Experiment Report

Name: Yezhou Wang
Student Number: 515030910614

Part I
Introduction

Who is more likely to gain a large number of citations? Predicting the future influential researchers in big scholarly networks. If we want to know in some certain research field who is most likely to win a prize, or in the future several years, who will publish most papers in a famous journal, we need to analyze the data of a huge number of researchers in the last several years and make a closest rediction.

This project is to predict the most influential researchers in the future. The definition of it is not definite, and many factors can contribute to it, such as paper numbers, paper citations, research field, co-authors and so on.

To simplify and quantificat the problem, we just use the citation numbers to measure the level of influence.

The problem then becomes how to use the data of the last several years to predict the citation numbers, or other data like H-index and G-index in the next few years.

Part II
Dataset

The dataset is crawled by Naixuan Wang. We crawled 1000 researchers' data from *www.xueshu.baidu.com*, after throwing the data which is not useful, there are 987 left, so this is our data set. Time limiting, we can't find another larger data set, but with more time, we can gain much more researchers' data.

Part III
Model Introduction

The model we use to predict the H-index is a regression model. To get the regression function, we use Elastic Net model in machine learning.

Elastic net is a linear regression model using L1 and L2 norm as a priori regularization training. Like Lasso, Elastic net allows study a sparse model that a few parameter is not zero. But it remains some regular properties like Ridge. We can use the l1 ratio parameter to control the convex combination of L1 and L2.

In our project, we use a scholar's current H-index and past H-index to predict the scholar's future H-index. Obviously, all the variables influence the scholar's future H-index. The Lasso model tend to choose one of all the variables to get the regression function, so when the number of variables increases it may face the overfitting problem.

In Elastic net, we add penlty terms to the loss function to deal with overfitting problem while keeping all the variables.

To explain in formula, the loss function of the Ridge regression is:

$$J(\theta) = \frac{1}{2m}[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \lambda \sum_{j=1}^{n} \theta_j^2]$$

In matrix form:

$$J(\beta) = \arg\min_{\beta \in \mathbb{R}} \underbrace{\|y - X\beta\|_2^2}_{Loss} + \lambda \underbrace{\|\beta\|_2^2}_{Penalty}$$

That is loss term add penalty term. The difference between Lasso regression and Ridge regression is the penalty term:

$$J(\beta) = \arg\min_{\beta \in \mathbb{R}} \underbrace{\|y - X\beta\|_2^2}_{Loss} + \lambda \underbrace{\|\beta\|_1}_{Penalty}$$

The Lasso regression uses the absolute value as the penalty term in stead of square value. In Elastic net the loss function is:

$$J(\beta) = \arg\min_{\beta \in \mathbb{R}} + \lambda_2 \|y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1$$

That is the combination of Lasso regression and the Ridge regression.

In this project we use the current h-index and the h-index difference between current year, last year and the year before last to predict the future h-index.

Part IV
Data Handle

After getting all the data we need, We need to handle it for the second time in order to get the data we need. First, because baidu scholar only provide a scholar's current h-index, but we need to know the scholar's past h-index and use them to predict current h-index so that we can check whether our model is

correct. Because we have already get a paper's every year citation number data, so we can use this to get the h-index data. Then we calculate The number of journals a scholar' papers published on. After getting these data, we can begin using our model to predict a scholar's future h-index.

The program code has been uploaded online:
https://github.com/PonyFarmer/YDHLW/blob/master/dataHandle.py


```
1 {"name": "CN-B0734CVJ", "journal_num": [27, 33, 39, 44, 46, 48, 51, 53, 58, 60], "h_index":
  [10, 11, 12, 14, 14, 16, 17, 18, 20, 21], "paper_num": [82, 94, 111, 128, 140, 155, 163, 168,
  189, 199], "citation_sum": [398, 483, 593, 717, 843, 988, 1134, 1272, 1413, 1538],
  "start_year": "1992"}
2 {"name": "CN-B0735M7J", "journal_num": [79, 97, 107, 117, 126, 133, 142, 148, 165, 176],
  "h_index": [18, 20, 22, 24, 27, 29, 33, 36, 38, 40], "paper_num": [172, 210, 261, 291, 318,
  354, 395, 417, 467, 495], "citation_sum": [1455, 1784, 2238, 2753, 3292, 3899, 4590, 5294,
  6010, 6702], "start_year": "1962"}
3 {"name": "CN-B0737QAJ", "journal_num": [45, 46, 51, 55, 59, 60, 62, 67, 69, 72], "h_index":
  [18, 20, 22, 25, 28, 33, 35, 39, 42, 45], "paper_num": [108, 121, 140, 166, 189, 206, 233, 250,
  261, 269], "citation_sum": [1450, 1821, 2280, 2856, 3553, 4308, 5192, 6089, 6996, 7895],
  "start_year": "1997"}
4 {"name": "CN-B0739VSJ", "journal_num": [3, 5, 6, 7, 10, 13, 15, 20, 25, 29], "h_index": [1, 2,
  3, 3, 4, 4, 6, 8, 9, 10], "paper_num": [3, 6, 11, 14, 20, 27, 31, 40, 49, 58], "citation_sum":
  [4, 9, 19, 27, 43, 72, 107, 162, 215, 277], "start_year": "2007"}
5 {"name": "CN-B073R0FJ", "journal_num": [16, 18, 21, 24, 26, 27, 30, 33, 35, 39], "h_index": [7,
  8, 10, 10, 11, 12, 13, 14, 14, 16], "paper_num": [57, 63, 69, 76, 81, 83, 89, 96, 104, 113],
  "citation_sum": [318, 373, 433, 496, 564, 634, 704, 777, 848, 915], "start_year": "1989"}
```


Figure 1: the data after handling

Part V
Model Trainning

We use tensorflow to train the data to get the regression function. The program code hes beeen uploaded online:
https://github.com/PonyFarmer/YDHLW/blob/master/regression.py
In this program, we use the Elastic net above to get the regression function. We choos the top 700 data to train.

```
# 3 对于弹性网络回归算法，损失函数包括L1正则和L2正则
elastic_param1 = tf.constant(1.)
elastic_param2 = tf.constant(1.)
l1_a_loss = tf.reduce_mean(abs(A))
l2_a_loss = tf.reduce_mean(tf.square(A))
e1_term = tf.multiply(elastic_param1, l1_a_loss)
e2_term = tf.multiply(elastic_param2, l2_a_loss)
loss = tf.expand_dims(tf.add(tf.add(tf.reduce_mean(tf.square(y_target - model_output)), e1_term), e2_term), 0)
```

Figure 2: the loss funcion

```
loss_vec = []
for i in range(10000):
    rand_index = np.random.choice(len(x_vals), size=batch_size)
    rand_x = x_vals[rand_index]
    rand_y = np.transpose([y_vals[rand_index]])
    sess.run(train_step, feed_dict={x_data: rand_x, y_target: rand_y})
    temp_loss = sess.run(loss, feed_dict={x_data: rand_x, y_target: rand_y})
    loss_vec.append(temp_loss)
    if (i + 1) % 250 == 0:
        print('Step#' + str(i + 1) + 'A = ' + str(sess.run(A)) + 'b=' + str(sess.run(b)))
        print('Loss= ' + str(temp_loss))
```

Figure 3: train the data for 10000 iterations to get the result

Part VI
Result

The Loss rate change progress is shown is the figure below:

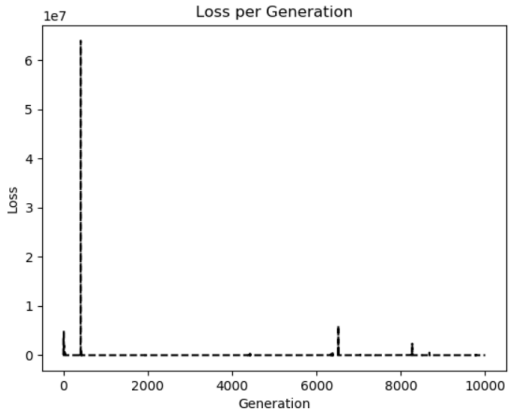

Figure 4: The Loss rate change progress

As it is shown in figure, the loss rate has converged after 10000 iteration. The regression function is:

$$h_1 = 0.038\sqrt{n} + 1.052h + 0.979\Delta h_1 + 0.940\Delta h_2 + -0.016y + 0.0632$$

n: the number of papers the scholar published. h: the current h-index. Δh_1: the difference value between current h-index and last year's index. Δh_2: the difference value between last year's index and the year before last year. y: the publishing year of the scholar's first paper.

Then we use the 190 data left to test the accuracy of the regression function we get. The error rate is show in the graph below.

From this graph we can find that about 150 data of the 190 data is within the -25% 25% error rate.

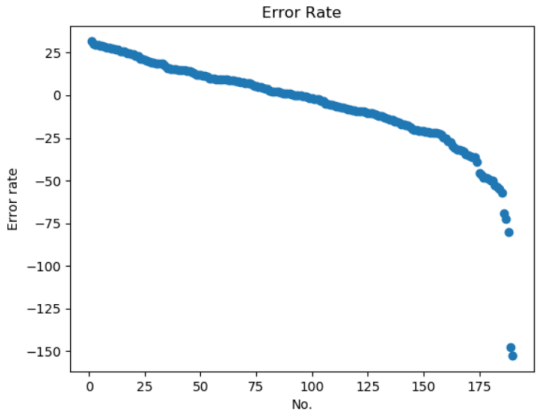

Figure 5: Error rate

Part VII
Conclusion

From the simulation result we can find that our regression model can predict a scholar's furture H-index. Then we can use this data to tell whether a scholar will be inflencial in the future. Most of our data's error rate is between -25% to 25%.

Since the dataset is crawled from the baidu scholar and because we only crawled 1000 data, our training result must be not accurate enough. We first crawled google scholar but we only get half of the data we need when our crawler are banned by google. If we can get more data, we can absolutely improve the result we get.

There are also some more methods to solve this problem. The regression function we get shows a scholar's future h-index is largely depends on the scholar's recent h-index. However, if we think more carefully, we can find that if a non-core paper which has not received citations recently can not do contributions to the h-index. Meanwhile, a core paper keep reciving citations is important for the h-index. If we can find a way to judge the type of a paper, then we can predict the future h-index more precisely.