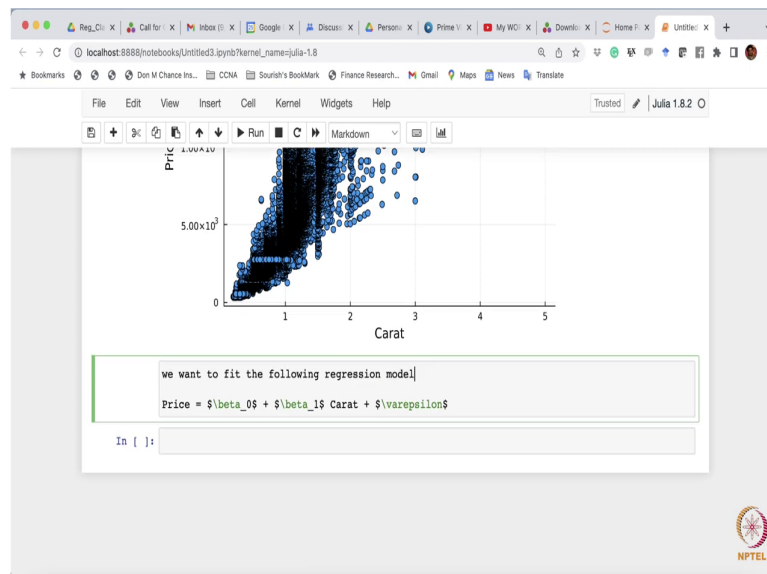
So, I have to put some labels `Plots.xlabel("Carat")` and `Plots.ylabel("Price")`.

(Refer Slide Time: 14:42)



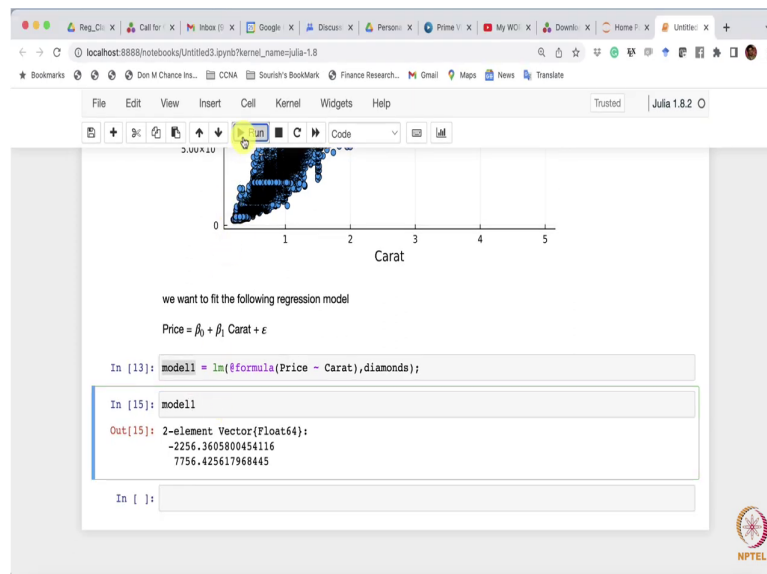
Now, if you run now you can see the price and the carat as the carat increases you can expect that the price will increase as well. Next, I want to fit linear regression may be between Carat and the Price.

(Refer Slide Time: 15:09)



So, what I will do what kind of model I will fit. So, I will fit Price equals to beta naught plus beta 1 times Carat plus varepsilon the error part. Now, if you run it this is the model I want to fit. So, we can just write it like this. Sorry, we want to fit we want to fit the following regression model ok.

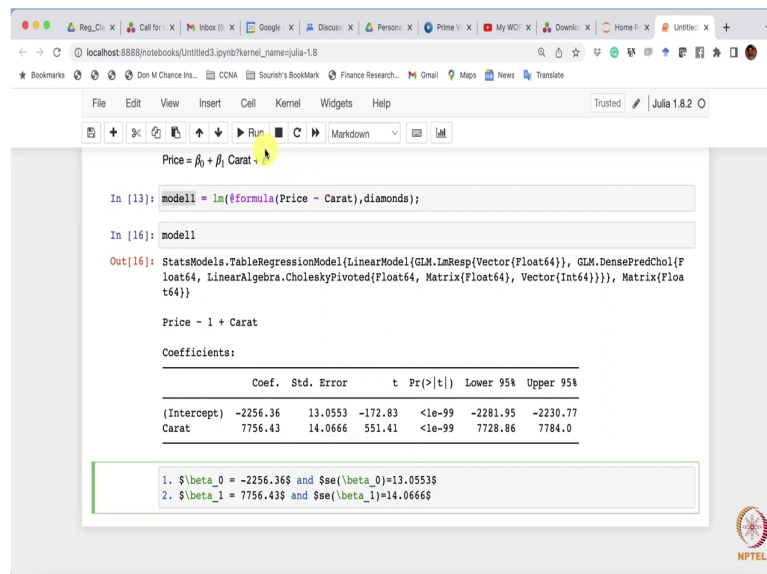
(Refer Slide Time: 16:12)



So, if we just run it. So, this is the model we want to fit. Fitting this model is almost same as fitting a regression model in R. So, so you just call model. So, let me just say model 1 we will try some other models as well. So, lm we call lm at the rate formula now in formula you have to keep what model you want to fit.

So, you say Price tilde Carat C a r a t and then you have to give the name of the data frame. So, in the name of the data frame where both variables are available ok. Now, if you just run it. So, it ran simply I will write model 1.

(Refer Slide Time: 17:14)



```
Price =  $\beta_0 + \beta_1$  Carat.  
  
In [13]: model1 = lm(@formula(Price ~ Carat), diamonds);  
  
In [16]: model1  
Out[16]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}}, Vector{Int64}}}}, Matrix{Float64}}  
  
Price ~ 1 + Carat  
  
Coefficients:  
  
          Coef.  Std. Error    t Pr(>|t|) Lower 95% Upper 95%  
-----  
(Intercept) -2256.36    13.0533  -172.83  <1e-99  -2281.95  -2230.77  
Carat        7756.43    14.0666   551.41  <1e-99   7728.86   7784.0  
  
1. $beta_0 = -2256.36$ and $se(beta_0)=13.0533$  
2. $beta_1 = 7756.43$ and $se(beta_1)=14.0666$
```

Let me just have run it. So, you can see that it has fitted a model with the coefficient this is the beta naught. So, we can say the beta naught value is beta naught. So, we can say beta naught equals to negative 2 point this. 2nd is we can just copy this line here and beta 1 is 7756.43 and then and standard error of beta naught is equal to 13.0533 and standard error of beta 1 is 14.0666 ok.

(Refer Slide Time: 18:40)

```
Out[16]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}}, Vector{Int64}}}}, Matrix{Float64}}

Price ~ 1 + Carat

Coefficients:

      Coef. Std. Error    t Pr(>|t|) Lower 95% Upper 95%
-----
(Intercept) -2256.36   13.0553  -172.83 <1e-99  -2281.95  -2230.77
Carat        7756.43   14.0666   551.41 <1e-99   7728.86   7784.0

1.  $\hat{\beta}_0 = -2256.36$  and  $se(\hat{\beta}_0) = 13.0553$ 
2.  $\hat{\beta}_1 = 7756.43$  and  $se(\hat{\beta}_1) = 14.0666$ 
```

And then what we can do we can also say what is the t value if you see this negative 2.5 2256.36 divided by 13.0553.

(Refer Slide Time: 18:56)

```
LinearAlgebra.UnorderedPairs{Float64, Matrix{Float64}, Vector{Int64}}, Matrix{Float64}}

Price ~ 1 + Carat

Coefficients:

      Coef.  Std. Error    t Pr(>|t|) Lower 95% Upper 95%
-----
(Intercept) -2256.36    13.0553  -172.83 <1e-99  -2281.45  -2230.77
Carat        7756.43    14.0666   551.41 <1e-99   7728.86   7784.0

1. $beta_0 = -2256.36$ and $se(beta_0)=13.0553$
2. $beta_1 = 7756.43$ and $se(beta_1)=14.0666$

In [17]: -2256.36/13.0553
Out[17]: -172.83095754214764

In [ ]: |
```

You see this is negative 172.83 the t value is here.

(Refer Slide Time: 19:13)

```
Price = 1 + Coef

Coefficients:

```

	Coef.	Std. Error	t	Pr(> t)	Lower 95%	Upper 95%
(Intercept)	-2256.36	13.0553	-172.83	<1e-99	-2281.95	-2230.77
Carat	7756.43	14.0666	551.41	<1e-99	7728.86	7784.0

```

1. $beta_0 = -2256.36$ and $se(beta_0)=13.0553$
2. $beta_1 = 7756.43$ and $se(beta_1)=14.0666$

In [18]: -2256.36/13.0553
         7756.43 /14.0666

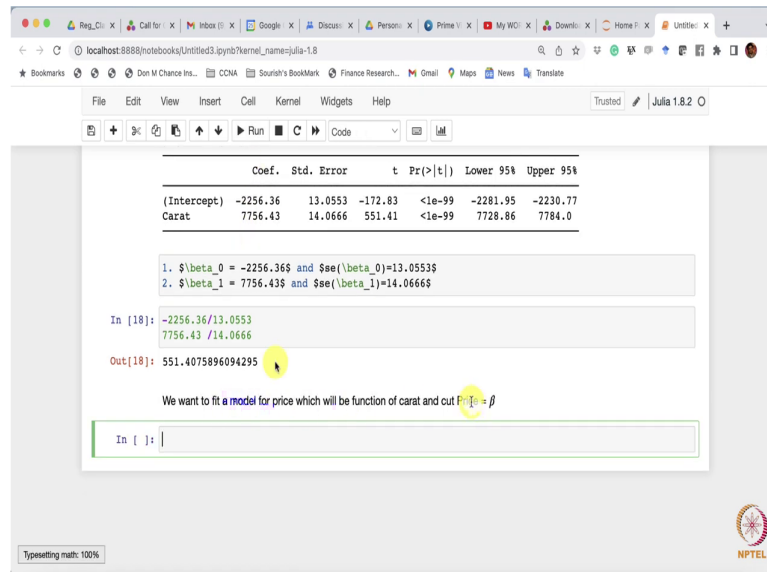
Out[18]: 551.4075896094295

We want to fit a model for price which will be function of carat and cut
Price = $beta$
```

And similarly, if you just take 7.56 divided by a standard error then you will get the t value is here 551. So, that is how the t value is being calculated and p value is calculated from the t distribution. The model now we want to fit another model, we want to fit say not only carat as a function of carat price as a function of carat, but we want to also fit the function of cut the second feature which is a categorical variable or string variable.

Now, how what model that will be. So, the model that will be is something like this. So, Price equal to beta naught. So, before that I want to write that we want to fit a model which fit a model for price which will be function of price which will be function of carat and cut.

(Refer Slide Time: 21:00)



The screenshot shows a Julia REPL interface with a table of regression coefficients and standard errors. The table is as follows:

	Coef.	Std. Error	t	Pr(> t)	Lower 95%	Upper 95%
(Intercept)	-2256.36	13.0553	-172.83	<1e-99	-2281.95	-2230.77
Carat	7756.43	14.0666	551.41	<1e-99	7728.86	7784.0

Below the table, there are two code snippets:

```
1.  $\beta_0 = -2256.36$  and  $\text{se}(\beta_0) = 13.0553$   
2.  $\beta_1 = 7756.43$  and  $\text{se}(\beta_1) = 14.0666$ 
```

The next input is:

```
In [18]: -2256.36/13.0553  
7756.43 /14.0666
```

The output is:

```
Out[18]: 551.4075896094295
```

Below the output, there is a text prompt:

We want to fit a model for price which will be function of carat and cut $F(\text{carat}) = \beta$

The final input is:

```
In [ ]: |
```

The NPTEL logo is visible in the bottom right corner.

So, now if you run it so, ok.

(Refer Slide Time: 21:05)

```
Conf. Std. Error t Pr(>|t|) Lower 95% Upper 95%
(Intercept) -2256.36 13.0553 -172.83 <1e-99 -2281.95 -2230.77
Carat 7756.43 14.0666 551.41 <1e-99 7728.86 7784.0
```

```
1.  $\beta_0 = -2256.36$  and  $\text{se}(\beta_0) = 13.0553$ 
2.  $\beta_1 = 7756.43$  and  $\text{se}(\beta_1) = 14.0666$ 
```

```
In [18]: -2256.36/13.0553
         7756.43 /14.0666
```

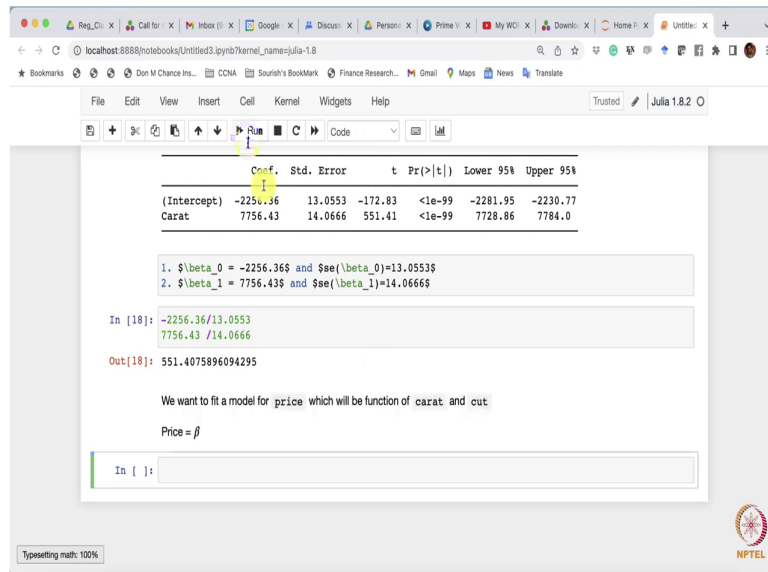
```
Out[18]: 551.4075896094295
```

```
We want to fit a model for "price" which will be function of "carat" and "cut"
Price =  $\beta_0 + \beta_1 \text{carat}$ 
```

```
In [ ]:
```

Let me just have a it was here. So, in fact, one thing we can do just to make sure that this price is actually coming from data it is a variable you can just put it like this.

(Refer Slide Time: 21:22)



The screenshot shows a Julia REPL window with the following content:

	Conf.	Std. Error	t	Pr(> t)	Lower 95%	Upper 95%
(Intercept)	-2256.36	13.0553	-172.83	<1e-99	-2281.95	-2230.77
Carat	7756.43	14.0666	551.41	<1e-99	7728.86	7784.0

```
1. \$\beta_0 = -2256.36$ and $se(\beta_0)=13.0553$  
2. \$\beta_1 = 7756.43$ and $se(\beta_1)=14.0666$  
  
In [18]: -2256.36/13.0553  
         7756.43 /14.0666  
  
Out[18]: 551.4075896094295  
  
We want to fit a model for price which will be function of carat and cut.  
  
Price =  $\beta$   
  
In [ ]:
```

And then it will be you know termed as Price ok.

(Refer Slide Time: 21:32)

(Intercept)	-2256.36	13.0553	-172.83	<1e-99	-2281.95	-2230.77
Carat	7756.43	14.0666	551.41	<1e-99	7728.86	7784.0

```
1.  $\beta_0 = -2256.36$  and  $\beta_1 = 13.0553$   
2.  $\beta_1 = 7756.43$  and  $\beta_2 = 14.0666$ 
```

```
In [18]: -2256.36/13.0553  
7756.43 /14.0666
```

```
Out[18]: 551.4075896094295
```

We want to fit a model for "Price" which will be function of "Carat" and "Cut"

```
"Price" =  $\beta_0$  +  $\beta_1$  "Carat" +  $\beta_2$  "Cut"
```

```
In [ ]:
```

Next is all we can say capital P this is capital C and this is capital C. Now, 'Price' is a function of beta naught plus beta 1 times 'Carat' maybe we just said like this plus beta 2 times Cut.

(Refer Slide Time: 22:16)

```
Out[18]: 551.4075896094295

We want to fit a model for Price which will be function of Carat and Cut
Price =  $\beta_0 + \beta_1$  Carat +  $\beta_2$  Cut

In [19]: model2 = lm(@formula(Price ~ Carat + Cut), diamonds);
model2

Out[19]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}}, Vector{Int64}}}}, Matrix{Float64}}

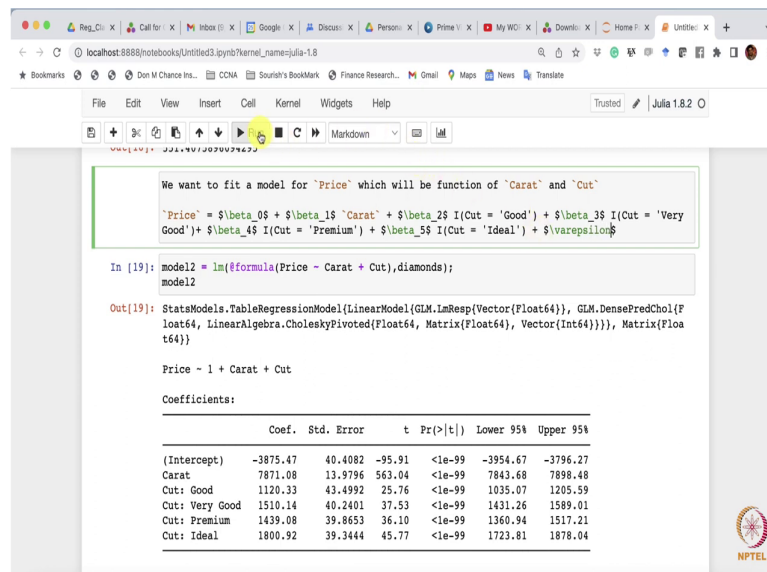
Price ~ 1 + Carat + Cut

Coefficients:

      Coef.  Std. Error    t    Pr(>|t|)  Lower 95%  Upper 95%
-----
(Intercept) -3875.47    40.4082  -95.91  <1e-99  -3954.67  -3796.27
Carat        7871.08    13.9796  563.04  <1e-99   7843.68  7898.48
Cut: Good    1120.33    43.4992   25.76  <1e-99  1035.07  1205.59
Cut: Very Good 1510.14    40.2401   37.53  <1e-99  1431.26  1589.01
Cut: Premium 1439.08    39.8653   36.10  <1e-99  1360.94  1517.21
Cut: Ideal   1800.92    39.3444   45.77  <1e-99  1723.81  1878.04
```

So, this is what I actually want to fit. So, ok we can do that. So, we will fit a second model. So, model 2 all you have to do you have to just add the end plus Cut and model 2 is this, but you see within Cut there are different values are there, good, very good, premium and ideal. There are you know four different values are coming actually there are five values of Cut fair, Good, Very Good and Premium and Ideal.

(Refer Slide Time: 23:09)



```
We want to fit a model for 'Price' which will be function of 'Carat' and 'Cut'

`Price` =  $\beta_0 + \beta_1 \text{Carat} + \beta_2 I(\text{Cut} = \text{'Good'}) + \beta_3 I(\text{Cut} = \text{'Very Good'}) + \beta_4 I(\text{Cut} = \text{'Premium'}) + \beta_5 I(\text{Cut} = \text{'Ideal'}) + \epsilon$ 

In [19]: model2 = lm(#formula(Price ~ Carat + Cut),diamonds);
model2

Out[19]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}}, Vector{Int64}}}}, Matrix{Float64}}

Price ~ 1 + Carat + Cut

Coefficients:



|                | Coef.    | Std. Error | t      | Pr(> t ) | Lower 95% | Upper 95% |
|----------------|----------|------------|--------|----------|-----------|-----------|
| (Intercept)    | -3875.47 | 40.4082    | -95.91 | <1e-99   | -3954.67  | -3796.27  |
| Carat          | 7871.08  | 13.9796    | 563.04 | <1e-99   | 7843.68   | 7898.48   |
| Cut: Good      | 1120.33  | 43.4992    | 25.76  | <1e-99   | 1035.07   | 1205.59   |
| Cut: Very Good | 1510.14  | 40.2401    | 37.53  | <1e-99   | 1431.26   | 1589.01   |
| Cut: Premium   | 1439.08  | 39.8653    | 36.10  | <1e-99   | 1360.94   | 1517.21   |
| Cut: Ideal     | 1800.92  | 39.3444    | 45.77  | <1e-99   | 1723.81   | 1878.04   |


```

So, what kind of model that is actually fitting? What it is fitting is it is indicator function of Cut equals to 'Good' plus beta 3 times indicator function for 'Very Good' plus beta 4 indicator function for 'Premium' plus beta 5 indicator function for 'Ideal' indicator function or you can say the one hot encoding or what dummy variable for each stream you are creating a you know indicator functions ok.

(Refer Slide Time: 24:23)

```

We want to fit a model for Price which will be function of Carat and Cut

Price = beta_0 + beta_1 * Carat + beta_2 * (Cut = 'Good') + beta_3 * (Cut = 'Very Good') + beta_4 * (Cut = 'Premium') + beta_5 * (Cut = 'Ideal') + epsilon

In [19]: model2 = lm(formula(Price ~ Carat + Cut), diamonds);
          model2

Out[19]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}}, Vector{Int64}}}}, Matrix{Float64}}

Price ~ 1 + Carat + Cut

Coefficients:

```

	Coef.	Std. Error	t	Pr(> t)	Lower 95%	Upper 95%
(Intercept)	-3875.47	40.4082	-95.91	<1e-99	-3954.67	-3796.27
Carat	7871.08	13.9796	563.04	<1e-99	7843.68	7898.48
Cut: Good	1127.39	43.4992	25.76	<1e-99	1035.07	1205.59
Cut: Very Good	1511.14	40.2401	37.53	<1e-99	1431.26	1589.01
Cut: Premium	1439.08	39.8653	36.10	<1e-99	1360.94	1517.21
Cut: Ideal	1800.92	39.3444	45.77	<1e-99	1723.81	1878.04

So, this will be a varepsilon ok. So, this is the model actually you are fitting. So, you have intercept then 1 2 3 4 5. So, six total six one intercept carried 1 2 3 4 good, very good, premium and 1 2 3 4 5 6 1 2 3 4 5 6 yeah there are six coefficient is being are being estimated beta naught beta 1, beta 2, beta 3, beta 4 and beta 5. So, there are six total six coefficient have been estimated.

Now, if suppose now this is the model these are the coefficient of the model these are the standard error. So, if you divide the standard error with by the coefficient by the standard error you get the 9, t value you can see the t values are very very large t values are very very small almost 0 it is less than you know point after 100 zeros 1, this is these are the 95 percent confidence intervals.

So, none of them include 0 so; that means, these all coefficients are have a significant effect on price. So, we are not going to I mean this is that is what the statistical analysis says. Now, after fitting the model what we are interested in for a particular new diamond can we estimate the expected price. For a new diamond ok sorry about that I think I just made a mistake ok yeah. So, this is fine and now this is the part I have to make markdown ok.

(Refer Slide Time: 26:23)

```
In [20]: model2 = lm(@Julia(Price ~ Carat + Cut),diamonds);
model2

Out[20]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}}, Vector{Int64}}}}, Matrix{Float64}}

Price ~ 1 + Carat + Cut

Coefficients:

          Coef.  Std. Error    t    Pr(>|t|)  Lower 95%  Upper 95%
-----
(Intercept) -3875.47    40.4082  -95.91  <1e-99  -3954.67  -3796.27
Carat        7871.08    13.9796  563.04  <1e-99   7843.68   7898.48
Cut: Good    1120.33    43.4992   25.76  <1e-99  1035.07  1205.59
Cut: Very Good 1510.14    40.2401   37.53  <1e-99  1431.26  1589.01
Cut: Premium 1439.08    39.8653   36.10  <1e-99  1360.94  1517.21
Cut: Ideal  1800.92    39.3444   45.77  <1e-99  1723.81  1878.04
```

For a new diamond of 2 carats with premium cut can we estimate the expected price?
`carat = 2` and `cut == Premium` then what is expected price?

For a new diamond for a new diamond with who diamond of 2 carats 2 carats and with premium cut can be estimate the expected price. So, suppose this is the question that we are interested, this is a new diamond which is not part of your diamond data set.

Now, given you have trained this model, you have fitted this model, now given this model you want to estimate the expected price of this new diamond which is of 2 carats and

premium cut so; that means, I can say 'carat' equals to 2 and 'cut' equals to Premium ok. Then what is the expected price? This is what my goal is from the model can we do that.

(Refer Slide Time: 28:03)

```

(Intercept)  -3875.47   40.4082  -95.91  <1e-99  -3954.67  -3796.27
Carat        7871.08   13.9796  563.04  <1e-99   7843.68   7898.48
Cut: Good    1120.33   43.4992   25.76  <1e-99   1035.07   1205.59
Cut: Very Good 1510.14   40.2401   37.53  <1e-99   1431.26   1589.01
Cut: Premium 1439.08   39.8653   36.10  <1e-99   1360.94   1517.21
Cut: Ideal   1800.92   39.3444   45.77  <1e-99   1723.81   1878.04
  
```

For a new diamond of 2 carats with premium cut can we estimate the expected price?

carat = 2 and cut == Premium then what is expected price?

```

In [21]: -3875.47 + 7871.08 * 2 + 1439.08 * 1
Out[21]: 13305.77
  
```

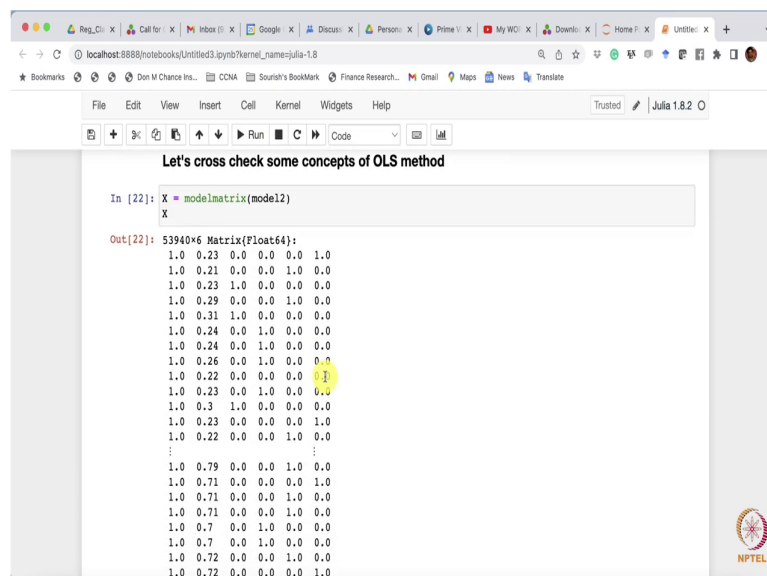
Let's cross check some concepts of OLS method

So, the first is. So, if we just plug in this so in fact, this is my whatever the beta naught this is my first value let me just copy this is my beta naught then plus beta 1. So, this is the beta 1 times carat plus beta 1 times carat equal to 2. Now, is it a good no so; that means, this is 0 you look at the model this is the model.

Since, it is not good so, it will be 0. So, 0 times better to whatever the value is 0. Very Good no so, this is 0. Premium yes, it is 1, if it is premium it is 1. So, beta 4 beta Premium value is 1439.08 plus 1439 times 1 ok. So, now if I run it so, 13305.77 is the price expected price I am not saying it will be the price.

So, because if you look into this so, 2 so, 13 is somewhere here part of this 13000 is somewhere here sorry somewhere here. So, it can be anywhere the range is very large you can see, but at least you can get some kind of expected price to be in that region. Now, we will take a little bit shift and we will go to some cross check of OLS method. So, let me do let us cross check some concepts of OLS method ok.

(Refer Slide Time: 30:11)



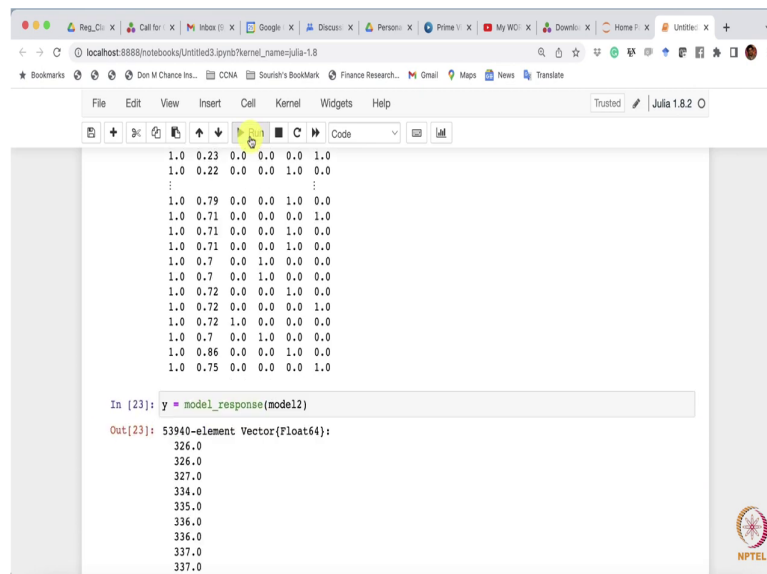
```
Let's cross check some concepts of OLS method

In [22]: X = modelmatrix(model2)
X

Out[22]: 53940×6 Matrix{Float64}:
 1.0  0.23  0.0  0.0  0.0  1.0
 1.0  0.21  0.0  0.0  0.0  1.0
 1.0  0.23  1.0  0.0  0.0  0.0
 1.0  0.29  0.0  0.0  0.0  1.0
 1.0  0.31  1.0  0.0  0.0  0.0
 1.0  0.24  0.0  1.0  0.0  0.0
 1.0  0.24  0.0  1.0  0.0  0.0
 1.0  0.26  0.0  1.0  0.0  0.0
 1.0  0.22  0.0  0.0  0.0  1.0
 1.0  0.23  0.0  1.0  0.0  0.0
 1.0  0.3  1.0  0.0  0.0  0.0
 1.0  0.23  0.0  0.0  0.0  1.0
 1.0  0.22  0.0  0.0  1.0  0.0
 ⋮
 1.0  0.79  0.0  0.0  1.0  0.0
 1.0  0.71  0.0  0.0  0.0  1.0
 1.0  0.71  0.0  0.0  1.0  0.0
 1.0  0.71  0.0  0.0  1.0  0.0
 1.0  0.7  0.0  1.0  0.0  0.0
 1.0  0.7  0.0  1.0  0.0  0.0
 1.0  0.72  0.0  0.0  1.0  0.0
 1.0  0.72  0.0  0.0  0.0  1.0
```

So, remember that I fit this model as model 2. So, what I am going to do, I am going to extract the design matrix. So, I can write each model all these model as a y equal to x beta plus model x beta plus epsilon. So, what I can do I can just say model matrix of model 2 and write it as a X ok let me just see.

(Refer Slide Time: 30:55)



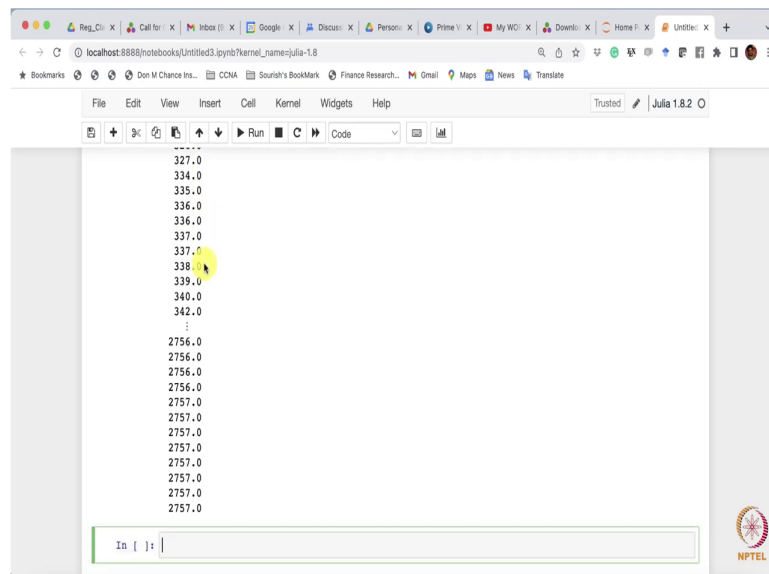
```
1.0 0.23 0.0 0.0 0.0 1.0
1.0 0.22 0.0 0.0 1.0 0.0
⋮
1.0 0.79 0.0 0.0 1.0 0.0
1.0 0.71 0.0 0.0 0.0 1.0
1.0 0.71 0.0 0.0 1.0 0.0
1.0 0.71 0.0 0.0 1.0 0.0
1.0 0.7 0.0 1.0 0.0 0.0
1.0 0.7 0.0 1.0 0.0 0.0
1.0 0.72 0.0 0.0 1.0 0.0
1.0 0.72 0.0 0.0 0.0 1.0
1.0 0.72 1.0 0.0 0.0 0.0
1.0 0.7 0.0 1.0 0.0 0.0
1.0 0.86 0.0 0.0 1.0 0.0
1.0 0.75 0.0 0.0 0.0 1.0
⋮
```

```
In [23]: y = model_response(model12)
Out[23]: 53940-element Vector{Float64}:
 326.0
 326.0
 327.0
 334.0
 335.0
 336.0
 336.0
 337.0
 337.0
 ⋮
```

So, this is my X matrix it has 53940 samples with 6 columns, now first column are all 1s because it corresponds to the intercept, second column are all corresponds to the values of carat ok, the third column corresponds to the cut equal to good wherever it got good it was it is there is 1 other rest of the places it 0.

The fourth column is corresponds to very good wherever it got very good it gives 1 rest of the places it gives 0 and similarly it is premium wherever it got premium it is says 1 rest of the places it is 0. If any value with the ideal condition then it will say 1 and rest of the places will be 0. So, that is how one hot encoding works or idea dummy variable creation or indicator variable creation works.

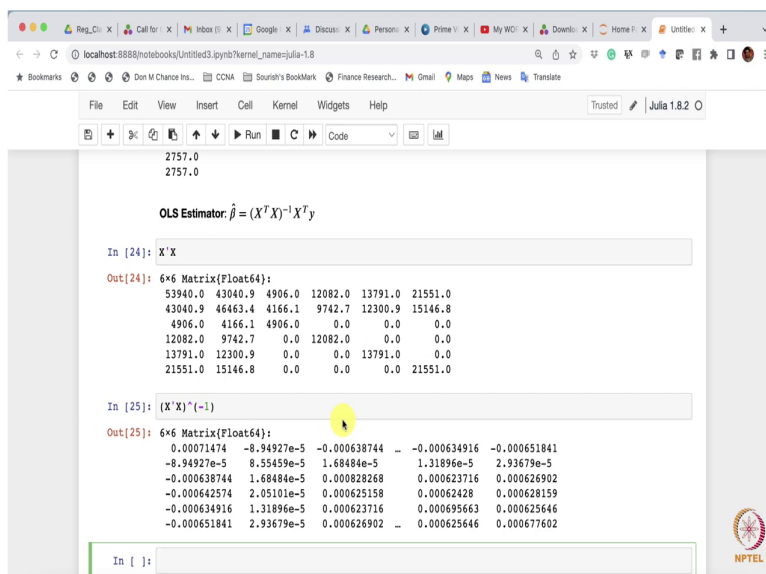
(Refer Slide Time: 32:33)



The screenshot shows a web browser window with a Julia REPL interface. The browser's address bar shows the URL `localhost:8888/notebooks/Untitled3.ipynb?kernel_name=julia-1.8`. The REPL window has a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. Below the menu bar is a toolbar with icons for file operations and a 'Run' button. The main area of the REPL displays a list of numerical values: `327.0`, `334.0`, `335.0`, `336.0`, `336.0`, `337.0`, `337.0`, `338.0`, `339.0`, `340.0`, `342.0`, followed by a vertical ellipsis `:`, and then `2756.0`, `2756.0`, `2756.0`, `2756.0`, `2756.0`, `2757.0`, `2757.0`, `2757.0`, `2757.0`, `2757.0`, `2757.0`, `2757.0`, `2757.0`, `2757.0`. A yellow highlight is visible on the `338.0` value. At the bottom of the REPL window, there is a prompt `In []:` followed by a cursor.

So, this is my design matrix ok. So, once I have the design matrix what I will require next thing is the response variable. So, y equal to `model_response` ok. So, these are my prices of 53940 prices ok.

(Refer Slide Time: 32:49)



```
2757.0
2757.0

OLS Estimator:  $\hat{\beta} = (X^T X)^{-1} X^T y$ 

In [24]: X'X
Out[24]: 6×6 Matrix{Float64}:
 53940.0  43040.9  4906.0  12082.0  13791.0  21551.0
 43040.9  46463.4  4166.1  9742.7  12300.9  15146.8
 4906.0   4166.1  4906.0    0.0    0.0    0.0
 12082.0  9742.7    0.0  12082.0    0.0    0.0
 13791.0  12300.9    0.0    0.0  13791.0    0.0
 21551.0  15146.8    0.0    0.0    0.0  21551.0

In [25]: (X'X)^(-1)
Out[25]: 6×6 Matrix{Float64}:
 0.00071474  -8.94927e-5  -0.000638744  ...  -0.000634916  -0.000651841
 -8.94927e-5  8.55459e-5  1.68484e-5  ...  1.31896e-5  2.93679e-5
 -0.000638744  1.68484e-5  0.000828268  ...  0.000623716  0.000626902
 -0.000642574  2.05101e-5  0.000625158  ...  0.00062428  0.000628159
 -0.000634916  1.31896e-5  0.000623716  ...  0.000695663  0.000625646
 -0.000651841  2.93679e-5  0.000626902  ...  0.000625646  0.000677602
```

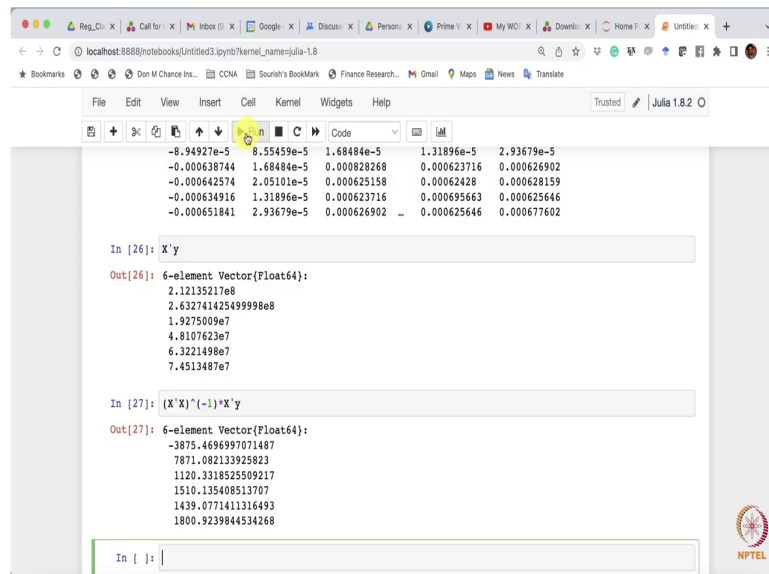
Now, what is the OLS estimator, the OLS Estimator is beta hat beta equals to X transpose X inverse X transpose y ok. So, this is the OLS estimator. So, what I am going to do first I am going to calculate X transpose X. So, I have already have X. So, I am going to calculate X transpose X. So, this is my X transpose X matrix.

So, you can see that you know there are about 53940 samples the first will be n then out of these samples the small block represents how many of them were good, how many of them were very good, how many of them were you know premium and how many of them were ideal. So, if you add them up I think you will get 53940.

Now, if you just say X transpose X inverse say minus 1. So, you get the inverse of X transpose X matrix and you can see this it is very intuitive the way you write the you know write it on your notebook, the way you write it in the you know in the blackboard almost you

can codify it you can code write your Julia code almost on the same way the way we have written it almost in the same way and then I am going to calculate X transpose y.

(Refer Slide Time: 35:14)



The screenshot shows a Julia REPL interface with the following content:

```
-8.94927e-5  8.55459e-5  1.68484e-5  1.31896e-5  2.93679e-5
-0.000638744  1.68484e-5  0.000828268  0.000623716  0.000626902
-0.000642574  2.05101e-5  0.000625158  0.00062428  0.000628159
-0.000634916  1.31896e-5  0.000623716  0.000695663  0.000625646
-0.000651841  2.93679e-5  0.000626902  0.000625646  0.000677602
```

In [26]: `X'y`

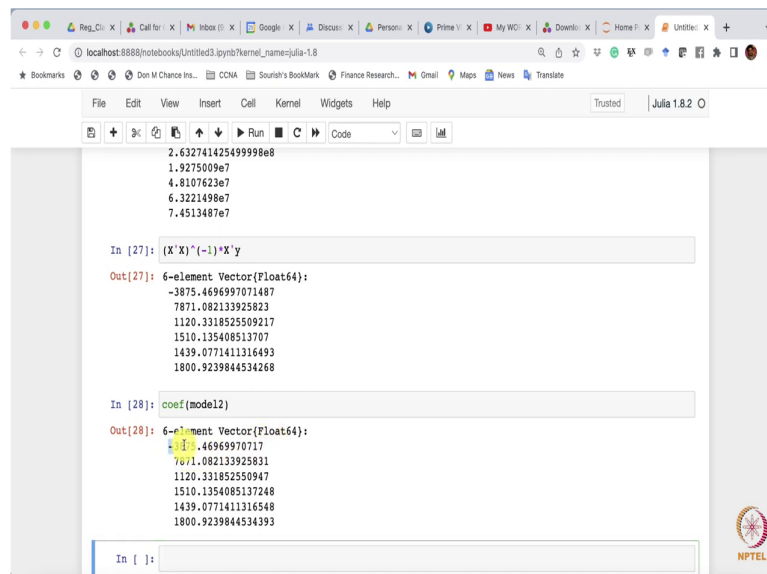
```
Out[26]: 6-element Vector{Float64}:
 2.12135217e8
 2.632741425499998e8
 1.9275009e7
 4.8107623e7
 6.3221490e7
 7.4513487e7
```

In [27]: `(X'X)^(-1)*X'y`

```
Out[27]: 6-element Vector{Float64}:
-3875.4696997071487
 7871.082133925823
 1120.3318525509217
 1510.135408513707
 1439.0771411316493
 1800.9239844534268
```

So, these are my X transpose y elements and now I am going to calculate X transpose X inverse X transpose y. So, X transpose X inverse times X transpose y. So, if you just. So, these are the elements and then from the model 2 now remember that if I go up there here is I have the model I have fitted model.

(Refer Slide Time: 35:58)



```
2.632741425499998e8
1.9275009e7
4.8107623e7
6.3221498e7
7.4513487e7

In [27]: (X'X)^(-1)*X'y
Out[27]: 6-element Vector{Float64}:
 -3875.4696997071487
  7871.082133925823
 1120.3318525509217
 1510.135408513707
 1439.0771411316493
 1800.9239844534268

In [28]: coef(model2)
Out[28]: 6-element Vector{Float64}:
  3875.469699707117
  7871.082133925831
 1120.331852550947
 1510.1354085137248
 1439.0771411316548
 1800.9239844534393

In [ ]:
```

From the fitted model if I just extract the coefficients it will give me the coefficient values. Now, carefully you check all these values are almost same actually up to 8 or 9 decimal places they are almost same ok.

So, lm when we are calling lm here with formula, price, carat and cut they are essentially what it is doing this particular entire call creating the X matrix creating the Y matrix and then running this ordinary least square estimation and estimating the coefficient values, along with it is also estimating the standard error also estimating the t values p values and the 95 percent confidence interval.

So, now, you can see that how to do you know typical fit a simple regression analysis model using Julia.

Thank you very much let us meet in the next video.