Human vs. Computer: Can we teach the computer to read residual plots?

A thesis submitted for the degree of

Master

by

Shuofan Zhang

Master, Monash University

Department of Econometrics and Business Statistics

Monash University

Australia

May 2018

Contents

Acknowledgements

I would like to thank my supervisor, Di, for being patient with me as always.

Declaration

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma in any university or equivalent institution, and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Shuofan Zhang

Abstract

When we fit a linear regression model, with one continuous dependent variable and some explanatory variables, many problems may occur. For example, the relationship between dependent-explanatory is not linear; the error terms may be correlated to each other; the variance of errors may not be constant; outliers and high leverage points may be present, etc. Plots of residuals versus predicted (or fitted) value are a useful graphical tool for detecting most of the problems (Tibshirani et al., 2013). It can be difficult for beginners to learn how to recognize patterns seen could arise by chance and that the model is proper.

Computer vision has advanced rapidly in recent years, primarily by building deep learning models.

Therefore, in this paper, we are going to investigate wether deep learning neural network could be used to teach a computer to do better than a human on reading residual plots. Two of the issues, non-linearity and heteroskedasticity, will be addressed here. After training the model, we will compare its performance with human database.

Chapter 1

Introduction and literature review

"The multiple regression model for cross-sectional data is still the most widely used vehicle for empirical analysis in economics and other social sciences" (Wooldridge, 2015). Detecting possible violations of the Gauss-Markov assumptions is crucial to interpret the data properly, especially in the early stage of analysis. There are several distribution tests that are commonly used, for instance, the Pearson correlation test for detecting linear relationship; the Breusch-Pagan test and White test for investigating heteroskedasticity. But primarily residual plots are the main diagnostic tool and these rely on human evaluation. Because data plots show a lot more information than a single statistics. A good example here would be Anscome's Quartet. "It is a set of four distinct datasets each consisting of 11 (x,y) pairs where each dataset produces the same summary statistics (mean, standard deviation, and correlation) while producing vastly different plots" (Anscombe, 1973). Matejka and Fitzmaurice also did an interesting study on this issue, they used 'datasaurus' data from Cairo (2016) and generated a series of data with same statistics but very different plots (Matejka and Fitzmaurice, 2017).

Former studies have shown that human eyes are sensitive to the systematic patterns in data plots. With proper manipulation, visualized plots can be used as test statistics and perform valid hypothesis test. One example of these protocols that provides inferential validity is lineup which is introduced by Wickham et al. (2010). "The protocol consists of generating 19 null plots (could be other number), inserting the plot of the real data in a

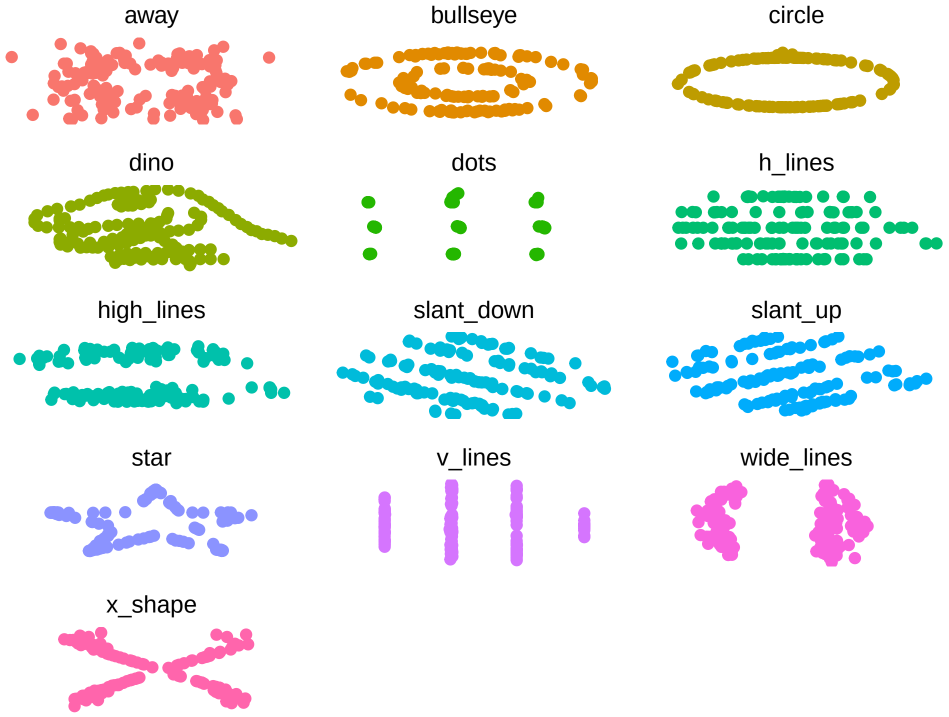

Figure 1.1: *Each dataset has the same summary statistics to two decimal places: (E(x)=54.26, E(y)= 47.83, sd(x) = 16.76, sd(y) = 26.93, Pearson's r = -0.06).*

random location among the null plots and asking the human viewer to single out one of the 20 plots as most different from the others" (Wickham et al., 2010). If the real plot is chosen, it means the real data is different from the null hypothesis, so we reject the null hypothesis with 5% chance to be wrong (Type I error). Figure 1.2 is an example of lineup. Which plot do you think is the most different? If you choose one, we are 95% confident to reject the no-relationship assumption between the two variables, hp and disp (Simchoni, 2018). This protocol has proved valid and powerful therotically as well as practically through human experiments, especially when the assumptions for doing conventional tests are violated (Majumder, Hofmann, and Cook, 2013).

The question that arises today is whether we can train a computer to read scatter plots, particularly with a computer vision approach such as deep learning.

Motivation for the task is provided in a blog post by Giora Simchoni (Simchoni, 2018). He has designed a deep learning model to test the significance of linear relationship between two variables for samples of size 50. The model reached over 93% accuracy on unseen test data. He also mentioned that the computer fails to pick up a strong non-linear

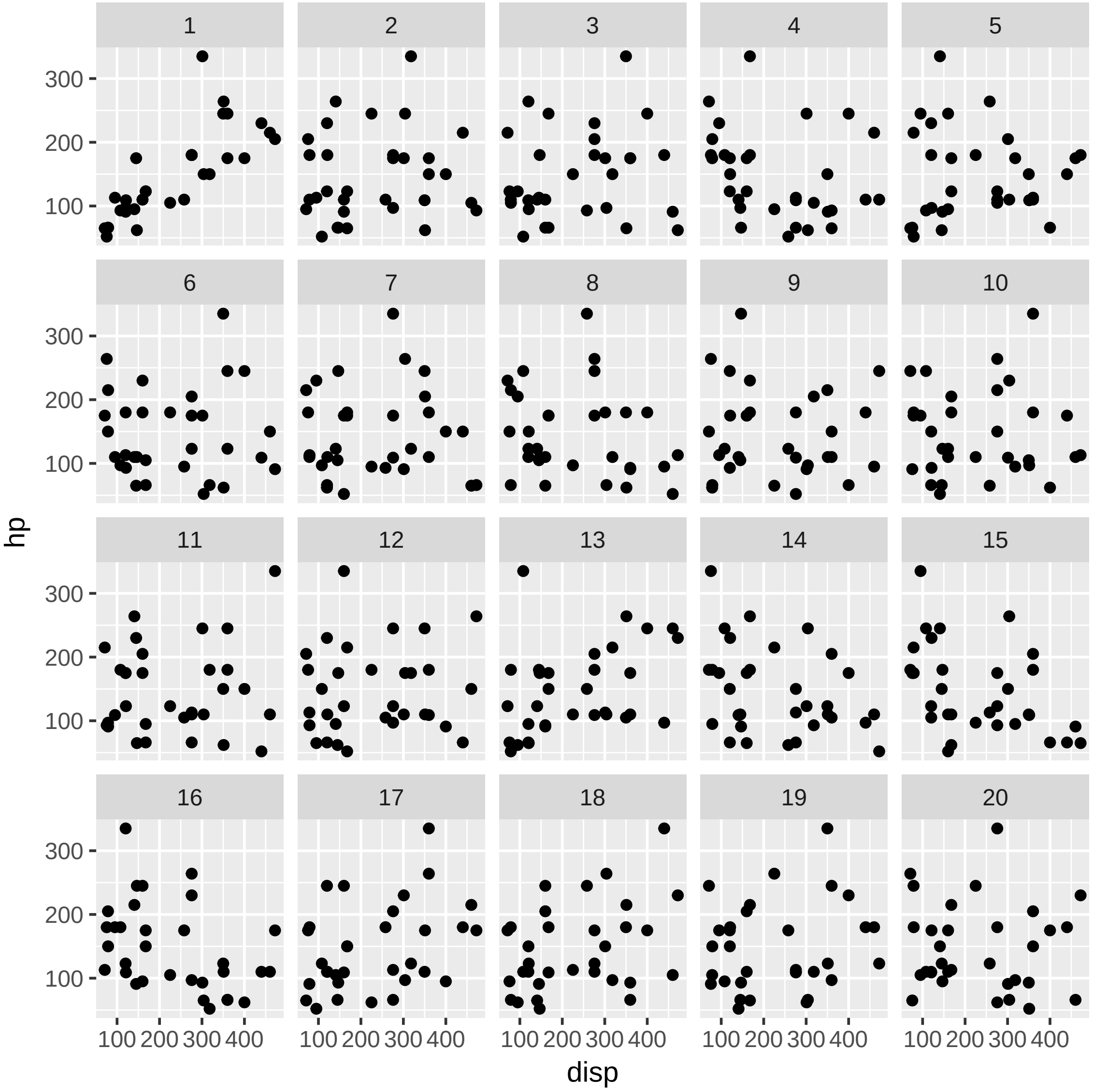

Figure 1.2: *Scatterplot lineup example: one plot is the data, the rest are generated from a null model assuming no relationship between the two variables. In this lineup it is easy to see that plot 1, which is the data plot, is different from the rest.*

relationship even though the Pearson'r is as high as -0.84 (Simchoni, 2018). So the short conclusion is the computer vision is not perfect, in that it is not as flexible as human vision. As Simchoni explained in his article, the model can only distinguish linear relationship from no-relationship as trained. However, we think this fact is just another example reflecting the importance of visualization as we discussed above. Strong correlation does not necessarily mean linear relationship. We should always refer to the plot before making any statement. What's more, if we want the model to be more flexible, we could simply adjust our design of training accordingly. Therefore, in this article, we are trying to further Simchoni's study. More specifically, the deep learning model will be trained to perform three hypothesis tests as following.

H_0: There are no relationships between the two variables. H_1: There is linear relationship between the two variables where all Gauss-Markov assumptions are met.

H_0: There is linear relationship between the two variables where all Gauss-Markov assumptions are met. H_1: There is linear relationship between the two variables where the variance of the error term is not a constant while all other Gauss-Markov assumptions are met.

H_0: There is linear relationship between the two variables where all Gauss-Markov assumptions are met. H_1: There is non-linear relationship between the two variables where all other Gauss-Markov assumptions are met.

To facilitate our study, we focus on the regression model with only one explanatory variable. The associated results can be extended and employed to multiple regression cases with modifications to the data in practice. Because the "statistics" we will use is the data plot or the residual plot, in terms of teaching the computer of reading these plots, one variable is enough for us to generate different patterns in that plot for convnets to learn. And this makes the design process much simpler.

The model we will use is the convolutional neural networks, also known as convnets, a type of deep-learning model almost universally used in computer vision applications (Chollet and Allaire, 2018). Unlike the classical programming where human input rules, in deep learning paradigm, we provide data and the answers associated with the data.

Deep learning algorithm will output the rules, and these rules can then be used on new data to make predictions. We can also think of the deep learning neural network as a complex nonlinear model which could estimate millions of parameters with big enough dataset. As usual regression problem, to get the estimates of unknown parameters, we need to provide the model with dependent variable and independent variables. In this case, the independent variable will be the images of residual plots simulated from the null distribution and the alternative distribution, and dependent variable will be the labels of that plot indicating the true relationship of the original data. Once we have these estimated parameters, we then can use them to classify unseen residual plots, eg. to perform hypothesis tests.

Chapter 2

Experimental design

Three experiments associated with the three hypothesis tests will be processed in this paper.

First experiment - Independent variables vs. Classic linear model Second experiment - Classic linear model vs. Linear model with Heteroskedasticity Third experiment - Classic linear model vs. Non-linear model

Main steps in the experiments are:

1. Simulate data from both the null distribution and the alternative distribution of each hypothesis test
2. Generate corresponding data plots for the simulated data
3. Save data plots as fixed-sized images with labels indicating which distribution they are from
4. Train a deep learning classifier to recognize different labels
5. Test the model's performance and compute the accuracy
6. Compare the model's performance with human's (exclusive for the first and second experiment)

2.1 Data simulation and model design

2.1.1 Classic linear model

This is the model implied in the alternative hypothesis of the first experiment and the null hypothesis of the second and third experiment in this paper. The design for this model is similar to what Simchoni (2018) did in his blog but is tailored to perform the comparison with human performance later on. After the deep learning model is trained on data generated in this section, it will be tested on the 70 data plots from the second experiment in Majumder, Hofmann, and Cook (2013). This model's accuracy will then be compared to the human's performance as well as the Pearson's correlation test.

The model is designed as:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \quad i = 1, \ldots, n$$

with elements of the model generated by the following processes

- $X \sim N[0,\ 1]$

  Distributions of X has impact on the shape of the scatters. For instance, if X is generated from a uniform distribution, then the plots will look like a square especially when the sample size is large; while more like a circle if X follows normal distribution. In this experiment, we will set it to be normal to match with the design in the second experiment of Majumder, Hofmann, and Cook (2013).

- $\beta_0 = 0$

  Intercept is set to be zero, because it will not change the data plots since all data will be standardized before being plotted. Any constant intercept has same effect on the data plots.

- $\beta_1 \sim U[-10, -0.1] \bigcup [0.1, 10]$

  β_1 is designed to be uniformly generated from -10 to 10 (excluding -0.1 to 0.1) to cover the range used in the second experiment in Majumder, Hofmann, and Cook (2013).

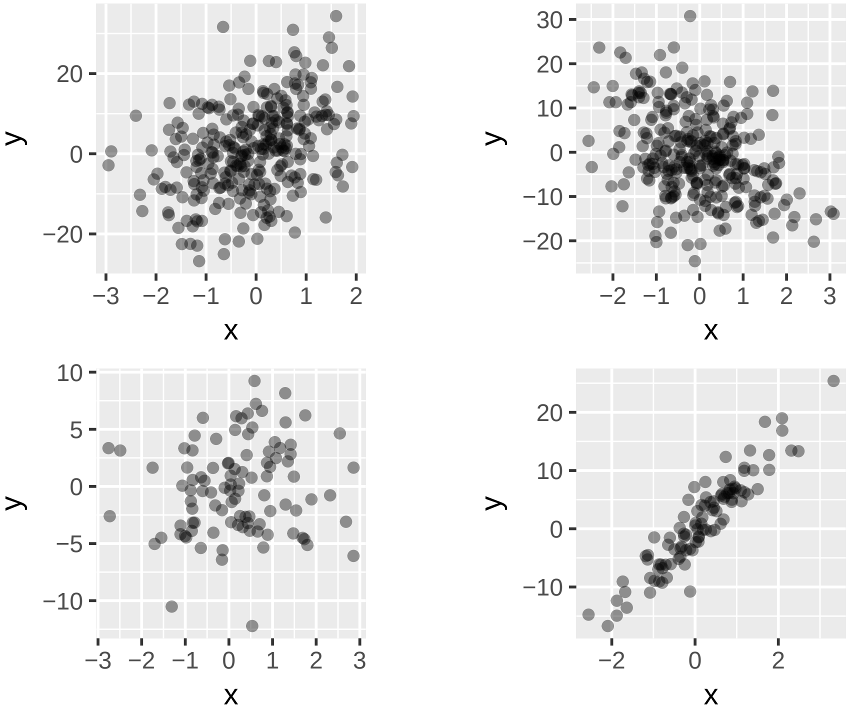

Figure 2.1: *Four examples of data plots generated from the classic linear model*

- $\varepsilon \sim N(0, \sigma^2)$ *where* $\sigma \sim U(1, 12)$

  ε is designed to be uniformly generated from 1 to 12 in order to cover the range used in the second experiment in Majumder, Hofmann, and Cook (2013).

- $n = 100, 300$

  Number of observations are the same with the the second experiment in Majumder, Hofmann, and Cook (2013).

2.1.2 Linear model with Heteroskedasticity

This is the model implied in the alternative hypothesis of the second experiment in this paper, where the constant variance assumption of the linear model is violated while all other conditions are met. By the definition given in Wooldridge (2015), “The homoskedasticity states that the variance of the unobserved error, u, conditional on the explanatory variables, is constant. Homoskedasticity fails whenver the variance of the unobserved factors changes across different segments of the population, where the segments are determined by the different values of the explanatory variables.” There are countless types of heteroskedasticity since the “change of the variance” could be related to “the explanatory

variables" in various ways. It is not feasible to list out all kinds of heteroskedasticity by a single function. For simplicity, we will focus on one example of them, a linear correlation between the explanatory variable X and the standard deviation of the error term. The conclusion from this experiment though can be generalized to more complicated cases.
A new human experiment will be rendered to produce the human's performance on detecting heteroskedasticity by reading residual plots. After the deep learning model is trained on the data generated from this section, it will be tested on the same data set that is used for the human experiment. And the accuracy of the model will be compared to the human's performance from this new human experiment. To provide a reference to evaluate how computer perform in detecing heteroskedasticity issues, a special case of the White test will be used. Each image in test set will be evaluated using White test. The associated accuracy of White test will be compared to computers' accuracy on the same set.

The model structure is the same with the classic linear model:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \;\; i = 1, \ldots, n$$

with elements of the model generated by the following processes

- $X \sim U[-1,\ 1]$ To better present the heteroskedasticity in the data, a uniform distribution of X is used instead of normal distribution. The range is set to be small (from -1 to 1) in order to produce the data with weak heteroskedasticity more frequently.
- $\beta_0 = 0$ Intercept is set to be zero for the same reason stated above.
- $\beta_1 \sim U[0.5,\ 1]$ β_1 has little impact in this case so it is set to be uniformly generated from 0.5 to 1. Because the residual plot but not data plot will be used here, therefore, the information contained in β_1 will be extracted by the linear regression we fit to the data.
- $\varepsilon \sim N(0,\ (aX + v)^2)$ The variance of the error term is a quadratic function of the explanatory variable which controls the magnitude of heteroskedasticity in the model.

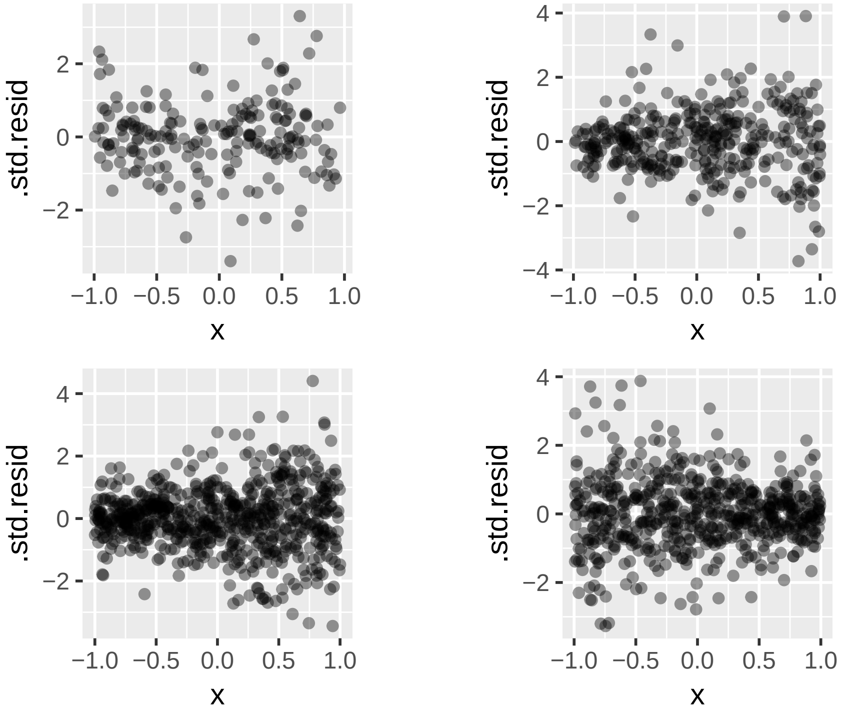

Figure 2.2: *(#fig:heter_plot)Four examples of residual plots generated from linear model with heteroskedasticity*

- $a \sim U(-5,\ -0.05) \bigcup (0.05,\ 5)$ The parameter a here, following uniform distribution from -5 to 5 (excluding -0.05 to 0.05), is the correlation coefficient between X and the standard deviation. Larger a gives stronger heteroskedasticity. This range is wide enough for our purpose.
- $v \sim N(0,\ 1)$ This new error term is added to the variance of ε so the relationship between the data can be more flexible.
- $n \sim U[20,\ 1500]$ The sample sizes will be randomly generated from 20 to 1500. We choose 20 to 1500, because when the sample size is smaller than 20, there is hardly any systematic patterns to see. And 1500 is large enough to give a good description about the true relationship within the data, and is light enough to be processed. Since we make the transparency of the points to be 0.4, more points being added to the plot will just make the plot looks darker.

In general, the choice of the parameters is an empirical work. Primarily, we want the residual plots to show more variation; on the other hand, we need to limit the range of these parameters in order to keep the key features in the data.

–>

2.1.3 Independent variables

This is the null scenario in the first experiment, eg. the two variables under tested are independent of each other. If the data arises from this situation, then the data plots will not show any systematic patterns theoretically.

The model is designed as:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \;\; i = 1, \ldots, n$$

with elements of the model generated by the following processes

- $X \sim N[0,\ 1]$
- $\beta_0 = 0$
- $\beta_1 = 0$
- $\varepsilon \sim N(0, \sigma^2) \; where \; \sigma \sim U(1, 12)$
- $n = 100, 300$

The specification of this model is the same with the classic linear model, the only difference is the correlation coefficient equals to zero in this case.

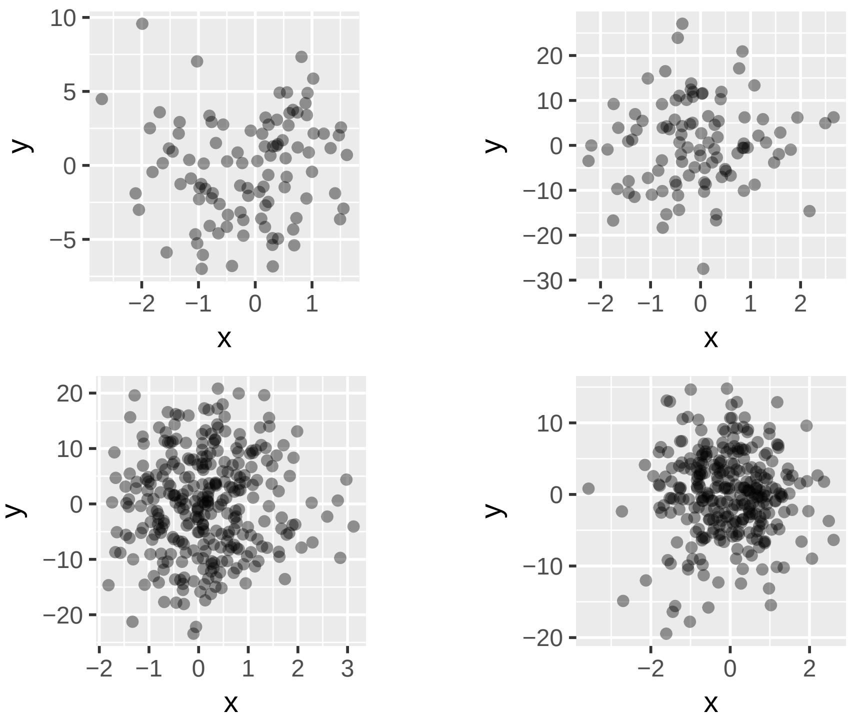

Figure 2.3: *Four examples of data plots generated with two independent variables*

2.2 Generate corresponding plots for the simulated data

2.2.1 First experiment - Independent variables vs. Classic linear model

The plot used for training and testing in the first experiment is the scatter plot between the dependent variable Y and the independent variable X.

2.2.2 Second experiment - Classic linear model vs. heteroskedastic

For the second experiment, a linear model will be fit to the data firstly. Residuals from the fitted model will be standardized and extracted. The residual plot will be made of standardized residuals against X.

2.2.3 Third experiment - Classic linear model vs. Non-linear model

The plot in the third experiment will be the same with the second experiment.

2.3 Save data plots as fixed-sized images with labels

We will use ggplot2 package in R to generate relevant data plots and use ggsave funtion to resize and save all plots to our local drive. So all plots will be a squared image with same width and height. This helps us to see better the patterns in the scatter plot and also is easier for the deep learning model to process.

As for the labels given the each image, we will use the true population as the samples' identification directly. It is true that there will be some undesired patterns formed out of randomness, especially when the sample size is small. Unlike what Simchoni did in his post, no conventional tests will be used to sort out the "significant" observations. Because the answer to the question that if the deep learning model can distinguish from patterns formed by chance and by nature is also interested.

2.4 Train a deep learning classifier to recognize different labels

All convolutional neural network related work will be done by Keras package in R. Three training processes will be done for the three hypothesis tests respectively. The structure of the deep learning model will be the same across three experiments.

```
## Model
## ___________________________________________________________________________
## Layer (type)                     Output Shape                  Param #
## ===========================================================================
## conv2d_5 (Conv2D)                (None, 148, 148, 32)          320
## ___________________________________________________________________________
## max_pooling2d_5 (MaxPooling2D)   (None, 74, 74, 32)            0
## ___________________________________________________________________________
## conv2d_6 (Conv2D)                (None, 72, 72, 64)            18496
## ___________________________________________________________________________
## max_pooling2d_6 (MaxPooling2D)   (None, 36, 36, 64)            0
## ___________________________________________________________________________
## conv2d_7 (Conv2D)                (None, 34, 34, 128)           73856
## ___________________________________________________________________________
## max_pooling2d_7 (MaxPooling2D)   (None, 17, 17, 128)           0
## ___________________________________________________________________________
## conv2d_8 (Conv2D)                (None, 15, 15, 128)           147584
## ___________________________________________________________________________
## max_pooling2d_8 (MaxPooling2D)   (None, 7, 7, 128)             0
## ___________________________________________________________________________
## flatten_2 (Flatten)              (None, 6272)                  0
## ___________________________________________________________________________
## dense_3 (Dense)                  (None, 512)                   3211776
## ___________________________________________________________________________
## dense_4 (Dense)                  (None, 1)                     513
## ===========================================================================
## Total params: 3,452,545
## Trainable params: 3,452,545
## Non-trainable params: 0
## ___________________________________________________________________________
```

2.4.1 First experiment - Independent variables vs. Classic linear model

180,000 samples are generated for each of the two groups in the first experiment where 100,000 of them is used for training; 40,000 of them is used for validation and 40,000 for testing. 10 epoches (1 epoch = 1 iteration over all samples) will be done by the model. The model specification given by each epoch will be saved, the one gives the highest test power is chosen to represent the computers.

From the plot of the traning history, we can see the fourth specification gives the overall best performance with lowest loss and highest accuracy on validation set. Its accuracy on the unseen test set is also the best among other models. After the fourth checkpoint, the model starts to overfit. ### Second experiment - Classic linear model vs. heteroskedastic

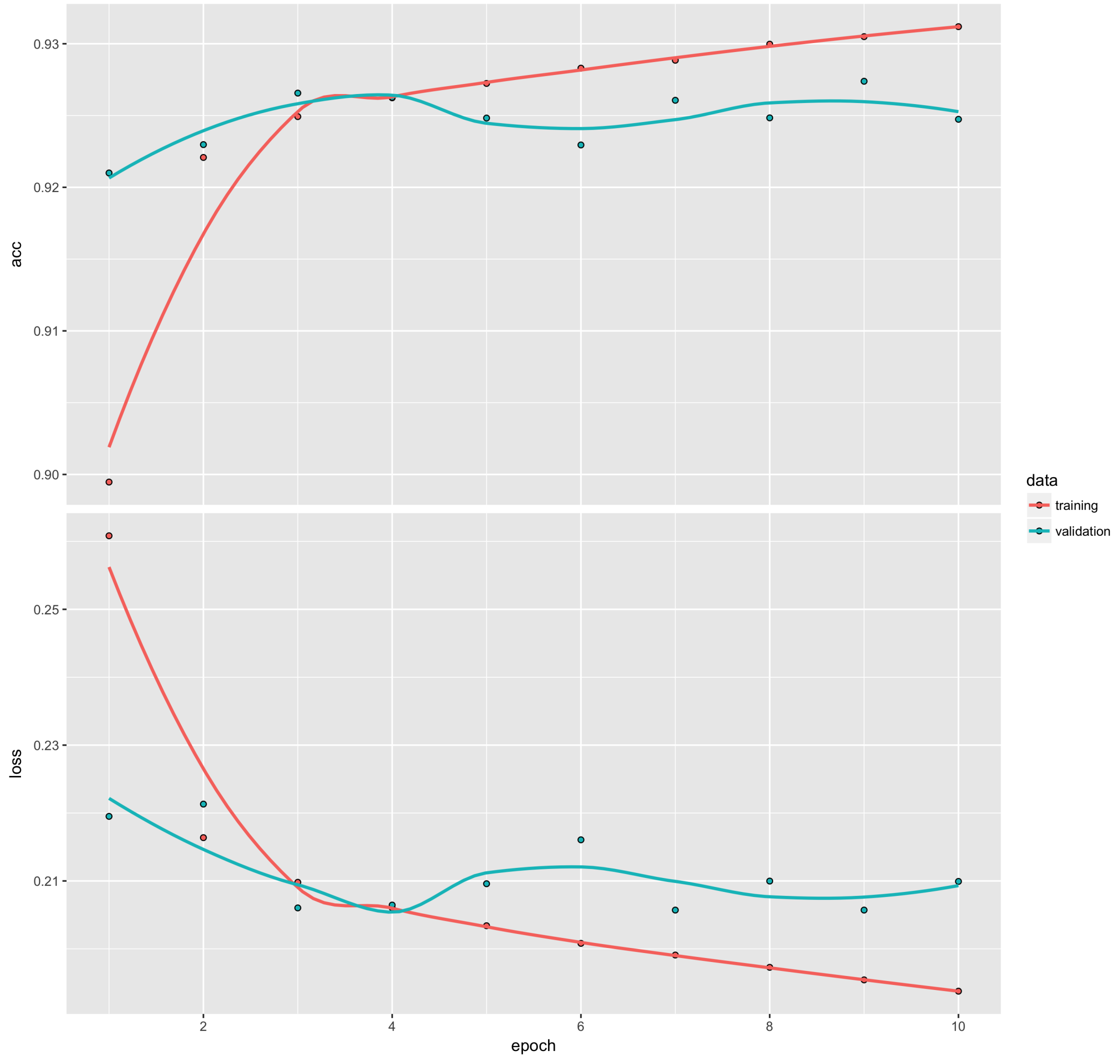

Figure 2.4: *Plot of the training history of the deep learning model for the first experiment.*

2.4.2 Third experiment - Classic linear model vs. Non-linear model

2.5 Test the model's performance and compute the accuracy

After 30 iterations, the accuracy of the training set has reached 99.25%, 93.3% for validation set, and 94% for test set. This performance is consistent with our expectation of the deep learning model.

2.6 Compare the model's performance with human's

Chapter 3

Comparison with Turk studies

A large database of results from a human subjects test conducted to validate the lineup protocol relative to a classical tests is going to be used for this part of the work. This data was collected as part of the work presented in Majumder, Hofmann, and Cook (2013). Experiment 2 examined the performance of humans in recognising linear association between two variables, in direct comparison to conducting a t-test of $H_o : \beta_k = 0$ vs $H_a : \beta_k \neq 0$ assessing the importance of including variable k in the linear model. An example lineup is shown in Figure 3.1. For this lineup, 63 of the 65 people who examined it selected the data plot (position 20) from the null plots. There is clear evidence that the data displayed in plot 20 is not from $H_o : \beta_k = 0$.

This experiment utilised 70 lineups of size 20 plot, with varying degrees of departure from the $H_o : \beta_k = 0$. There were 351 evaluations by human subjects. These results will be used for comparison with the deep learning model.

The trained deep learning model will be applied to the data from this experiment. The model will be asked classify each plot in each lineup. We will calculate how frequently the data plot is selected as not a null plot, and compare this to the frequencies obtained by human evaluation.

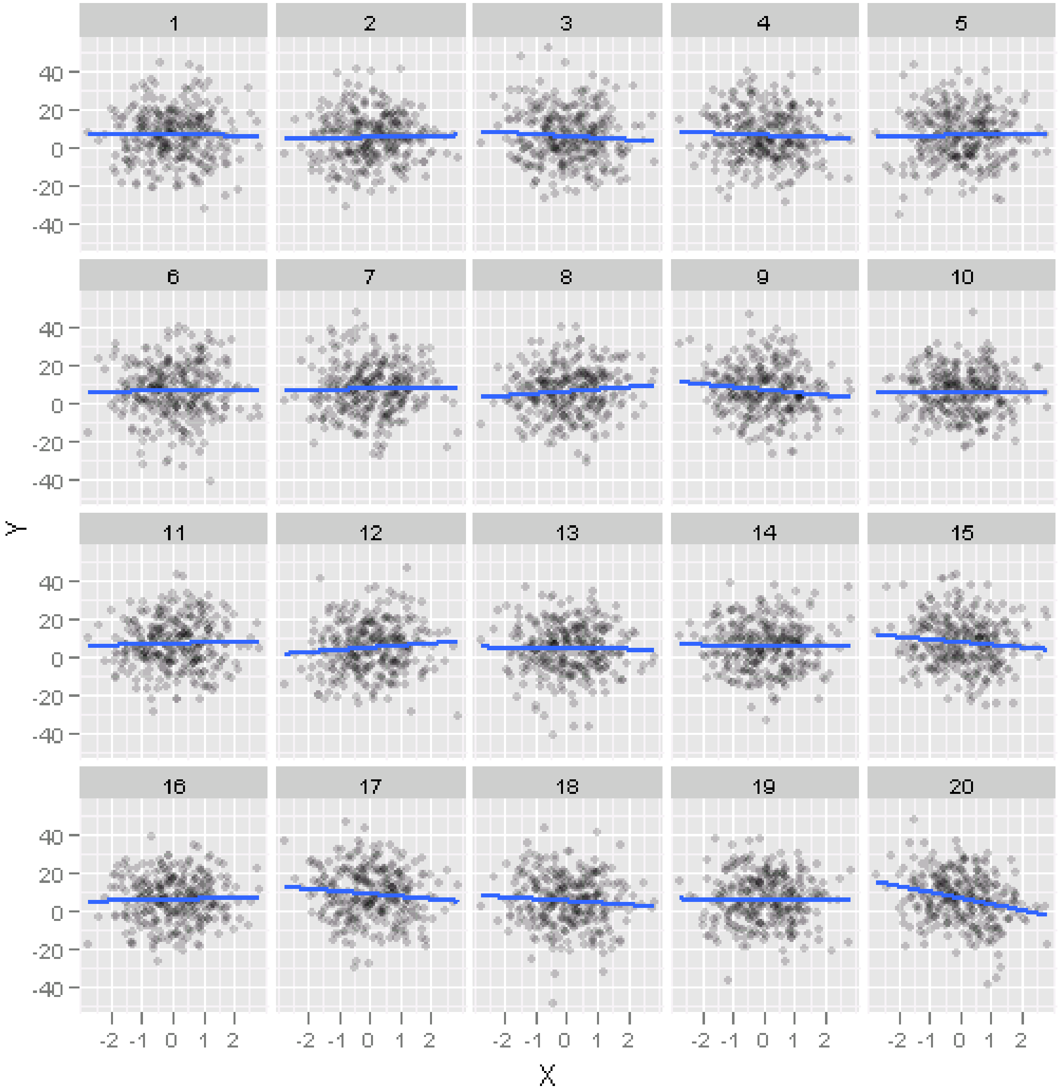

Figure 3.1: *One of 70 lineups used in experiment 2 Majumder et al (2012). Of the 65 people who examined the lineup, 63 selected the data plot, which is in position 20.*

3.1 Classify samples using our model

3.2 Compare accuracy of computer vs human reading

We first try to use the model classify the real plots directly. The accuracy is roughly 42% for the 70 real data plots. We think this result is reasonable since the averaged power of the conventional t-test of these 70 plots is calculated as 0.4567.

Next we will let the model select the highest probability of being linear within each 20 plots, as the same with what human do in the experiment 2.

Chapter 4

Software

- The thesis, code and data is available on the github repository `https://github.com/shuofan18/ETF5550`
- Software used to conduct this research is **R**, **Tensorflow**, **keras**, **tidyverse**

Chapter 5

Timeline

Date	Component
Apr 27	Deep learning model trained
May 4	Classification of new residual plots with model and results summarised
May 18	Comparison with Turk studies
May 24	Refinements made, final summaries written
May 31	Thesis finalised

Bibliography

Anscombe, F (1973). Graphs in Statistical Analysis. *The American Statistician* **27**(1), 17–21.

Cairo, A (2016). “Download the datasaurus: never trust summary statistics alone”. Personal blog.

Chollet, F and J Allaire (2018). Deep Learning with R.

Majumder, M, H Hofmann, and D Cook (2013). Validation of visual statistical inference, applied to linear models. *Journal of the American Statistical Association* **108**(503), 942–956.

Matejka, J and G Fitzmaurice (2017). Same stats, different graphs: generating datasets with varied appearance and identical statistics through simulated annealing. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, pp.1290–1294.

Simchoni, G (2018). “Applying deep learning for the visual inference lineup protocol”. Personal blog.

Tibshirani, R, G James, D Witten, and T Hastie (2013). *An introduction to statistical learning-with applications in R*.

Wickham, H, D Cook, H Hofmann, and A Buja (2010). Graphical inference for infovis. *IEEE Transactions on Visualization and Computer Graphics* **16**(6), 973–979.

Wooldridge, JM (2015). *Introductory econometrics: A modern approach*. Nelson Education.