Intermediate Econometrics Homework

Alexis Naudin & Pau Barba

22/11/2017

```
install.packages("caret")
install.packages("tidyverse")
install.packages("neuralnet")
install.package("np")
istall.package("dplyr")
install.package("caret")
install.packages("knitr")
```

Exercise 1

Step 1:Neural networks

In this exercise we will be using a neural network in order to estimate the regression. A neural network is a set of artificial neurons which take inputs and outputs, each neurons are connected by weighted connections. A particular network multiplies the value of the connection by the value of the neuron and applies a function before passing it to the next layer of neurons. Machine learnig fits a set of neurons and weights which best predict the output of the training data, and then use it to preduct the test data.

Firt off we clean the data by eliminating the missing values. Then we proceed to eliminate the data which is far from beeing significative in an ols, in order to not overcrowd the imputs with irrelevant data, which whould make it more difficut to establish correct relations. We do that by filtering out all the variables which have a p-value of 10% or higher on a standard OLS regression.

Step 2: Training the model

We train the model with the training data with a neural network of 40 neurons, with the possibility to skip a layer.

```
## # weights:  819
## initial  value 56449248860439.523437
## iter  10 value 6076526876915.302734
## iter  20 value 2466245013992.471191
## iter  30 value 2152515261488.982910
## iter  40 value 1816261879786.533447
## iter  50 value 1739707739229.120361
## final  value 1619569087056.497803
## converged
```

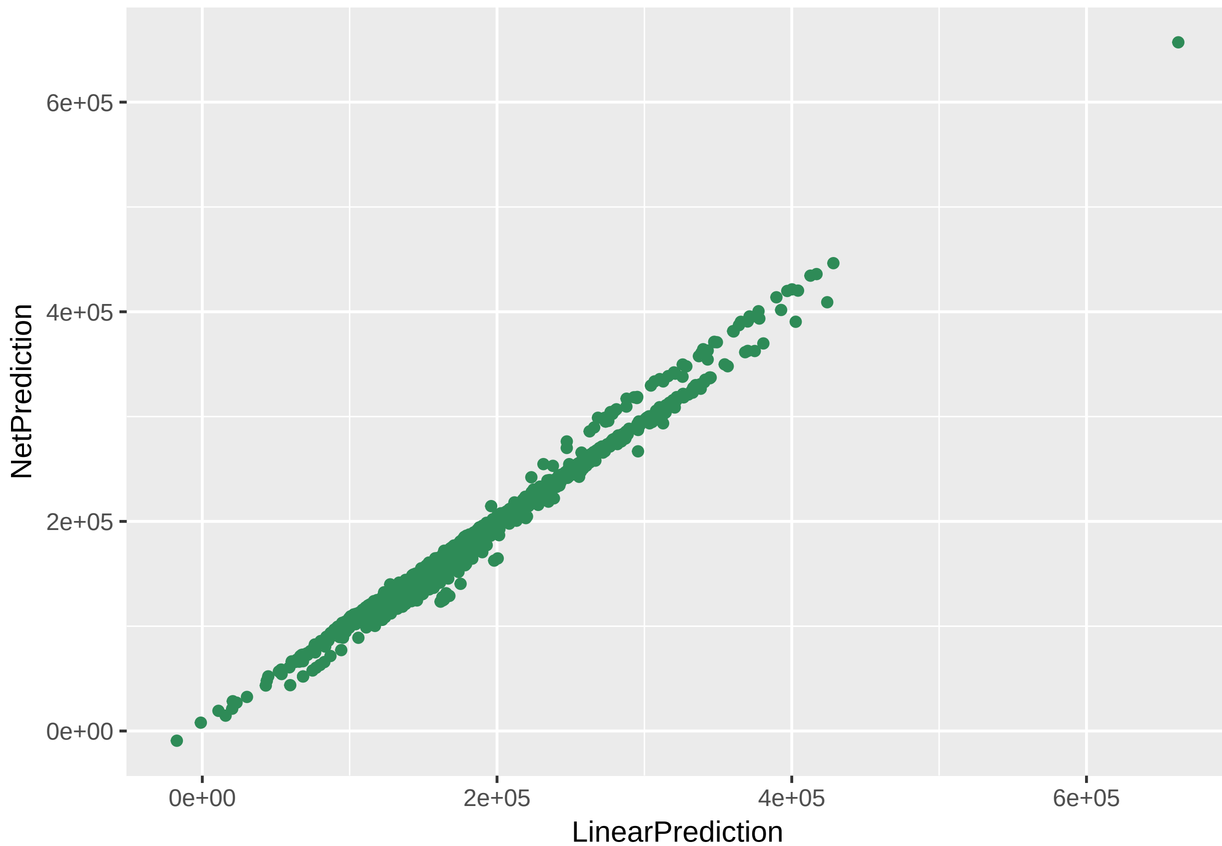
Step 3: Predicting with the model

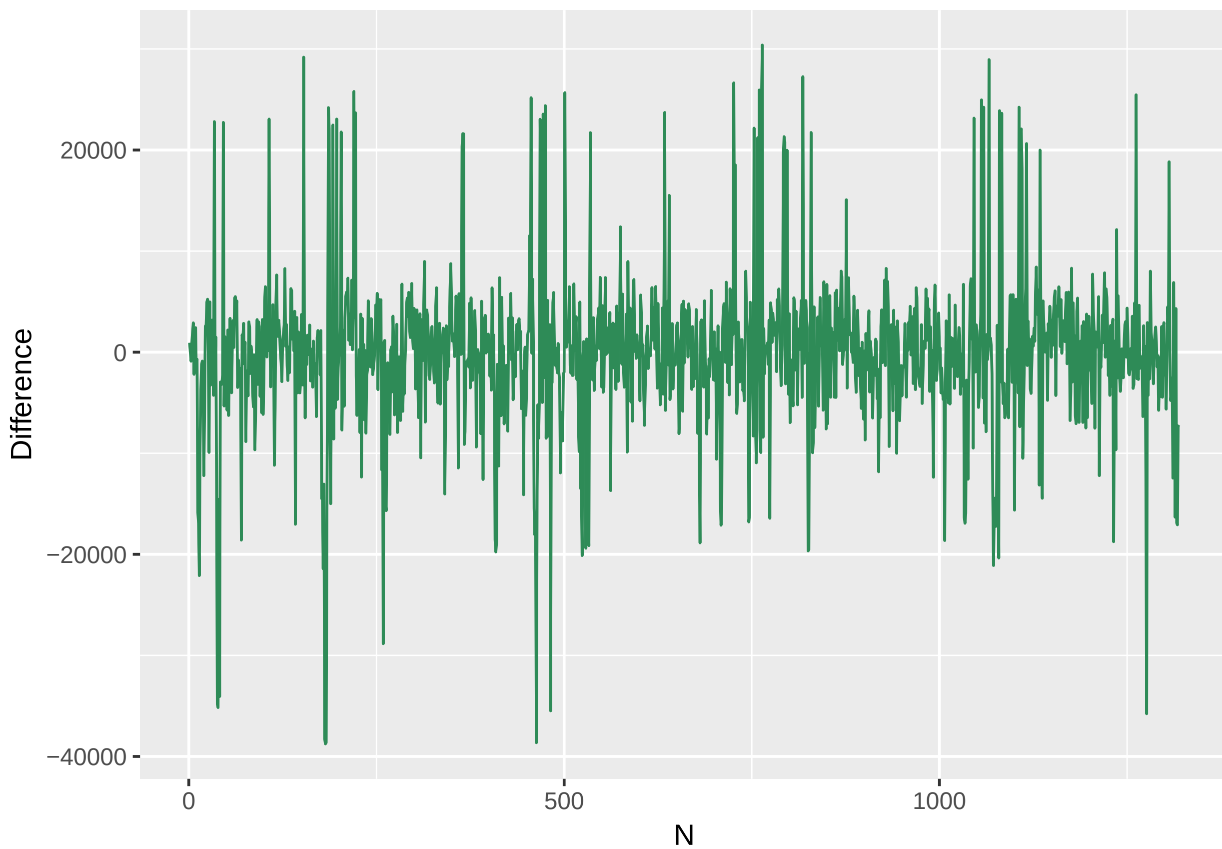
Last, we use both the neural network and a standard OLS to predict the results of the regression. In order to compare both results we two different graphs:

The first one plots the prediction of the linear results versus that of the neural net prediction.If the two methods were equal we would see a perfect 40 degree line. In this particlar seed we se that there seem to be two different trends in the graph, that could indicate that one of the two methods has found a relation in a particular variable that the other one has missed, which brings the price up.

The second one plots the differene of the two predictions and we can see a similar result. On some cases the neural net predics values above whilst in some others it predicts below, but the deviations seem to be consistent, which seems to indicate different weights for particular variables.

* note this seed is rather particular, and we see this double trend effect. Under most seeds the linear prediction is constantly higher which leads to a difference graph biased downwards.

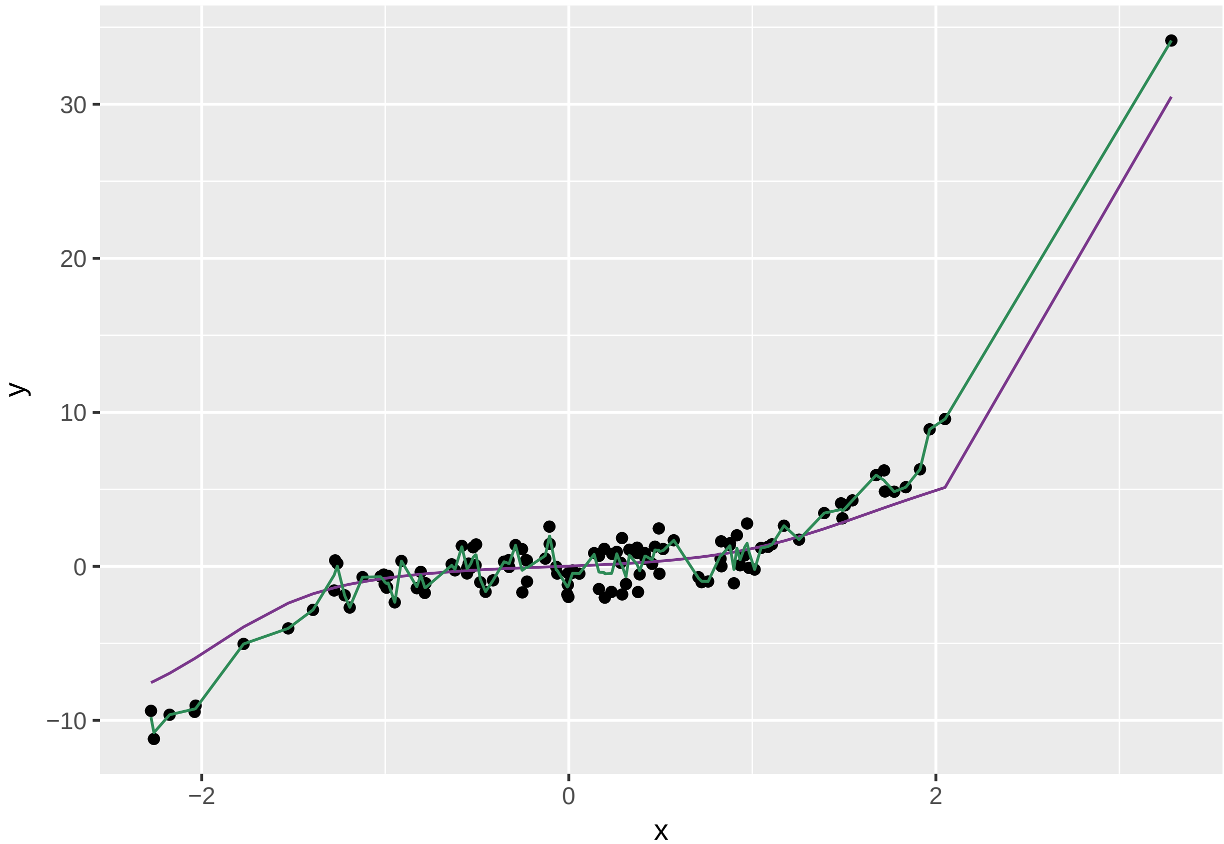




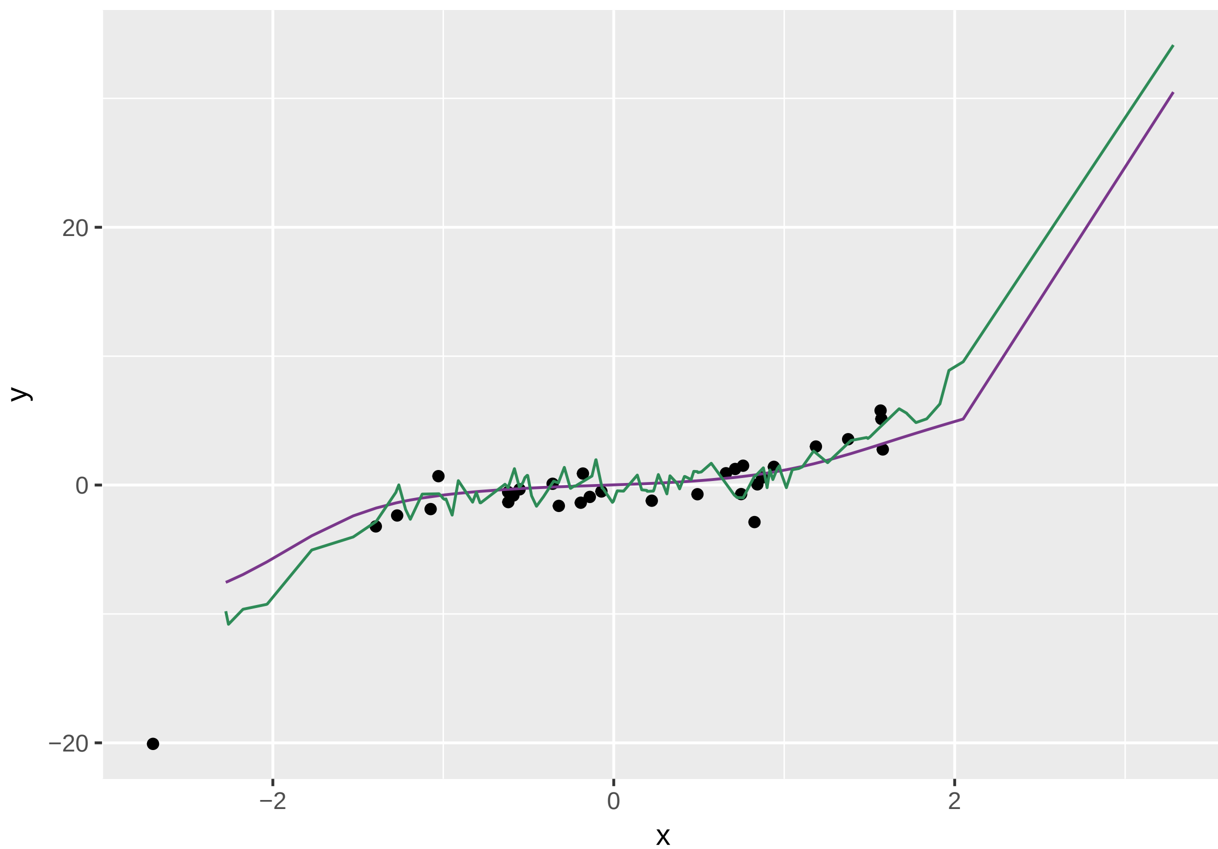
Exercise 2

First we generate the data and create a partition for the training and the testing data.Next we fit the model with low and high flexibility.

When we plot the data we observe that the high flexibility model is much more variant. We can see that the low flexibility model has more bias since the other one has a much better fit for this particular data.



We plot the test data and we see that this time the high flexibility model still has more variance, but it also has a higher bias since it is overfited to the training data and does not reflect on the test data.



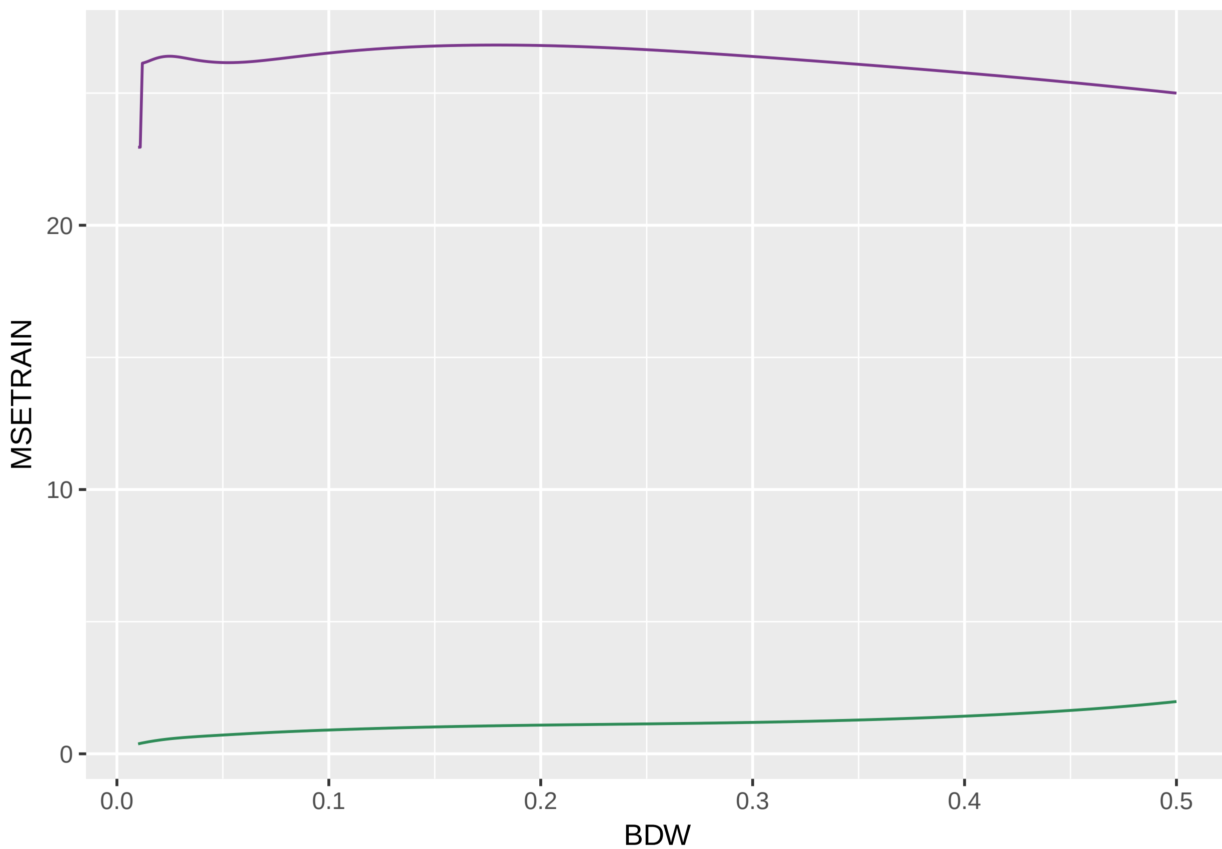
Checking Mean Square Error

First we create a vector for the bandwidth

We then create two empty matrix which will store the residuals for both models.Then we make a loop which runs the regression for both data sets and stores the residuals in the matrix.

We then compute the mean squared error, by doing the mean of the matrix rows, we are then left the mean residuals for the regressions which we can square to obtain the mean squared error.

We proceed to plot the data. The data for the train variable appears in green, while the one for the test data appears in purple. First off we can see that the train errors are always going to be lower since we are using the actual data, and since it's the actual data there is no risk of over fitting therefore the best model is going to be the most flexible (lowest bandwith). In the train model we can see that the shape is more interesting, the lowest poit seems to be arround 0.6, lower flexibility increases error due to less acurate preddictions, but more flexivility also increases error since it runs the risk of overfitting the data.



Task3

In this exercise we download the data sets (we will be using a freduced version of the SIREN dataset because it melted one of our computers and crashed the other one), load them and merge them by their SIREN number.Then we plot the histogram for the size of the variables which were in CNIL. (note that by size we understood CATEGORIE which refers to the categorical classification of the company by size)

```
## # A tibble: 18,629 x 8
##     ï..Siren              Responsable                              Adresse
##        <int>                   <fctr>                               <fctr>
##  1 788349926     "\"LA RIVE BLEUE\""                         3/5 RUE BOILEAU
##  2 421715731                01 DIRECT              58 AVENUE DE RIVESALTES
##  3 409869708             01DB-METRAVIB               200 CHEMIN DES ORMEAUX
##  4 444600464                1.2.3. SAS        57-59 -61 RUE HENRI BARBUSSE
##  5 922002968            100 % ASNIERES             70 AVENUE D'ARGENTEUIL
##  6 429621311               1000MERCIS               28 RUE DE CHATEAUDUN
##  7 429621311               1000MERCIS               28 RUE DE CHATEAUDUN
##  8 453465379               118000 SAS               38, RUE DES JEUNEURS
##  9 528420565 123 MEDIA COMMUNICATION                        64 GRAND RUE
## 10 524772753                      2&GO Z.A DU COUQUIOU - 433 AV. DU CLAPIER
## # ... with 18,619 more rows, and 5 more variables: Code_Postal <fctr>,
## #   Ville <fctr>, NAF <fctr>, TypeCIL <fctr>, Portee <fctr>
```

```
## Warning in scan(file = file, what = what, sep = sep, quote = quote, dec =
## dec, : EOF within quoted string
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```