Week 2

Week 2

Interpretation of a learning curve

Jean roman Week 2 · 6 years ago · Edited

Hello,

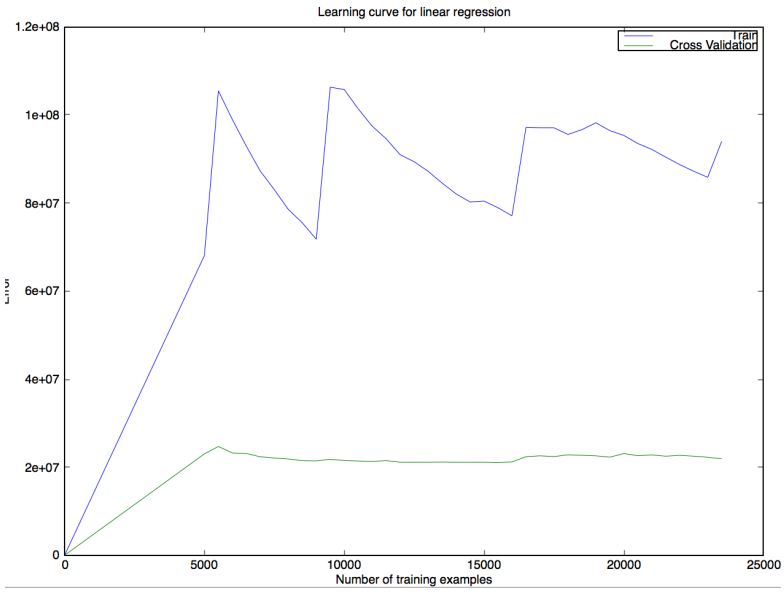
This question may be out of the bonds of this forum, I apologize if it is the case (I also posted id on Stack Overflow)

While following this class, I wanted to test what I learned on another dataset and plot the learning curve for different algorithms.

I (quite randomly) chose the Online News Popularity Data Set, and tried to apply a linear regression to it.

Note : I'm aware it's probably a bad choice but I wanted to start with linear reg to see later how other models would fit better.

I trained a linear regression and plotted the following learning curve :



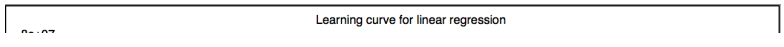
This result is particularly surprising for me, so I have questions about it :

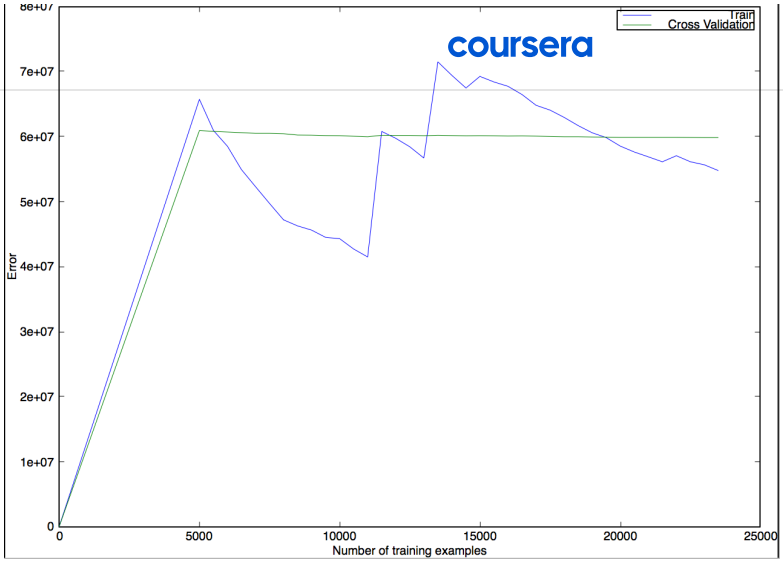
- Is this curve even remotely possible or is my code necessarily flawed?
- If it is correct, how can the training error grow so quickly when adding new training examples? How can the cross validation error be lower than the train error?
- If it is not, any hint to where I made a mistake?

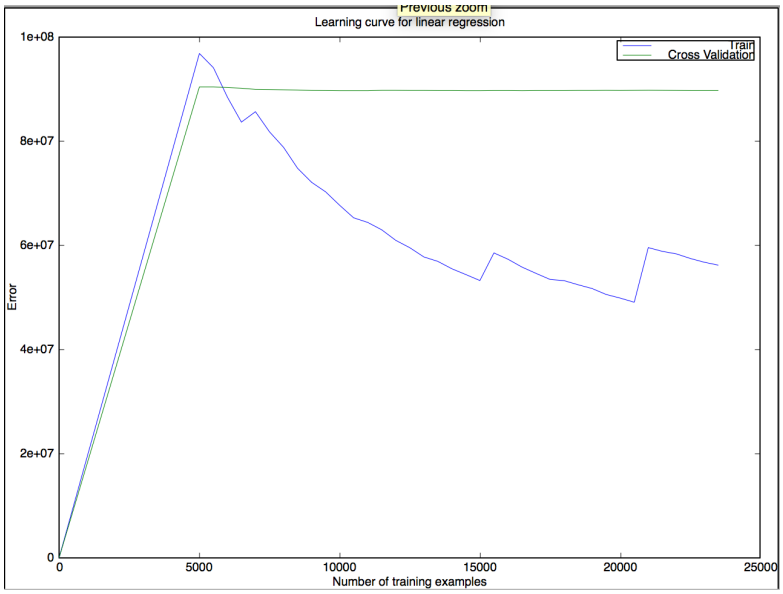
Most of my code is coming directly from answers to ex1 and ex5, so I won't post it here.

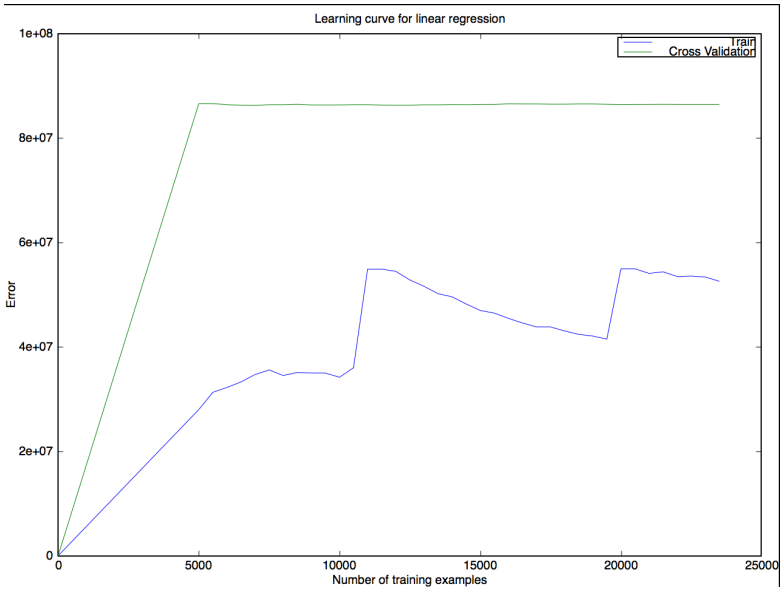
Thanks in advance !

Edit : if I shuffle the whole dataset before splitting train/validation and doing the learning curve, I have very different results, like the 3 following :









Note : the training set size is always around 24k examples, and validation set around 8k examples.

1 Upvote Reply Follow this discussion

coursera

Earliest Top Most Recent

Tom Mosher Mentor · 6 years ago · Edited

Thanks for posting this, it is good material to study.

The most critical question before considering your others:

- Did you randomly select the members of the training and validation sets, or are they sorted in some way that includes new features or values at the points where there are abrupt changes in the cost curve?

0 Upvotes Hide 12 Replies

Jean roman · 6 years ago

I did not. I just updated my question with 3 new learning curves from the same dataset, shuffled 3 times.

0 Upvotes

Tom Mosher Mentor · 6 years ago · Edited

What method are you using to randomly select the different sets? It's OK to post that here, since this isn't a graded programming exercise.

One method works like this:

```
m = size(X, 1)
sel = randperm(m);
```

This starts by creating a vector of random values between 1 and the number of rows in X.

If we want a 60/40 split between the training set and the validation set:

```
sel_train = sel(1:floor(m * 0.6))
sel_val = sel(floor(m*0.6) + 1:end)
X_train = X(sel_train,:)
y_train = y(sel_train);
X_val = X(sel_val,:)
y_val = y(sel_val)
```

We split the sel vector into two parts - the first 60% for training, and the remainder for validation.

Then then use vector indexing to copy the rows of X and y into the training set and the validation set.

0 Upvotes

Jean roman · 6 years ago

I used the following :

```
% Prepare data
```

```
load('rawData.mat');

% randomize
data = data(randperm(size(data,1)),:);

% Number of elements for training, validation, and test sets
mTotal = size(data,1);
pctTrain = 0.6;
pctVal = 0.2;
mTrain = ceil(mTotal*pctTrain);
mVal = ceil(mTotal*pctVal);

% Define sets
X_train = data(1:mTrain,1:end - 1);
y_train = data(1:mTrain,end); % Last column is the output value
X_val = data(mTrain + 1:mTrain + mVal,1:end - 1);
y_val = data(mTrain + 1:mTrain + mVal,end);
X_test = data(mTrain + mVal + 1:end,1:end - 1);
y_test = data(mTrain + mVal + 1:end,end);
```

Is that alright ?

0 Upvotes

Tom Mosher Mentor · 6 years ago

I tested your method a little bit on a small and regular data set, it seems OK.

The next questions to address are how is the raw data set organized, and is your random method suitable for it?

Another question is, what does your learning curve tell you about how the model works for this data set? Look at bias, variance, etc.

1 Upvote

Jean roman · 6 years ago

First of all : thanks for engaging in this discussion with me, it is very helpful.

how is the raw data set organized, and is your random method suitable for it?

The dataset represents a set of mashable articles : each row is an article, the features are different attributes of the article, and the y value is the number of social network "shares". Looking at it, I see that it is organized by date : the first row is the earliest article, and the last is the latest.

Knowing that (i) the date of the post (or more exactly its distance in days to the last post of the dataset) is a feature of the dataset, and (ii) that the trend of the number of shares is stable over time (see chart at the end), I would say that the random method is OK.

Am I correct in this assumption ?

what does your learning curve tell you about how the model works for this data set?

In a nutshell, we can conclude that it is a pretty bad fit.

The error is always stable at a very high level (curves start at 5000 examples, what we see between 0 and 5k is just me not knowing how to change the X axis), so it is very high biais.

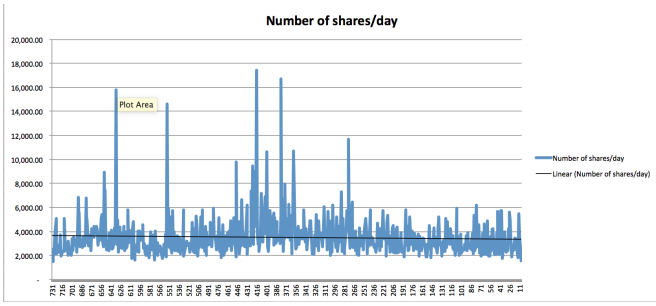
As for the variance, I don't know what to conclude as the training error has different trends according to the way it is shuffled.

My last question for this thread is the following : how do you recommend I spend my time now that we have this ? Should I keep working on the linear regression until the learning curve makes sense ? I know I will never have anything close to a good fit but maybe the learning curve shows that I have done something wrong that I need to fix.

Or should I leave it and try another model ? If so, I am guessing that the

polynomial makes no senses as I have 58 features. So it leaves me with a neural network. If neural networks is the right choice, I will post another question on how to adapt ex4 to a value predicting model.

Appendix : trend of the number of shares over time



I think this flat trend proves that the date is not an important feature in the model, so the shuffle should not be a problem.

0 Upvotes

Tom Mosher Mentor · 6 years ago

Time can be a difficult parameter for a learning system to use well. What is the format being used, and the range of values?

Before you abandon linear regression, have you looked at whether the features would benefit from normalization?

A NN seems like a good next step, due to their ability to form complex relationships between the features.

1 Upvote

Jean roman · 6 years ago

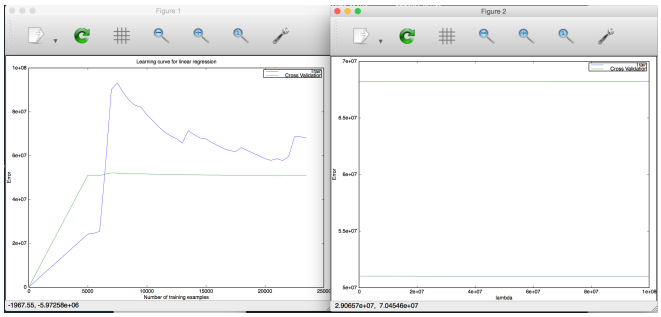
Time is included as the number of days prior to the dataset acquisition. The range goes from 8 (latest articles) to 731 (oldest articles).

Furthermore, we have 7 features reflecting the day of the week the article was published :

```
Mashable
31. weekday_is_monday:          Was the article published on a Monday?
32. weekday_is_tuesday:         Was the article published on a Tuesday?
33. weekday_is_wednesday:       Was the article published on a Wednesday?
34. weekday_is_thursday:        Was the article published on a Thursday?
35. weekday_is_friday:          Was the article published on a Friday?
36. weekday_is_saturday:        Was the article published on a Saturday?
37. weekday_is_sunday:          Was the article published on a Sunday?
```

I understand that time can be misleading in such models, but is it safe to assume that in our case, shuffling the values like I did has no impact ?

I have tried a validation curve using the code from ex5, and I get the following :



(on the left the learning curve on the same shuffled data as the validation curve)

So the lambda has basically no impact on the errors. They change a little, but because of the scale we cannot see the changes on the chart:

```
lambda    Train Error    Validation Error
```

0.000000	67988063.301771	50958988.308065
0.001000	67988063.342778	50958987.942168
0.003000	67988063.322438	50958987.341032
0.010000	67988063.309913	50958985.164196
0.030000	67988063.320017	50958978.891106
0.100000	67988063.337211	50958956.967514
0.300000	67988063.401076	50958894.448638
1.000000	67988063.905754	50958676.871708
3.000000	67988068.679159	50958064.594206
10.000000	67988120.112301	50956027.683669
30.000000	67988028.231424	50953974.159911
100.000000	67986597.852496	50932117.844696
300.000000	67983941.674356	50921619.849255
1000.000000	68021756.888694	50925611.015221
3000.000000	68061732.718100	50945119.017075
10000.000000	68103881.431537	50975246.208069
30000.000000	68132178.929157	50998444.257439
100000.000000	68151150.310427	51013734.756245
300000.000000	68161877.789205	51019904.757760
1000000.000000	68172351.219211	51021900.417415
3000000.000000	68185437.523343	51024633.916008
10000000.000000	68201735.490266	51027418.089103
30000000.000000	68211881.432115	51019552.832835
100000000.000000	68218712.027418	51000108.429408

So now I guess we left no stone unturned and we can move on to NN, right ?

0 Upvotes

Tom Mosher Mentor · 6 years ago

That seems like a safe representation of time for this dataset. Seems like the day of week fields are a logical valued, that's good. Randomly selecting the examples should not cause a problem.

Your lambda values are huge. The fit of this model is so bad that even heavy regularization doesn't hurt much.

It is often the case that regularization doesn't have much impact if there are lots of features and training examples. Systems like this can be self-averaging, so regularization doesn't do much.

I wouldn't give up on this yet. I think there is more to be learned before you go to an NN.

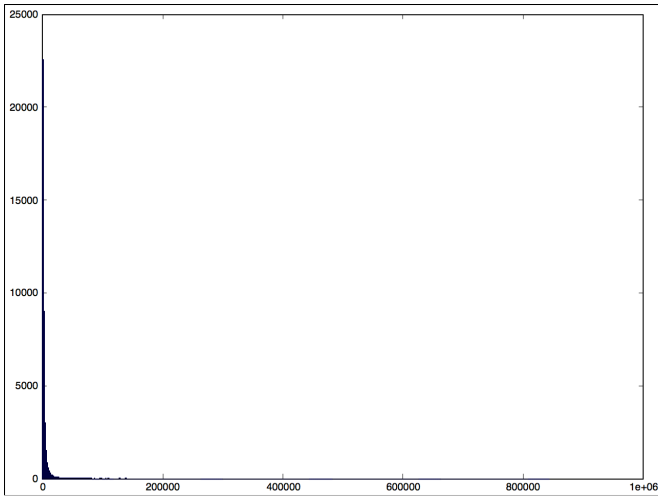
- You didn't answer whether the features are already normalized.
- It might be worth adding 2nd order polynomial terms, and see if it helps much. You can do it with a simple call to your polyFeatures() function from ex5.

1 Upvote

Jean roman · 6 years ago

OK I think I got it !

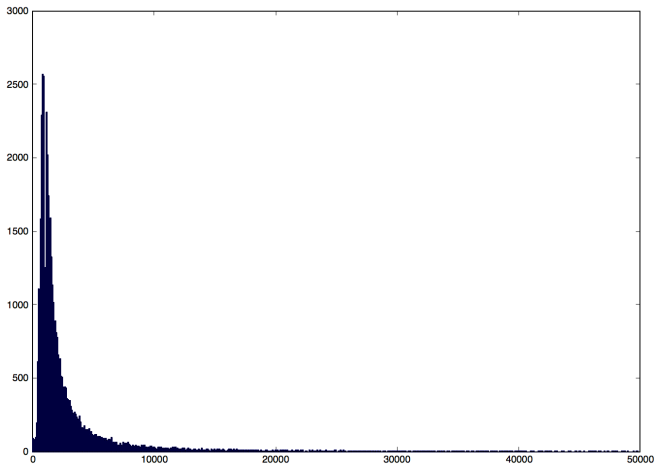
Here is the distribution of the y values of the dataset :



It seems like most of the values are below 50k and we have a few outliers

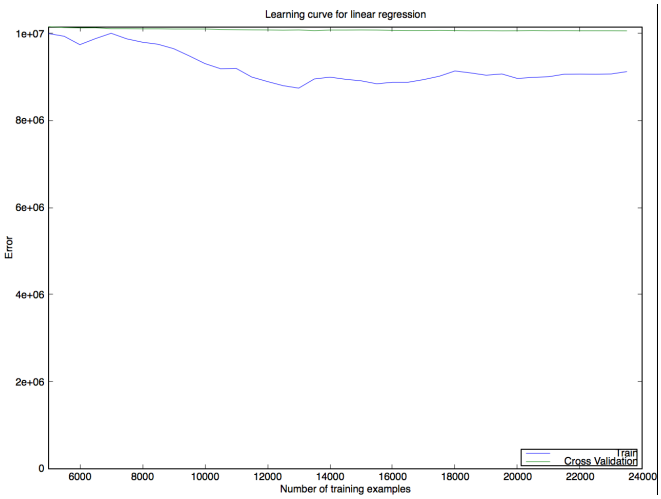
above, and a crazy one above 800k.

I counted 203 y values above 50k (ie 0.5% of the dataset), so I removed them and got the following distribution :



It looks much better !

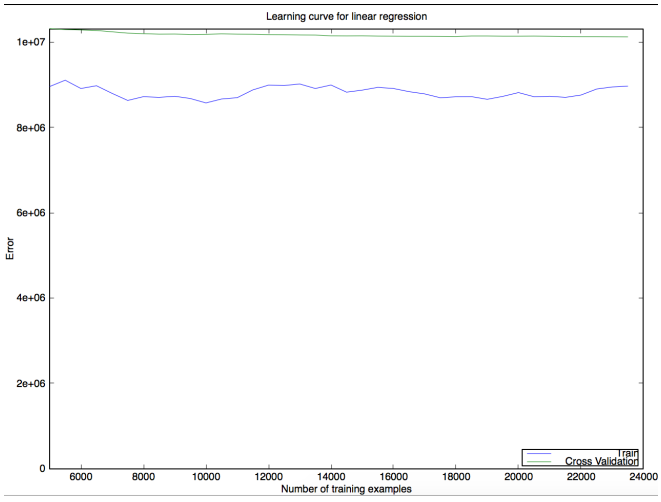
And now, when I train the linear regression, I get results that make much more sense :



(I also made some progress on plot scales)

Do you agree it was a good move to remove as many outliers ?

So now, if I try to answer your question, adding normalization and polynomials gives very similar results (the error is even slightly higher):



To be fair : the polynomial function that we use in ex5 will only take the

squares of every feature, and not add any combination of different features. For example, if we have two features, I think using polyfeatures can only add x1^2, x2^2, but not x1*x2.

But in any case I think it's safe to say that we have explored all the possibilities of the linear regression, would you agree ?

0 Upvotes

Tom Mosher Mentor · 6 years ago

The polyFeatures() function from ex5 only works on one individual feature - not a complex set with many features.

You could use a method like the mapFeature() function from ex2 (which creates quadratic features), but you would need to modify it so it works for more than 2 features. Perhaps use a lower degree.

Or maybe use PCA to find the features that are the principal components, and only use mapFeature() on the highest two.

How to handle anomalies is covered later in the course. Whether you can discard the outliers depends heavily on whether you need to predict them accurately for the intended goals of your system.

Yes, I think you've explored linear regression sufficiently.

Just curious, what validation prediction accuracy percentage are you getting with your best solution?

1 Upvote

Jean roman · 6 years ago · Edited

Your last question is surprisingly hard for me.

I don't think that we covered prediction accuracy percentage for a linear regression in the course. So, after some googling, I chose to answer with the Mean absolute percentage error.

For the regular linear regression, I have something around 130%.

Is that the metric I should be using to measure prediction accuracy for regressions ?

Also, the underlying question beside that is : what metric do I use to compare two different models ? Each one having different cost functions, I'm guessing we cannot use them as references.

0 Upvotes

Tom Mosher Mentor · 6 years ago · Edited

Yes, my last question was ill-considered. Sorry about that. For linear regression, what Prof Ng does is use the unregularized cost value. That does a fair job with comparing different models as well.

0 Upvotes

PK Reply

Reply

PK Reply

Reply