1

00:00:00,180 --> 00:00:01,030

Okay my friends.

2

00:00:01,050 --> 00:00:07,530

Are you ready for the final tool of our data pricing tool kit feature scaling which will allow to put

3

00:00:07,650 --> 00:00:10,120

all our features on the same scale.

4

00:00:10,190 --> 00:00:11,100

So that's the what.

5

00:00:11,110 --> 00:00:14,870

And let me quickly remind the why why do we need to do this.

6

00:00:14,880 --> 00:00:22,110

Well that's because for some of the machinery models that's in order to avoid some features to be dominated

7

00:00:22,200 --> 00:00:28,050

by other features in such a way that the dominated features are not even considered by some machinery

8

00:00:28,050 --> 00:00:28,870

model.

9

00:00:28,890 --> 00:00:34,620

Now you also need to be aware that we won't have to apply feature scaling for all the machinery morals

10

00:00:34,890 --> 00:00:36,210

just for some of them.

11

00:00:36,210 --> 00:00:41,490

Therefore we want to include this in our data pricing template which I will show you by the way at the

12

00:00:41,490 --> 00:00:47,400

end of this material so we will just add this tool in the toolkit because indeed for a lot of missionary

13

00:00:47,400 --> 00:00:53,700

morals we won't even have to apply feature scaling even if we have features taking very different values.

14

00:00:53,700 --> 00:00:59,940

For example if you already know a bit about the multiple in our regression model you know that each

15

00:01:00,000 --> 00:01:03,090

variable is actually multiplied by a coefficient.

16

00:01:03,090 --> 00:01:08,760

You know in the linear regression equation And so well you know if you have variables that take much

17

00:01:08,760 --> 00:01:10,170

higher values than others.

18

00:01:10,200 --> 00:01:16,170

Well when learning the coefficients the coefficients will just compensate by taking small values for

19

00:01:16,200 --> 00:01:18,090

the variables that take high values.

20

00:01:18,090 --> 00:01:18,390

Right.

21

00:01:18,390 --> 00:01:19,940

We will explain this more.

22

00:01:19,940 --> 00:01:21,410

In part two regression.

23

00:01:21,540 --> 00:01:27,420

But for now just know that this is a tool that will be applied from time to time for certain machinery

24

00:01:27,420 --> 00:01:30,800

models but not all the time as you will see in this course.

25

00:01:30,810 --> 00:01:31,710

All right.

26

00:01:31,710 --> 00:01:38,070

And also in the period material I actually told you that I was about to answer one of the most important

27

00:01:38,070 --> 00:01:43,890

questions or most frequently asked questions by the data science community and I will keep of course

28

00:01:43,890 --> 00:01:47,050

my promise I will answer this when it's time.

29

00:01:47,060 --> 00:01:53,400

Meaning at around the middle of this tutorial Minoan is all questions about feature scaling will be

30

00:01:53,400 --> 00:01:57,050

answered so that you have absolutely no confusion.

31

00:01:57,050 --> 00:01:58,920

All right so we have the what and the why.

32

00:01:58,920 --> 00:02:04,280

And now let's proceed to the how many how are we going to apply feature scaling.

33

00:02:04,320 --> 00:02:11,640

And to answer this question I'm gonna show you the following slide which are the main two feature scaling

34

00:02:11,640 --> 00:02:16,780

techniques that indeed put all your features in the same scale.

35

00:02:16,890 --> 00:02:24,840

And these two techniques are first standardization which consists of subtracting each value of your

36

00:02:24,840 --> 00:02:31,380

feature by the mean of all the values of the feature and then dividing by the standard deviation which

37

00:02:31,380 --> 00:02:32,960

is the square root of the variance.

38

00:02:32,970 --> 00:02:39,280

And this will put all the values of the feature between around minus three and plus three right.

39

00:02:39,330 --> 00:02:45,660

All the different features when you apply this transformation on all the features of your data set while

40

00:02:45,690 --> 00:02:49,410

all your features will take value between around minus three and plus three.

41

00:02:49,410 --> 00:02:56,430

So that standardization and then you have normalization which consists of subtracting each value of

42

00:02:56,430 --> 00:03:02,220

your feature by the minimum value of the feature and then dividing by the difference between the maximum

43

00:03:02,220 --> 00:03:05,040

value of the feature and the minimum value of the feature.

44

00:03:05,070 --> 00:03:12,540

And so since this is positive this is positive and this is always larger than this well that means that

45

00:03:12,660 --> 00:03:17,610

all the values of your features will become between 0 and 1.

46

00:03:17,610 --> 00:03:22,830

All right so this will result in having values of features between minus three plus three more or less.

47

00:03:22,830 --> 00:03:27,810

And this will result in having all the values of your features between 0 and 1.

48

00:03:27,810 --> 00:03:34,050

Now the question is also much asked by the data science community should we go for standardization or

49

00:03:34,200 --> 00:03:35,710

normalization.

50

00:03:35,730 --> 00:03:38,850

Well we're gonna be here very pragmatic.

51

00:03:38,850 --> 00:03:45,070

Normalization is recommended when you have a normal distribution in most of your features.

52

00:03:45,090 --> 00:03:47,380

This will be a great feature scaling technique.

53

00:03:47,400 --> 00:03:51,760

In that case standardization actually works well all the time.

54

00:03:51,780 --> 00:03:53,930

It will do the job all the time.

55

00:03:53,970 --> 00:03:59,850

Therefore since this is a technique that will work all the time and this is a technique that is more

56

00:03:59,850 --> 00:04:05,760

recommended for some specific situations where you have most of your features following a normal distribution

57

00:04:06,090 --> 00:04:12,930

then my ultimate recommendation for sure is to go for standardization because indeed this will always

58

00:04:12,930 --> 00:04:13,230

work.

59

00:04:13,230 --> 00:04:18,930

You will always do some relevant feature scaling and this will always improve the training process.

60

00:04:19,080 --> 00:04:21,430

So I am going to teach you this technique.

61

00:04:21,430 --> 00:04:26,610

I am going to teach you on how to apply it on r matrices of features and I'm seeing matrices of features

62

00:04:26,610 --> 00:04:30,900

because now we have two matrices of features which are extra in an excess.

63

00:04:31,050 --> 00:04:36,770

And since we understood in the previous tutorial that feature scaling must be applied after the split.

64

00:04:36,900 --> 00:04:42,390

While you understand that we want apply features getting on the whole matrix of features X but of course

65

00:04:42,480 --> 00:04:50,850

on both extra and an x test separately and actually the scalar will be fitted to only extreme and then

66

00:04:50,850 --> 00:04:56,910

will transform x test you know will apply for just killing on excess because indeed since access is

67

00:04:56,910 --> 00:05:01,850

something that we're not supposed to have during the training but only after like when going in production

68

00:05:02,090 --> 00:05:08,870

well we're not allowed to fit our features going to on the test set right by fitting the features going

69

00:05:08,870 --> 00:05:09,530

to on the test.

70

00:05:09,530 --> 00:05:14,120

That means that we're going to get the mean of the whole set and then the standard deviation in the

71

00:05:14,120 --> 00:05:14,850

feature.

72

00:05:14,870 --> 00:05:19,700

No we don't have the right to do this because X does the supposed to be something new and therefore

73

00:05:19,700 --> 00:05:24,760

we'll just get the mean of the values and extra and then get a standard deviation of the values next

74

00:05:24,760 --> 00:05:30,010

train then apply this formula to transform all the values and rain and then apply that same formula

75

00:05:30,050 --> 00:05:36,440

but with the same mean and send the deviation of the values in exchange to scale the values of Exodus.

76

00:05:36,470 --> 00:05:41,750

It's really really important that you understand this and this is once again to give some further elements

77

00:05:41,750 --> 00:05:47,060

of response to that previous question Should with guilt before or after the split.

78

00:05:47,060 --> 00:05:47,450

All right.

79

00:05:47,840 --> 00:05:54,890

So now that we are all clear on this let's proceed to the implementation of the how meaning the implementation

80

00:05:55,040 --> 00:06:02,830

of feature scaling All right so as usual you know in order to be as much efficient as we can.

81

00:06:02,840 --> 00:06:05,270

We're going to do this with psychic learn right.

82

00:06:05,300 --> 00:06:11,360

This data science library that has all the tools and the tool that we're about to use is a class called

83

00:06:11,480 --> 00:06:18,470

standard scalar and which will exactly perform standardization on both your matrix of features of the

84

00:06:18,470 --> 00:06:21,810

training set and the matrix of features of the test.

85

00:06:21,830 --> 00:06:22,630

So let's do this.

86

00:06:22,730 --> 00:06:30,410

Let's start by importing this class which we have to take first from Well cyclone of course as K learn

87

00:06:30,740 --> 00:06:35,780

and then from which we're going to get access to the pre processing module.

88

00:06:35,780 --> 00:06:39,970

Perfect which is a module that contains that standard scalar class.

89

00:06:39,980 --> 00:06:40,710

All right.

90

00:06:40,780 --> 00:06:46,690

So then we're ready to import what we want which is the standard scalar class.

91

00:06:46,700 --> 00:06:47,630

Perfect.

92

00:06:47,630 --> 00:06:53,480

So now we have the class and then the natural next step here is of course to create an object of the

93

00:06:53,480 --> 00:06:55,560

class I'm about to reveal soon.

94

00:06:55,760 --> 00:06:59,710

That answer to one of the most we can questions in the data science community.

95

00:06:59,900 --> 00:07:01,400

So let's create this object.

96

00:07:01,400 --> 00:07:08,060

We're going to call it as C for standard scalar and then well this object will be created as an instance

97

00:07:08,300 --> 00:07:09,670

of the standard scalar class.

98

00:07:09,680 --> 00:07:15,950

I'm taking it here basing that right here adding some parentheses and good news here we don't have any

99

00:07:16,040 --> 00:07:22,610

arguments to input because what we simply want to do is get that mean get that standard deviation and

100

00:07:22,610 --> 00:07:25,600

then apply this formula to all the values in the feature.

101

00:07:25,640 --> 00:07:27,780

And for this we don't need actually any parameters.

102

00:07:27,890 --> 00:07:29,870

This will automatically do the job.

103

00:07:30,500 --> 00:07:33,110

All right so then next step.

104

00:07:33,110 --> 00:07:33,800

And now.

105

00:07:33,830 --> 00:07:40,520

Well now is the time for me to reveal the answer to that question which is one of the most frequently

106

00:07:40,520 --> 00:07:43,570

asked questions in the data science community.

107

00:07:44,210 --> 00:07:54,290

And that question is do we have to apply feature scaling you know standardization to the dummy variables

108

00:07:54,530 --> 00:07:56,460

in the matrix of features.

109

00:07:56,480 --> 00:08:00,720

This is one of the most frequently asked questions you will find it everywhere online as well.

110

00:08:00,740 --> 00:08:06,620

And once again actually the answer is pretty obvious but only after you get the explanation.

111

00:08:06,620 --> 00:08:07,950

So let me tell you the answer.

112

00:08:08,060 --> 00:08:10,220

The answer is No.

113

00:08:10,220 --> 00:08:12,630

The answer is no because simply.

114

00:08:12,770 --> 00:08:19,040

Well remember the goal of standardization or feature scaling in general it is to have all the values

115

00:08:19,040 --> 00:08:21,560

of the features in the same range.

116

00:08:21,560 --> 00:08:28,340

And since I told you that standardization actually transforms your features so that they take values

117

00:08:28,340 --> 00:08:30,840

between more or less minus three and plus three.

118

00:08:30,980 --> 00:08:37,240

Well since here are dummy variables already take values between minus three and plus three because they're

119

00:08:37,250 --> 00:08:39,280

equal to either 1 or 0.

120

00:08:39,290 --> 00:08:46,340

Well there is nothing extra to be done here with standardization and actually standardization will only

121

00:08:46,340 --> 00:08:52,070

make it worse because indeed it will still transform these values between minus three plus three.

122

00:08:52,070 --> 00:08:57,410

But then you will totally lose the interpretation of these variables in other words you will lose the

123

00:08:57,410 --> 00:09:01,520

information of which country corresponds to the observation.

124

00:09:01,520 --> 00:09:06,980

Now we perfectly know that you know remember one zero and zero corresponds to France because that's

125

00:09:06,980 --> 00:09:10,580

how it was encoded and then 0 0 and 1 corresponds to Spain.

126

00:09:10,790 --> 00:09:17,000

But you know after we apply features killing if we apply it on the dummy variables we will get nonsense

127

00:09:17,000 --> 00:09:23,150

numerical values and we will be absolutely incapable to say which couple of three values here correspond

128

00:09:23,150 --> 00:09:24,490

to which country.

129

00:09:24,530 --> 00:09:26,570

So we will totally lose the interpretation.

130

00:09:26,600 --> 00:09:32,350

And besides this won't improve at all to trainee performance because indeed are dummy variables are

131

00:09:32,400 --> 00:09:37,170

any way already between the same skill range as your other variables.

132

00:09:37,310 --> 00:09:43,160

You will see online that applying standardisation to your dummy variables might still increase slightly

133

00:09:43,160 --> 00:09:45,860

the performance you know the final accuracy of your model.

134

00:09:45,920 --> 00:09:51,560

But I've experimented many times and I've never seen you know a considerable difference that would justify

135

00:09:51,560 --> 00:09:54,070

here to apply features getting on the dummy variables.

136

00:09:54,170 --> 00:09:59,270

So really don't do this only apply feature scaling to your numerical values right.

137

00:09:59,290 --> 00:10:04,240

Here we have clearly some variables taking values in a very different range.

138

00:10:04,310 --> 00:10:04,550

Right.

139

00:10:04,540 --> 00:10:09,980

The age goes between 0 and 100 and the salary goes between 0 and one hundred thousand.

140

00:10:09,980 --> 00:10:14,630

So clearly here if it's better for the machinery model we have to apply for scaling.

141

00:10:14,690 --> 00:10:21,400

But let's leave these dummy variables alone so that we can keep the interpreter ability of the model.

142

00:10:21,390 --> 00:10:22,010

All right.

143

00:10:22,100 --> 00:10:24,430

So that was the other very important questions.

144

00:10:24,440 --> 00:10:25,960

Now I've covered everything.

145

00:10:26,030 --> 00:10:30,280

There should not be any confusion left in data processing.

146

00:10:30,290 --> 00:10:31,770

I'm really glad that you know this.

147

00:10:31,820 --> 00:10:37,820

And therefore I encourage you now to press pause on this video and guess what will be the next step

148

00:10:37,820 --> 00:10:43,550

to apply feature scaling to our matrices of feature extreme and access.

149

00:10:43,550 --> 00:10:44,140

All right.

150

00:10:44,240 --> 00:10:44,540

Good.

151

00:10:44,540 --> 00:10:46,090

Now I'm going to give you a solution.

152

00:10:46,160 --> 00:10:53,550

So first of all we're gonna fit our scalar you know our standardization tool on the training set.

153

00:10:53,540 --> 00:10:58,900

So I'm taking the training set first X train all right.

154

00:10:58,900 --> 00:11:04,560

And then since I've just explained that we won't apply feature scaling on the dummy variables.

155

00:11:04,610 --> 00:11:11,810

Well then that means that we will fit our standard scalar object only on these two columns here containing

156

00:11:11,810 --> 00:11:13,480

the ages and the salaries.

157

00:11:13,490 --> 00:11:19,010

And therefore here I'm going to take only the two columns here for the age and the salary and then of

158

00:11:19,010 --> 00:11:24,410

course all the rows and remember the trick to take all the rows we just it's add a colon which means

159

00:11:24,410 --> 00:11:29,370

we're taking the range from the lower bound to the upper bound meaning everything and then two data

160

00:11:29,390 --> 00:11:29,960

columns.

161

00:11:30,050 --> 00:11:31,910

Well be careful.

162

00:11:31,910 --> 00:11:37,670

This is what we have to look at not this because indeed we want to take this column and this one.

163

00:11:37,730 --> 00:11:40,550

And so now the question is what are the indexes of these columns.

164

00:11:40,880 --> 00:11:44,120

Well remember that index is in both and start from zero.

165

00:11:44,120 --> 00:11:45,930

So this hasn't 0.

166

00:11:45,980 --> 00:11:48,650

Then the second column has an X1 index too.

167

00:11:48,770 --> 00:11:54,310

And this one has index with each column has index 3 and the salary column has index 4.

168

00:11:54,800 --> 00:11:59,480

But since I want to make this template as much generic as we can.

169

00:11:59,480 --> 00:12:05,420

And since when you one hot encode your categorical variables they always automatically go as the fourth

170

00:12:05,420 --> 00:12:06,110

column.

171

00:12:06,110 --> 00:12:08,020

Well we're going to do something even better.

172

00:12:08,090 --> 00:12:14,720

We're going to specify the index as we want here by 3 which is the index of the H column and then a

173

00:12:14,720 --> 00:12:16,400

simple column.

174

00:12:16,400 --> 00:12:16,640

Right.

175

00:12:16,640 --> 00:12:22,850

Because this will take the range from the column of next three which is the age up to all the other

176

00:12:22,850 --> 00:12:23,500

columns.

177

00:12:23,550 --> 00:12:28,520

You know there is not a minus one here so this will take all the remaining columns from the age meaning

178

00:12:28,520 --> 00:12:29,840

the age and the salary.

179

00:12:29,840 --> 00:12:31,740

Basically this will take these two columns.

180

00:12:31,880 --> 00:12:36,770

And if you have a larger matrix of features with numerical values in your future as well this will just

181

00:12:36,770 --> 00:12:38,150

take all the columns.

182

00:12:38,150 --> 00:12:41,370

All right so that's a little trick more elegant let's say.

183

00:12:41,420 --> 00:12:49,420

And so now we are of course going to use our object which we called as C from which well we're going

184

00:12:49,420 --> 00:12:57,350

to use that fit method that will indeed for each feature of extra gain compute the mean of the future

185

00:12:57,470 --> 00:13:02,900

meaning the mean of the age and then the mean of the salary and then compute the standard deviation

186

00:13:02,900 --> 00:13:05,180

of the future the age and the salary.

187

00:13:05,180 --> 00:13:07,140

And that's exactly what the fifth method will do.

188

00:13:07,160 --> 00:13:12,320

It will only compute the mean and the standard deviation of all the values.

189

00:13:12,320 --> 00:13:18,620

And then you have the transform method that will indeed apply this formula by you know transforming

190

00:13:19,010 --> 00:13:25,760

each of the values here if each feature into this value resulting from this formula.

191

00:13:25,760 --> 00:13:30,270

All right so it's important to understand the difference between fit and transform fit.

192

00:13:30,290 --> 00:13:35,450

We'll just get the mean and standard deviation of each of your features and transform will apply this

193

00:13:35,450 --> 00:13:40,680

formula to indeed transform your values so that they can all be in the same scale.

194

00:13:40,700 --> 00:13:41,030

All right.

195

00:13:41,420 --> 00:13:48,290

And now the good news is that one of the methods of the standard scalar class is actually fit transform

196

00:13:48,590 --> 00:13:54,860

which of course will proceed to the two tools at the same time meaning it will fit your matrix of features

197

00:13:54,860 --> 00:14:00,320

to get the mean and standard deviation and then right after that transform all the values of the features

198

00:14:00,530 --> 00:14:02,770

to turn them into this formula.

199

00:14:02,780 --> 00:14:03,140

All right.

200

00:14:03,380 --> 00:14:10,610

So let's call this method right away to make it efficient you know fit on the score trend forum then

201

00:14:10,610 --> 00:14:15,700

some parenthesis and now obviously you know what to input inside this fit transfer method.

202

00:14:16,040 --> 00:14:24,050

Well that's of course exactly the same as extra in here because indeed we will only apply feature scaling

203

00:14:24,140 --> 00:14:29,290

to our numerical columns here containing non integer values.

204

00:14:29,300 --> 00:14:29,560

Right.

205

00:14:29,570 --> 00:14:32,090

Non dummy variables values.

206

00:14:32,090 --> 00:14:32,540

All right.

207

00:14:32,540 --> 00:14:36,250

Great so this will apply for just killing two or two comes into training set.

208

00:14:36,380 --> 00:14:42,320

And now I hope you know what the next step is going to be and you won't fall into the trap.

209

00:14:42,350 --> 00:14:50,780

Now we have to also transform our matrix of features of a tacit meaning x test this matrix of features.

210

00:14:50,930 --> 00:14:57,310

But since this data is like new data which we get you know later on in production.

211

00:14:57,610 --> 00:15:03,590

Well for this data we will only apply the transform method because indeed the features of the test set

212

00:15:03,740 --> 00:15:09,500

need to be scaled by the same scalar that was used on the training set.

213

00:15:09,560 --> 00:15:15,050

We can not get a new scalar you know if we apply the fit transfer method here on excess we would get

214

00:15:15,110 --> 00:15:22,220

a new scalar and that would absolutely not make sense because excess will actually be the input of the

215

00:15:22,370 --> 00:15:24,900

predict function that will return to predictions.

216

00:15:24,890 --> 00:15:30,110

You know after the missionary model is trained and since this machine learning model will be trained

217

00:15:30,200 --> 00:15:36,530

with a particular scalar you know the scalar applied on the training set while in order to make predictions

218

00:15:36,740 --> 00:15:39,730

that will be congruent with the way the model was trained.

219

00:15:39,920 --> 00:15:45,800

Well we need to apply the same skill that was used on the training set onto the test set so that we

220

00:15:45,800 --> 00:15:52,370

can get indeed the same transformation and therefore Indian some relevant predictions with the predict

221

00:15:52,370 --> 00:15:54,470

method applied to X test.

222

00:15:54,470 --> 00:15:59,660

So here it's clearly the transfer method that must only be applied and therefore what we're going to

223

00:15:59,660 --> 00:16:01,090

do to make it efficient.

224

00:16:01,100 --> 00:16:05,570

Well we're going to copy this line of code and just below we're going to paste it.

225

00:16:05,570 --> 00:16:09,020

We're going to replace of course extra and by excess.

226

00:16:09,140 --> 00:16:16,970

And then here as well x trained by X test and then just call of course the transform method from that

227

00:16:17,090 --> 00:16:23,780

same scalar that was applied on the training set because indeed this is part of the training right.

228

00:16:23,800 --> 00:16:30,170

Even if we haven't started the training well this operation that we apply here on our training set is

229

00:16:30,170 --> 00:16:32,590

you know the preparation of the training.

230

00:16:32,600 --> 00:16:32,890

All right.

231

00:16:32,900 --> 00:16:35,660

So I hope it's clear it's very important that you understand this.

232

00:16:35,870 --> 00:16:36,500

And now.

233

00:16:36,680 --> 00:16:42,530

Well I have to say congratulations because we're actually done implementing our final tool.

234

00:16:42,530 --> 00:16:48,740

And of course I'm going to show you the result of feature scaling here so let me create two more code

235

00:16:48,740 --> 00:16:58,460

cells inside which we're going to print first X train and then let me copy this and then we're going

236

00:16:58,460 --> 00:17:00,840

to print x test.

237

00:17:00,860 --> 00:17:01,530

All right.

238

00:17:01,550 --> 00:17:01,940

Perfect.

239

00:17:01,940 --> 00:17:05,550

So let's first run this to apply for scaling.

240

00:17:05,600 --> 00:17:05,960

Perfect.

241

00:17:05,960 --> 00:17:06,470

There we go.

242

00:17:06,470 --> 00:17:07,840

No execution error.

243

00:17:07,970 --> 00:17:15,190

Then let's print x train and of course we get while still the same values for the dummy variables which

244

00:17:15,190 --> 00:17:18,150

are indeed still between minus three and plus three.

245

00:17:18,220 --> 00:17:26,290

But then our age and salary variables were transformed so that they take new values between minus two

246

00:17:26,350 --> 00:17:27,340

and plus two.

247

00:17:27,400 --> 00:17:29,860

Sometimes you will see values between minus three and plus three.

248

00:17:29,860 --> 00:17:31,370

Here it's minus two and plus two.

249

00:17:31,390 --> 00:17:35,880

But anyway now all are variables are on the same scale.

250

00:17:35,890 --> 00:17:41,960

And this will be perfect to improve or optimize the training of certain machinery models.

251

00:17:41,980 --> 00:17:47,710

And of course you will see exactly which ones there are going to be the further we progressed in this

252

00:17:47,710 --> 00:17:49,090

machine during course.

253

00:17:49,090 --> 00:17:53,370

So now you know everything that is also execute this sale to print excess.

254

00:17:53,380 --> 00:17:59,290

And once again well you still have your dummy variables here for the same two customers that were here

255

00:17:59,500 --> 00:18:04,750

but then the age and the salary were scaled so that they take once again values between minus two plus

256

00:18:04,750 --> 00:18:05,320

two.

257

00:18:05,410 --> 00:18:06,820

All right.

258

00:18:06,820 --> 00:18:07,320

Okay.

259

00:18:07,360 --> 00:18:08,240

So great.

260

00:18:08,250 --> 00:18:14,170

I'm really happy that we're now done with this data depressing tool kit because that means only one

261

00:18:14,170 --> 00:18:14,740

thing.

262

00:18:14,740 --> 00:18:20,860

That means that we are ready to start the exciting steps of the journey which is to build machine learning

263

00:18:20,860 --> 00:18:23,770

models that will perform amazing predictions.

264

00:18:23,770 --> 00:18:29,350

And we're going to start with the regression models which will predict some continuous numerical values

265

00:18:29,740 --> 00:18:32,640

and we will learn how to do that on different data sets.

266

00:18:32,710 --> 00:18:39,490

But before we move on to this next part I just want to show you the data processing template which will

267

00:18:39,490 --> 00:18:43,170

be so useful for us to tackle in the flashlight.

268

00:18:43,240 --> 00:18:49,870

The data pricing phase for each of our future machinery models because indeed you will see that this

269

00:18:49,870 --> 00:18:56,510

template was made so that we will only have each time one or two things to change and most of the time.

270

00:18:56,530 --> 00:19:04,060

One thing to change because indeed in this template I included the three always used tools that we will

271

00:19:04,060 --> 00:19:07,540

use for our machinery models which are important the libraries right.

272

00:19:07,540 --> 00:19:11,550

We will always need these libraries then importing the data set in here.

273

00:19:11,560 --> 00:19:16,780

Appreciate that we will only have one thing to change which will be the name of the data set because

274

00:19:16,780 --> 00:19:22,840

indeed this line of code will automatically take all the columns except the last one meaning all your

275

00:19:22,840 --> 00:19:23,570

features.

276

00:19:23,710 --> 00:19:27,670

And this line of code will take automatically the dependent variable.

277

00:19:27,760 --> 00:19:33,670

So here you will only have the name of the dataset change and then of course include this tool because

278

00:19:33,910 --> 00:19:39,580

for most of our missionary models we will have to split the dataset into these two separate sets one

279

00:19:39,580 --> 00:19:44,620

training set to train our missionary model and one to set to evaluate its performance.

280

00:19:44,620 --> 00:19:48,880

And here once again we will actually have nothing to change.

281

00:19:48,880 --> 00:19:55,030

So in the whole template we will only have one thing to change which will be the name of the dataset.

282

00:19:55,040 --> 00:20:01,580

And that's why this data pricing template will be so useful for us because we will each time take all

283

00:20:01,600 --> 00:20:04,250

the data repricing phase in a flashlight.

284

00:20:04,390 --> 00:20:07,150

So make sure to have this template ready.

285

00:20:07,240 --> 00:20:11,890

Each time we are going to build our future machinery models and now take a good break.

286

00:20:11,890 --> 00:20:17,620

You really deserve it after this data repricing phase and this answers to the questions that reduce

287

00:20:17,740 --> 00:20:23,710

any confusion to digest it well and as soon as you're ready to tackle the first branch of machinery

288

00:20:23,710 --> 00:20:28,660

moral which is regression well let us continue our journey together in this next part.

289

00:20:28,730 --> 00:20:30,640

And until then enjoy machine learning.