IMT574 Problem Set 6: Condition numbers and regularization

February 29, 2020

Introduction

This individual problem set has three aims:

- learn to find problematic data (multicollinearity) using condition numbers
- see how Ridge and Lasso regressions are more stable in case of ill-conditioned data
- learn to manually create design matrices for a survey data set.

We return here to the World Value Survey data, a large survey with ~90k observations and 430 variables. Your task is to compile a large number of variables into a design matrix. The process easily leads to multicollinearity which you have to detect using condition numbers.

Thereafter we overfit, but as this is a fairly large dataset, we only achieve this by selecting a small subsample of the data. You can see that overfitting is often related to large condition numbers, and may lead to linear regression validation RMSE to be ridiculously large. But Ridge and Lasso can handle the situation much better.

When reasonably coded, the code should run fast (~30s), except the stepwise condition number procedure (~10min). So you may want to be careful and not run that code too often.

Please submit a) your code (notebooks, rmd, whatever) *and* b) the results in a final output form (html or pdf).

World Values Survey

World Value Survey (WVS) is a large survey, conducted in many countries simultaneously. It revolves around public opinion about traditions, economy, politics, life and other things. I recommend you to consult the official questionnaire (uploaded in canvas/files/wvs).

In this task the central question is V23: "All things considered, how satisfied are you with your life as a whole these days?" with answers ranging between 1 (completely dissatisfied) and 10 (completely satisfied). We are going to model this variables using linear regression.

1 Explore and clean the data

First, let's load data and take a closer look at it.

1. Browse the WVS documentation and make sure you are familiar with coding of the variable *V23*. Note: you also have to consult the codebook to understand all the missings and how to remove those.
2. Load the data. Remove all the missing observations of *V23*. I mean all the missings, including the valid numeric codes that denote missing/invalid answers.

3. Now make a table (or a plot) of different answers. What is the mean satisfaction level on this planet? How large a proportion of people are at 6 or more satisfied?

   (Note: without knowing more about how the sample was created, we should not talk about the planet. We should refer to "respondents" instead.)

2 Create the Design Matrix

Now it is time to make the data suitable for a regression model. So far we have either used R-style formulas, or fed data into a ML model directly without much preparatory work. Now it is time to construct the *design matrix* manually. In case of linear regression, the design matrix is the data matrix that will be directly fed into the formula $(\mathtt{X}^\intercal \cdot \mathtt{X})^{-1}\mathtt{X}^\intercal\mathtt{y}$, or any function that uses this or another similar formula. Design matrix can also be fed directly into other kind of models, such as logistic regression or decision tree. Design matrix is also needed by various libraries, in particular *sklearn*'s *LinearRegression*, *Ridge*, and *Lasso*. There is an example of how to create a design matrix in the lecture notes, Section 2.1.7.

Your task is to add the variables to the design matrix, one-by-one, and each time doing the necessary encoding if appropriate.

- Many variables are categorical. For instance, variable *V2*, country, is numeric with different numbers representing different countries. So in essence it is a categorical variable where categories are coded as numbers. The same is true for *V80*, most serious problem in the world. You should convert such variables to dummies (do your still remember `pd.get_dummies`?) and remove the original variable. But don't forget to remove missings!

- A large number of variables contain ordered values instead. For instance, *V55* asks how much choice do you feel do you have over your life. The answers range from 1 (no choice at all) to 10 (a great deal of choice). We treat these as numeric response. Although, strictly speaking not correct, the model would be too messy if we were creating a category for each response. However, the missings (-5: inapplicable, -4: not asked etc) are not ordered in any meaningful sense. Hence your task is to remove missings.

Note that many variables, e.g. *v74b* (important to help people nearby) and *v90* (signing petition), contain a very large number of missings, and hence you essentially lose all your data if you include such variables. So you should remove such variables and replace with others that have more valid answers.

Was it clear? Good. Enough of talk, let's dive into the real thing.

1. Create your outcome variable $\mathtt{y}$ out of life satisfaction *V23* (remove missings!)

2. Create a design matrix that contains at least 100 variables from the WVS data. Your selected variables should contain at least a few categorical ones, such as *V2* country. In each case:

   (a) remove missing observations

   (b) convert categorical variable to dummies if appropriate. Don't forget to drop the reference category.

   This will result in a large amount of code that is essentially copy-paste, but not exactly. It is a little bit tedious to do though. Think about options to make it more automatic but... hint: there are no good options...

   Note that when converting 100 variables into a design matrix, the latter may end up with many more columns.

Keep in mind that you end up deleting quite a few observations, everything that contains missings in any of your selected variables.

3 Condition numbers

The next task is to compute the condition number of your design matrix. It will quite likely be way too big: some of the answers may be highly correlated, you may forget to drop the reference category, the reference category you drop may have no observations, there may be several variables that contain exactly the same information... But we want to know which variables are the culprits. So instead of computing just a single κ, let's add columns to the design matrix one-by-one, and each time printing the column we added, and the resulting condition number.

1. Compute the condition number of your output matrix in such a manner. The output may look something like:

   ```
   condition numbers
   country31, 1 columns, k= 1.0
   country32, 2 columns, k= 1.0039840795232402
   country36, 3 columns, k= 1.2079245987756626
   country48, 4 columns, k= 1.2079245987756626
   country51, 5 columns, k= 1.2079245987756626
   ...
   v50, 145 columns, k= 8115.798611270699
   v51, 146 columns, k= 8325.72582639652
   v52, 147 columns, k= 8640.26470767111
   v53, 148 columns, k= 8866.77374231709
   v54, 149 columns, k= 9020.840804566187
   v55, 150 columns, k= 10673.439284968747
   v56, 151 columns, k= 11609.942814916583
   ...
   ```

   Hint: this is slow (10min). You may run these lines of code only occasionally, when you test new variables.

2. if the condition number turns out too big (say, over 100,000 or so), identify the culprit, and fix it in the design matrix above. In some case you may remove the variable and replace by another one, you may also consider merging small categories into a larger one or something else. You may go back-and forth quite a few times before you get a suitably well-conditioned design matrix.

You final result should be a nice design matrix X with condition number that is not outrageously large.

4 Do Some Social Science

Before getting further, let's do a simple social science analysis. How is life satisfaction related to health (*v11*), perceived control over life (*v55*) and financial situation (*v59*)? Let's analyze association between satisfaction and just these three variables.

1. run a linear regression models explaining satisfaction with these three variables. Present the output table.

   I recommend to use *statsmodels.formula.api* for this task (but you have to use *sklearn* later).

2. comment the output table in terms of relative effect size and statistical significance. Any surprises for you?

3. compute and present RMSE (just on training data). This will serve as the benchmark for the future.

5 Back to ML: Model

Now it is time to use all these variables to model satisfaction. Use `sklearn.linear_model.LinearRegression` here as this is easy to be switched with ridge and lasso, and it takes in the design matrix directly.

1. compute the condition number for your design matrix (just a single number, not the stepwise procedure).
2. Split the data into training-validation chunks (80-20 or so)
3. compute the condition number for your training design matrix (just a single number, not the stepwise procedure).
4. fit a linear regression model where you describe satisfaction with the design matrix X you just created.
5. predict and compute RMSE on training data
6. predict and compute RMSE on testing data
7. repeat the previous with Ridge regression, play a little with different α-s. Which α gave you the best testing RMSE? (No need for a rigorous analysis, just play a little)
8. and repeat with Lasso regression again playing a little with different α-s.
9. comment your results:
   (a) compare RMSE on testing/training data. What does this suggest in terms of overfitting?
   (b) compare RMSE for OLS, Ridge and Lasso
   (c) compare the resulting RMSE with the small benchmark model you did above

If your results are like mine, you see that a) RMSE on both testing-training sets are similar; b) RMSE for OLS, Ridge, Lasso are similar; and c) all these 100 or so extra variables add very little explanatory power to the model.

6 Let's Overfit!

As WVS is a relatively large dataset we cannot easily overfit by adding more variables. But we can go another easy route instead: we take a subsample.

1. Create a subsample of your design matrix and the outcome variable. Choose a large-ish sample that overfits.

   The size depends on which variables do you exactly choose, in my case 2000 obs rarely overfits (it depends on the train-validation split), 1000 typically overfits.
2. repeat the steps you did above.
3. comment how do OLS, Ridge, Lasso perform on testing/training in case of overfitting.
4. comment the condition number of design matrix and overfitting.