Binary Classification Problem of Breast Cancer by Logistic Regression and KNN (midterm-project)

Name: WangYexiang
SID: 12012529

***Abstract*—In this project, the Logistic Regression and KNN are used to solve a binary classfication problem of breast cancer. By the researching, the correct method of pre-process of data is proved important and influence the performance of the direction a lot. How to use less data to train a better model with effective algorithm is the direction of future.**

***Keywords*—Dataset, binary classification, Logistic Regression, KNN, sklearn, pre-process, metrics**

I. Introduction

To deal with binary classification problem of breast cancer by logistic regression and KNN, there are two datasets for training models with different data, ”origin_breast_cancer_data.csv” containing 357 benign and 212 malignant samples and ”breast_cancer_data_357B_100M.csv” containing 357 benign and 100 malignant samples, which is unbalanced for positive and negative samples.

For the original dataset ”origin_breast_cancer_data.csv”, use Logistic Regression method and KNN method for binary classification problem (to predict whether it is malignant or benign) with the two datasets mentioned above. Use pre-process the data and tune possible hyper-parameters, to get the best binary classification results for each method. The metrics (recall, precision, and F1 score) will be calculated for both training set and validation set. Compare these two methods and discuss for the conclusions.

For the unbalanced dataset “breast_cancer_data_357B-_100M.csv”, if the performance of the previous ways is degraded, then try some modification for these two methods to see if there is improvement.

In this project, the sklearn is used to help training the prediction models and calculate relative parameters.

II. Problem Formualtion

A. Logistic Regression with original dataset

Firstly, use all data(with 30 dimensions in this project) to train the Logistic Regression model and observe the weights. If there are some weights very small, which means the relative data will not influence the results too much, and even worsen the prediction. What’s more, the standardization and normalization of data will be considered. After testing, adjust the model and find whether the optimization works. For the hyper-parameters, it will be shown by drawing the graph of learning rate.

B. KNN with original dataset

To begin with, use all data(with 30 dimensions in this project) to train the KNN model and use the standardization and normalization to pre-process the data. Then, traverse many values of k to find the best k according to different dataset used to train. By drawing the graph of different k(hyper-parameters here) to indicate the process. Finally, observe the results about different ways to pre-process.

C. Try to deal with unbalanced dataset

After two cases above, the unbalanced dataset will cause that too few malignant samples, whose results is high error rate about the malignant cases. Therefore, when selecting the samples to train the model randomly, try to keep as much as possible malignant cases to promise the necessary training samples. The pre-process methods of standardization and normalization will also be used and observe the results.

III. Method and Algorithms

In this project, the Logistic Regression method and KNN method are used to figure out the binary classification problem. Relative knowledge and algorithms will be introduced briefly below.

A. Logistic Regression

In simple terms, logistic regression is a machine learning method used to solve binary classification (0 or 1) problems to estimate the likelihood of something. For example, the possibility of a user purchasing a certain product, the possibility that a patient has a certain disease, and the possibility of an advertisement being clicked by the user. Note that ”probability” is used here, not mathematical ”probability”, and the result of logisitic regression is not a probability value in the mathematical definition and cannot be used directly as a probability value. This result is often used for weighted summation with other eigenvalues rather than direct multiplication.

- Sigmoid function (or Logistic function)

$$g(z)=\frac{1}{1+e^{-z}} \tag{1}$$

- Cost Function
  In logistic regression, the most commonly used cost function is Cross Entropy:

$$\begin{gathered}J(\theta)=-\frac{1}{m}\left(\sum_{i=1}^{m}\left(y^{(i)}\log h_\theta\left(x^{(i)}\right)+\right.\right.\\ \left.\left.\left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)\right)\right.\end{gathered} \tag{2}$$

- Optimize

$$\min_{\theta} J(\theta) \tag{3}$$

B. KNN (K Nearest Neighbors)

The full name of KNN is K Nearest Neighbors, which means K nearest neighbors. From this name, we can see some clues of the KNN algorithm. K nearest neighbors, there is no doubt that the value of K must be crucial. In fact, the principle of KNN is that when predicting a new value x, it determines which category x belongs to based on what class it is the nearest K points.

- Distance calculation

$$d(x, y) := \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}. \tag{4}$$

- Selection of k
  Start by picking a smaller K value, increase the value of K, then calculate the variance of the validation set, and finally find a suitable K value.

C. Pre-process of data

- Normalization
  In this project, the Min-Max Normalization was used.

$$x^* = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{5}$$

- Standardization
  In this project, Z-score normalization (standard deviation normalization / zero mean normalization) was used.

$$x^* = \frac{x - \bar{x}}{\sigma} \tag{6}$$

D. sklearn

SciKit-Learn, also known as sklearn, is an open source machine learning toolkit based on the Python language. It enables efficient algorithm applications through python numerical computing libraries such as NumPy, SciPy and Matplotlib, and covers almost all mainstream machine learning algorithms. In this project, the sklearn.linear_model, sklearn.neighbors, sklearn.metrics, sklearn.preprocessing and so on will be used.

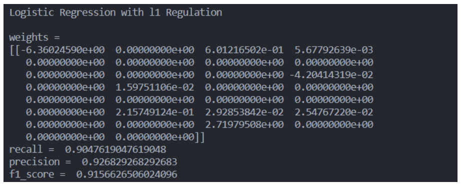
IV. Experiment results and analysis

A. Logistic Regression model trained by original dataset

First of all, all data are used to train the model, and the parameters of Logistic Regression model is used with l1 regulation. It is obvious that many parameters are very small, which means they even don't influence the prediction of the model at all. the result is shown below as Figure 1:

In other side, the l2 regulation, Normalization and Standardization are uesed beacuse of the difference of the 30 sets of data. The effects are better, results shown as Figure 2, Figure 3 and Figure 4:

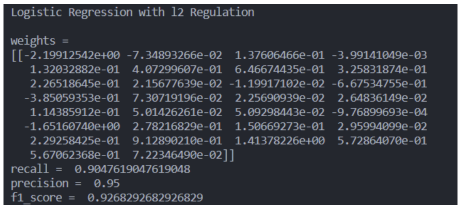
By selecting the effective data to train the model, the result is getting better according to the case using all data, shown as Figure 5: It is obvious that after selecting the proper data, the effect of model is better according to the higher f1_score.



```
Logistic Regression with l1 Regulation

weights =
[[-6.36024590e+00  0.00000000e+00  6.01216502e-01  5.67792639e-03
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00 -4.20414319e-02
   0.00000000e+00  1.59751106e-02  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  2.15749124e-01  2.92853842e-02  2.54767220e-02
   0.00000000e+00  0.00000000e+00  2.71979508e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00]]
recall =  0.9047619047619048
precision =  0.926829268292683
f1_score =  0.9156626506024096
```

Fig. 1. Figure1:all data with regulation l1 of LR



```
Logistic Regression with l2 Regulation

weights =
[[-2.19912542e+00 -7.34893266e-02  1.37606466e-01 -3.99141049e-03
   1.32032882e-01  4.07299607e-01  6.46674435e-01  3.25831874e-01
   2.26518645e-01  2.15677639e-02 -1.19917102e-02 -6.67534755e-01
  -3.85059353e-01  7.30719196e-02  2.25690939e-02  2.64836149e-02
   1.14385912e-01  5.01426261e-02  5.09298443e-02 -9.76899693e-04
  -1.65160740e+00  2.78216829e-01  1.50669273e-01  2.95994099e-02
   2.29258425e-01  9.12890210e-01  1.41378226e+00  5.72864070e-01
   5.67062368e-01  7.22346490e-02]]
recall =  0.9047619047619048
precision =  0.95
f1_score =  0.9268292682926829
```

Fig. 2. Figure2:all data with regulation l2 of LR

B. KNN model trained by original dataset

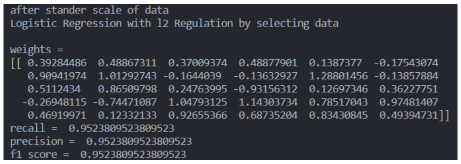
To begin with, the all data without pre-process was used to train the KNN model, whose score graph of different k was shown as Figure 6. After finding the best k, the result of this best k was shown as Figure 7.

Then the Normalization and Standardization are uesed. For Standardization, the score graph of different k was shown as Figure 8. After finding the best k, the result of this best k was shown as Figure 9.

For Normalization, the score graph of different k was shown as Figure 10. After finding the best k, the result of this best k was shown as Figure 11.

C. Logistic Regression model trained by unbalanced dataset

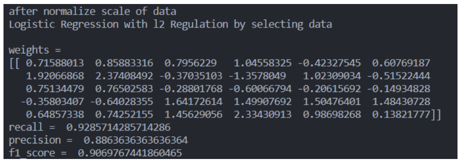
First of all, all data are used to train the model, and the parameters of Logistic Regression model is used with l1 regulation. It is obvious that many parameters are very small, which means they even don't influence the prediction of the model at all. the result is shown below as Figure 12: In other side, the l2 regulation, Normalization and Standardization are



```
after stander scale of data
Logistic Regression with l2 Regulation by selecting data

weights =
[[ 0.39284486  0.48867311  0.37009374  0.48877901  0.1387377  -0.17543074
   0.90941974  1.01292743 -0.1644039  -0.13632927  1.28801456 -0.13857884
   0.5112434   0.86509798  0.24763995 -0.93156312  0.12697346  0.36227751
  -0.26948115 -0.74471087  1.04793125  1.14303734  0.78517043  0.97481407
   0.46919971  0.12332133  0.92655366  0.68735204  0.83430845  0.49394731]]
recall =  0.9523809523809523
precision =  0.9523809523809523
f1 score =  0.9523809523809523
```

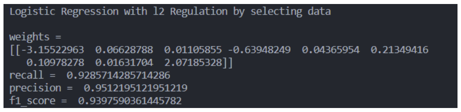
Fig. 3. Figure3:all data with standardization of LR



```
after normalize scale of data
Logistic Regression with l2 Regulation by selecting data

weights =
[[ 0.71588013  0.85883316  0.7956229   1.04558325 -0.42327545  0.60769187
   1.92066868  2.37408492 -0.37035103 -1.3578049   1.02309034 -0.51522444
   0.75134479  0.76502583 -0.28801768 -0.60066794 -0.20615692 -0.14934828
  -0.35803407 -0.64028355  1.64172614  1.49907692  1.50476401  1.48430728
   0.64857338  0.74252155  1.45629056  2.33430913  0.98698268  0.13821777]]
recall =  0.9285714285714286
precision =  0.8863636363636364
f1_score =  0.9069767441860465
```

Fig. 4. Figure4:all data with normalization scale of LR



```
Logistic Regression with l2 Regulation by selecting data

weights =
[[-3.15522963  0.06628788  0.01105855 -0.63948249  0.04365954  0.21349416
   0.10978278  0.01631704  2.07185328]]
recall =  0.9285714285714286
precision =  0.9512195121951219
f1_score =  0.9397590361445782
```

Fig. 5. Figure5:after selecting data of LR

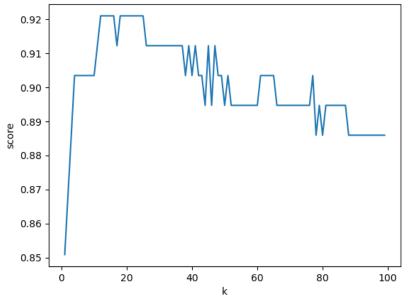
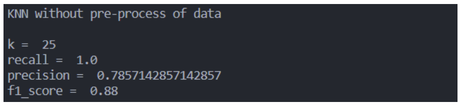


Fig. 6. Figure6:find best k without pre-process of KNN



```
KNN without pre-process of data

k =  25
recall =  1.0
precision =  0.7857142857142857
f1_score =  0.88
```

Fig. 7. Figure7:result without pre-process of KNN

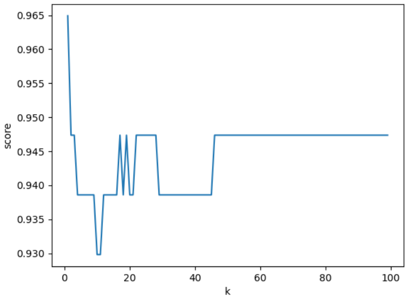
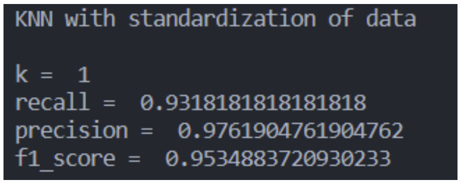


Fig. 8. Figure8:find best k with Standardization of KNN



```
KNN with standardization of data

k =  1
recall =  0.9318181818181818
precision =  0.9761904761904762
f1_score =  0.9534883720930233
```

Fig. 9. Figure9:result with Standardization of KNN

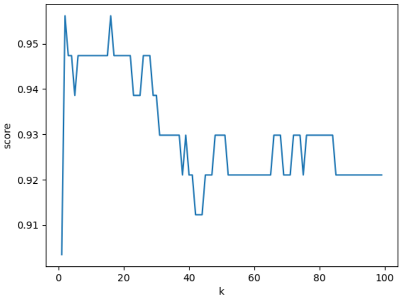
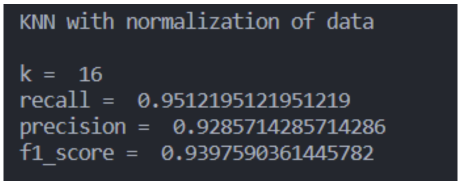


Fig. 10. Figure10:find best k with Normalization of KNN



```
KNN with normalization of data

k =  16
recall =  0.9512195121951219
precision =  0.9285714285714286
f1_score =  0.9397590361445782
```

Fig. 11. Figure11:result with Normalization of KNN

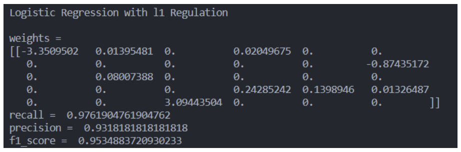
uesed beacuse of the difference of the 30 sets of data. However, in this time, the effects are uncertain. Some of them even become worse. The reason inferred is unbalanced data and improper pre-process operation of dataset. The results shown as Figure 13, Figure 14 and Figure 15:

By selecting the effective data to train the model, the result is also uncertain according to the case using all data. Some even get wore a lot, shown as Figure 16:

D. *KNN model trained by unbalanced dataset*

To begin with, the all data without pre-process was used to train the KNN model, whose score graph of different k was shown as Figure 17. After finding the best k, the result of this best k was shown as Figure 18.

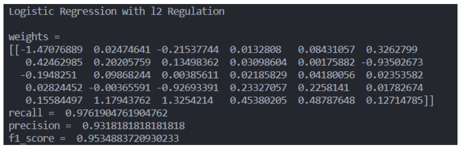
Then the Normalization and Standardization are uesed. For Standardization, the score graph of different k was shown as



```
Logistic Regression with l1 Regulation

weights =
[[-3.3509502   0.01395481  0.          0.02049675  0.          0.
   0.          0.          0.          0.          0.         -0.87435172
   0.          0.08007388  0.          0.          0.          0.
   0.          0.          0.          0.24285242  0.1398946   0.01326487
   0.          0.          3.09443504  0.          0.          0.        ]]
recall =  0.9761904761904762
precision =  0.9318181818181818
f1_score =  0.9534883720930233
```

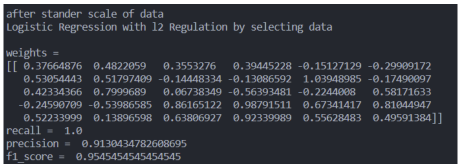
Fig. 12. Figure12:all data with regulation l1 of LR



```
Logistic Regression with l2 Regulation

weights =
[[-1.47076889  0.02474641 -0.21537744  0.0132808   0.08431057  0.3262799
   0.42462985  0.20205759  0.13498362  0.03098604  0.00175882 -0.93502673
  -0.1948251   0.09868244  0.00385611  0.02185829  0.04180056  0.02353582
   0.02824452 -0.00365591 -0.92693391  0.23327057  0.2258141   0.01782674
   0.15584497  1.17943762  1.3254214   0.45380205  0.48787648  0.12714785]]
recall =  0.9761904761904762
precision =  0.9318181818181818
f1_score =  0.9534883720930233
```

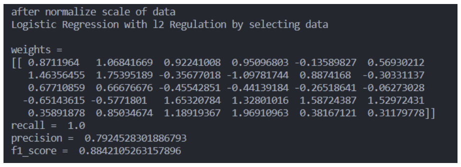
Fig. 13. Figure13:all data with regulation l2 of LR



```
after stander scale of data
Logistic Regression with l2 Regulation by selecting data

weights =
[[ 0.37664876  0.4822059   0.3553276   0.39445228 -0.15127129 -0.29909172
   0.53054443  0.51797409 -0.14448334 -0.13086592  1.03948985 -0.17490097
   0.42334366  0.7999689   0.06738349 -0.56393481 -0.2244008   0.58171633
  -0.24590709 -0.53986585  0.86165122  0.98791511  0.67341417  0.81044947
   0.52233999  0.13896598  0.63806927  0.92339989  0.55628483  0.49591384]]
recall =  1.0
precision =  0.9130434782608695
f1_score =  0.9545454545454545
```

Fig. 14. Figure14:all data with standardization of LR



```
after normalize scale of data
Logistic Regression with l2 Regulation by selecting data

weights =
[[ 0.8711964   1.06841669  0.92241008  0.95096803 -0.13589827  0.56930212
   1.46356455  1.75395189 -0.35677018 -1.09781744  0.8874168  -0.30331137
   0.67710859  0.66676676 -0.45542851 -0.44139184 -0.26518641 -0.06273028
  -0.65143615 -0.5771801   1.65320784  1.32801016  1.58724387  1.52972431
   0.35891878  0.85034674  1.18919367  1.96910963  0.38167121  0.31179778]]
recall =  1.0
precision =  0.7924528301886793
f1_score =  0.8842105263157896
```

Fig. 15. Figure15:all data with normalization scale of LR

Figure 19. After finding the best k, the result of this best k was shown as Figure 20.

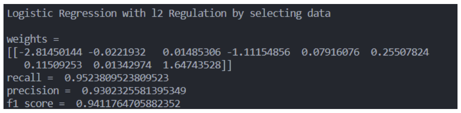
For Normalization, the score graph of different k was shown as Figure 21. After finding the best k, the result of this best k was shown as Figure 22.

It is clear that the pre-process of data which used Normalization and Standardization is effective and helpful for KNN model in the condition unbalanced data, which means reduce the influence caused by one-result samples

V. Conclusion and Future Problems

For the machine learning, the pre-process of data is very important. How to select proper data to train the model decide the result of the model.

In this project, the pre-process operation of data almost implement the f1_score of the model, which means the rate of correct prediction is lifted. For instance, as KNN model, the Normalization and Standardization are very notable. However,



```
Logistic Regression with l2 Regulation by selecting data

weights =
[[-2.81450144 -0.0221932   0.01485306 -1.11154856  0.07916076  0.25507824
   0.11509253  0.01342974  1.64743528]]
recall =  0.9523809523809523
precision =  0.9302325581395349
f1_score =  0.9411764705882352
```

Fig. 16. Figure16:after selecting data of LR

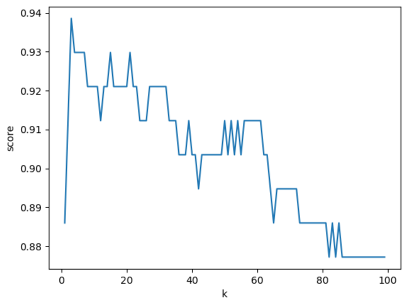
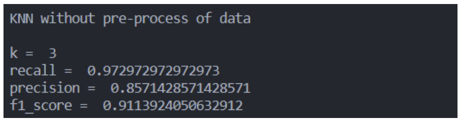


Fig. 17. Figure17:find best k with Normalization of KNN



```
KNN without pre-process of data

k =  3
recall =  0.972972972972973
precision =  0.8571428571428571
f1_score =  0.9113924050632912
```

Fig. 18. Figure18:result with Normalization of KNN

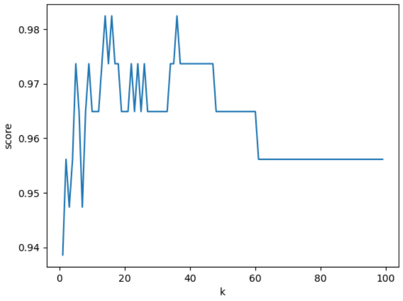
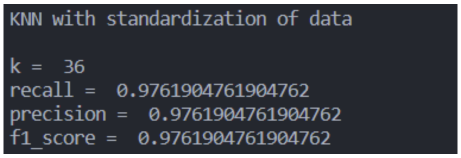


Fig. 19. Figure19:find best k with Normalization of KNN



```
KNN with standardization of data

k =  36
recall =  0.9761904761904762
precision =  0.9761904761904762
f1_score =  0.9761904761904762
```

Fig. 20. Figure20:result with Normalization of KNN

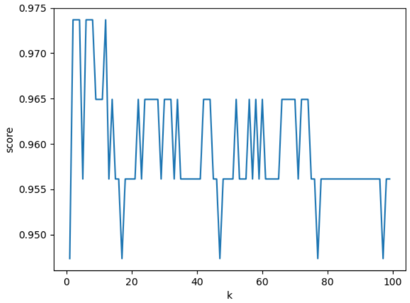
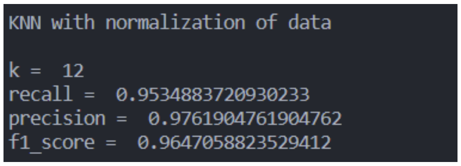


Fig. 21. Figure10:find best k with Normalization of KNN



```
KNN with normalization of data

k =  12
recall =  0.9534883720930233
precision =  0.9761904761904762
f1_score =  0.9647058823529412
```

Fig. 22. Figure11:result with Normalization of KNN

some times it don’t perform very well, like the Logistic Regression in this project.

About the future problems, when the number of data is small, just like this project, how to use the more effective algorithms to reduce the dependence of dataset, which means the model can find effective features and laws is a direction deserving consideration.

References

[1] Python—KNN https://zhuanlan.zhihu.com/p/143092725

[2] http://t.csdn.cn/WuV4W

[3] http://t.csdn.cn/dTS0C

[4] https://www.knowledgedict.com/tutorial/sklearn-lr.html

[5] https://zhuanlan.zhihu.com/p/462192060

[6] https://developer.aliyun.com/article/692322: :text=Python

[7] http://t.csdn.cn/ANXzE

.