Exercise 15 - Overfitting

Zoltan Kekecs

8 May 2020

Table of Contents

Abstract

This exercise will show some model selection strategies which are discouraged because they lead to unreliable predictions on new data due to overfitting.

Data management and descriptive statistics

Load data about housing prices in King County, USA

In this exercise we will predict the price of apartments and houses.

We use a dataset from Kaggle containing data about housing prices and variables that may be used to predict housing prices. The data is about accomodations in King County, USA (Seattle and sorrounding area).

We only use a portion of the full dataset now containing information about N = 200 accomodations.

The data can be downloaded at:

https://github.com/kekecsz/SIMM32/blob/master/2020/Lab_2/House%20price%20King%20County.sav

Check the dataset

You should always check the dataset for coding errors or data that does not make sense, by eyeballing the data through the data view tool, checking descriptive statistics and through data visualization.

Overfitting

First rule of model selection:

Always go with the model that is grounded in theory and prior research, because automatic model selection can lead to bad predictions on new datasets due to overfitting!

"Predicting" variability of the outcome in your original data is easy If you fit a model that is too flexible, you will get perfect fit on your intitial data.

For example you can fit a line that would cover your data perfectly, reaching 100% model fit… to a dataset where you already knew the outcome.

However, when you try to apply the same model to new data, it will produce bad model fit. In most cases, worse, than a simple regression.

In this context, data on which the model was built is called the training set, and the new data where we test the true prediction efficiency of a model is called the test set. The test set can be truely newly collected data, or it can be a set aside portion of our old data which was not used to fit the model.

Linear regression is very inflexible, so it is less prone to overfitting. This is one of its advantages compared to more flexible prediction approaches.

Comparing model performance on the training set and the test set

In this exercise we will demonstrate that the more predictors you have, the higher your R^2 will be, even if the predictors have nothing to do with your outcome variable.

First, we will generate some random variables for demonstration purposes. These will be used as predictors in some of our models in this exercise. It is important to realize that these variables are randomly generated, and have no true relationship to the sales price of the apartments. Using these random numbers we can demonstrate well how people can be mislead by good prediction performance of models containing many predictors. These random variables can be created in the **Transform > Compute variables** tab, by selecting Random Numbers from the "Functions Group" list, and "Rv.normal" from the list of functions in the "functions and special variables" list. And specify that you want a random variable with a mean of 0 and a standard deviation of 1. Or just enter this into the compute

field: RV.NORMAL(0,1). This should be done 20 times. The easiest way to create these variables is probably by running the syntax below:

```
COMPUTE RND1=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND2=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND3=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND4=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND5=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND6=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND7=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND8=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND9=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND10=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND11=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND12=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND13=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND14=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND15=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND16=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND17=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND18=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND19=RV.NORMAL(0,1).
EXECUTE.
COMPUTE RND20=RV.NORMAL(0,1).
EXECUTE.
```

We create a new data file from the first half of the data (N = 100). We will use this to fit our models on. This is our training set. We set aside the other half of the dataset so that we will be able to test prediction performance on it later. This is called the test set.

You can do that by creating a variable that indicates whether a case is in the first or the second 100 cases, and after that using the **Data > Split** into files tab. Or you can just manually cut and paste one half of the dataset into a new file.

Now we will perform a hierarchical regression where first we fit our usual model predicting price with sqft_living and grade on the training set. Next, we fit a model containing sqft_living and grade and the 20 randomly generated variables that we just created.

(the names of the random variables are RND1, RND2, RND3, …)

Now we can compare the model performance. First, if we look at the normal R^2 indexes of the models or the RSS, we will find that the model using the random variables was much better at predicting the training data. The error was smaller in this model, and the overall variance explained is bigger. There might even be some random predictors that are identified as having significant added prediction value in this model, even though they are not supposed to be related to price at all, since we just created them randomly. This is because some of these variables are alligned with the outcome to some extend by random chance.

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	,690[a]	,477	,466	148313,785
2	,756[b]	,572	,450	150483,327

a. Predictors: (Constant), grade, sqft_living

b. Predictors: (Constant), grade, sqft_living, RND19, RND10, RND9, RND3, RND12, RND18, RND8, RND20, RND14, RND16, RND1, RND4, RND13, RND15, RND6, RND17, RND2, RND11, RND5, RND7

Coefficients[a]

Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
1	(Constant)	-318985,277	123358,639		-2,586	,011
	sqft_living	115,708	32,834	,368	3,524	,001
	grade	77790,963	21506,050	,378	3,617	,000
2	(Constant)	-235807,022	137669,849		-1,713	,091
	sqft_living	120,234	38,128	,383	3,153	,002
	grade	65475,457	24187,818	,318	2,707	,008
	RND1	3832,793	15281,763	,020	,251	,803
	RND2	15291,531	15369,171	,084	,995	,323
	RND3	-22208,967	15849,571	-,119	-1,401	,165
	RND4	15985,829	19366,155	,071	,825	,412

RND5	-10685,191	18072,117	-,053	-,591	,556
RND6	7940,422	17328,540	,040	,458	,648
RND7	-26094,531	19446,220	-,122	-1,342	,184
RND8	-1518,301	16405,738	-,008	-,093	,927
RND9	-18576,104	15931,616	-,096	-1,166	,247
RND10	15002,114	18428,021	,069	,814	,418
RND11	-22179,160	17455,542	-,115	-1,271	,208
RND12	-1717,518	16795,635	-,008	-,102	,919
RND13	11413,491	18627,803	,054	,613	,542
RND14	5233,468	19653,514	,022	,266	,791
RND15	2369,449	16898,231	,012	,140	,889
RND16	-7014,619	18976,595	-,032	-,370	,713
RND17	6462,126	15814,416	,036	,409	,684
RND18	-19081,517	18459,133	-,084	-1,034	,305
RND19	14498,961	21202,507	,059	,684	,496
RND20	22909,002	19550,777	,102	1,172	,245

a. Dependent Variable: price

That is why we need to use model fit indexes that are more sensitive to the number of variables we included as predictors, to account for the likelihood that some variables will show a correlation by chance. Such as adjusted R^2, or the AIC. The likelihood ratio test for the difference between the different models is also sensitive to the number of predictors in the models, so it is not easy to fool by adding a bunch of random data as predictors. These indices should all indicate that the simpler model only including sqft_linving and grade is better than the model including the random predictors.

Result-based models selection

(Result-based models selection is only mentioned here with demonstration purposes, to show how it can mislead researchers. Whenever possible, stay away from using such approaches, and rely on theoretical considerations and previous data when building models.)

After seeing the performance of mod_house_rand_train, and not knowing that it contains random variables, one might be tempted to build a model with only the predictors that were identified as having a significant added predictive value, to improve the model fit indices (e.g. adjusted R^2 or AIC). And that would achieve exactly that: it would result in the increase of the indexes, but not the actual prediction efficiency, so the better indexes would be just an illusion resulting from the fact that we have “hidden” from the statistical tests, that we have tried to use a lot of predictors in a previous model.

Excluding variables that seem "useless" based on the results will blind the otherwise sensitive measures of model fit. This is what happens when using automated model selection procedures, such as backward regression.

In the example below we use backward regression. This method first fits a complete model with all of the specified predictors, and then determines which predictor has the smallest amount of unique added explanatory value to the model, and excludes it from the list of predictors, refitting the model without this predictor. This procedure is iterated until there is no more predictor that can be excluded without significantly reducing model fit, at which point the process stops.

We can run backward regression by in the **Analyze > Regression > Linear** panel selecting "backward" instead of "enter" in the dropdown menu in the middle of the panel. So lets build a simple multiple regression model (not a hieararchical regression) where we would predict price with sqft_living, grade, and al the random predictors, with selecting Backward as a method (instead of Enter). The output will show the process through which predictors were excluded one-by-one until the program arrived at a model from which it could no longer exclude any predictors without significantly decreasing its predictive efficiency (based on the likelihood ratio test's p-value. If you want, you can adjust the rule that is used for this exclusion process in the regression panel in the Options menu).

The final model with the reduced number of predictors will have much better model fit indexes than the original complex model, because the less useful variables were excluded, and only the most influential ones were retained, resulting in a small and powerful model. Or at least this is what the numbers would suggest us on the training set.

Lets compare the prediction performance of the final model returned by backward regression with the model only containing our good old predictors, sqft_living and grade on the training set.

All of the model comparison methods (Adj. R^2, AIC, F-test of the likelihood ratio test) will indicate that the final model that the backward regression arrived at (we will refer to this as the backward model) performs better. We know that this model can't be too much better than the smaller model, since it only contains a number of randomly generated variables in addition to the two predictors in the smaller model. So if we would only rely on these numbers, we would be fooled to think that the backward regression model is better.

Testing performance on the test set

A surefire way of determining actual model performance is to test it on new data, data that was not used in the "training" of the model. Here, we use the set aside test set to do this.

Note that we did not re-fit the models on the test set, we use the regression equation produced by the models fitted on the training set to make our predictions on the test_set!!! You can do this by opening the other datafile containing the test-set, and in the **Transform > Compute Variable** tab computing the predicted values based on the regression equation obtained from the training set datafile. For example in my case, I would enter into the compute field, to compute the predicted value of price on the test-set

Using the simple model:

-318985,277 + 115,708* sqft_living + 77790,963*grade

Using the backward model:

-270028,072 + 118,417*sqft_living + 70653,440*grade -23318,155*RND3 - 28667,658*RND7 + 28105,694*RND20

Now you can calculate the residual sum of squares by comparing the actual price and the predicted price by the two models, and compare the two residual sum of squares produced by the two models.

This test reveals that the backward regression model has more error than the model only using sqft_living and grade.

BOTTOM LINE

1. Model selection should be done pre-analysis, based on theory, previous results from the literature, or conventions on the field. Post-hoc result-driven predictor selection can lead to overfitting.
2. The only good test of a model's true prediction performance is to test the accuracy of its predictions on new data (or a set-asid test set)