INF552: Programming Assignment 4

Student: Jiashi Chen USCID: 4684194123

Part 1: Implementation

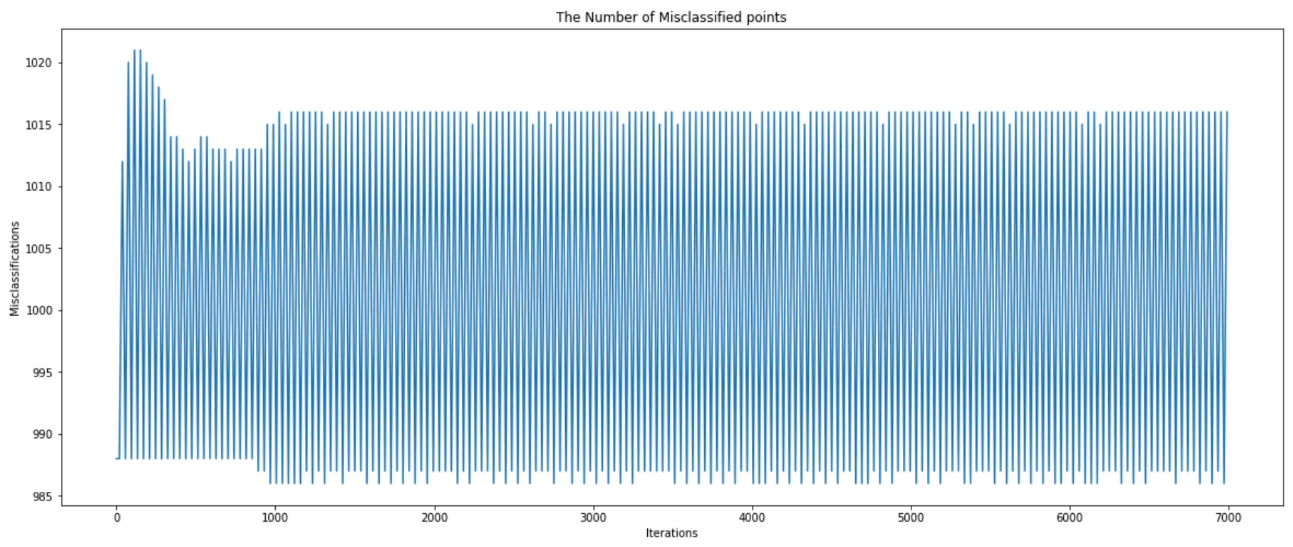
The code is in ***Jiashi_Chen_Homework_4_Code.ipynb*** file.

Perceptron Learning algorithm

Weight: [0.006, 2.46411518, -1.98308819, -1.47998528] *Accuracy*: 100%

Pocket algorithm

Weight: [0.049, -1.76521756, -0.03557024, -0.13354639] *Accuracy*: 50.7%



We can see that the number of misclassified points keep changing back and forth. What we can learn from the plot is that the dataset is not linear separable. After 7000 iterations, we still can't get a good result.

Logistic Regression

Weight: [-0.03150075, -0.17769619, 0.11445235, 0.07670126] *Accuracy*: 52.95%

Linear Regression

Weight: [0.01523535, 1.08546357, 3.99068855]

The **data structure** I use is *matrix* from numpy module and *dataframe* from pandas module. Dataframe could help me store the orginal data. I use the matrix to store the coordinates of a point, classification label and weight, which makes it easier for me to do matrix operations.

Any code-level **optimizations** I perform is that I try to use some vector and matrix operations to get the values at once. This will be much faster than calculating the value one by one for all 2000 points.

The **challenges** I face is that how to make the code work more efficiently. At first, I use for loop to calculate the value, which is expensive. Now I use matrix operations to solve this problem. I pay a lot of efforts to find out the difference between the operation of array in numpy and the operation of matrix in pandas when it comes to the same operator.

Part 2: Software Familiarization

Sklearn package offers a good implementation of linear classification, linear regression, and logistic regression.

How to use

I used this package to fit the data in ***Jiashi_Chen_Homework_4_Code.ipynb*** file too.

Linear classification

API for linear classification(https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html#sklearn.linear_model.Perceptron)

```
from sklearn.linear_model import Perceptron  ## Obtain necessary package

classifier = Perceptron()  #Get classifier
classifier.fit(X, y)  #Fit linear model with Stochastic Gradient Descent.
```

Linear regression

API for linear regression(hhttps://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html#sklearn.linear_model.LinearRegression)

```
from sklearn.linear_model import LinearRegression  ## Obtain necessary package

classifier = LinearRegression()  #Get classifier
classifier.fit(X, y)  #Fit linear model
```

Logistic regression

API for logistic regression(https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression)

```
from sklearn.linear_model import LogisticRegression  ## Obtain necessary package

classifier = LogisticRegression()  #Get classifier
classifier.fit(X, y)  #Fit the model according to the given training data.
```

Comparsion and Improvement

After I read the API document of Sklearn, I find it add many methods to do feature engineering. There is a parameter called penalty in Sklearn. We could choose 'l2','l1','elasticnet' as input value. The default value of it in logistic regression is l2 and the default value of it in perceptron is None. In the process of regression, regularization is very useful. It efficiently solve the overfitting problem. So I could also consider this in my program to improve the accuracy of the algorithm.

Part 3: Applications

Linear classification in Text Categorization[1]

A number of linear classification methods such as linear least squares fit (LLSF), logistic regression, and support vector machines (SVMs) have been applied to text categorization problems. These methods find hyperplanes that approximately separate a class of document vectors from its complement. In this paper, authors compare a number of known linear classification methods as well as some variants in the framework of regularized linear systems. They discuss the statistical and numerical properties of these algorithms, with a focus on text categorization. They also provide some numerical experiments to illustrate these algorithms on a number of datasets

Linear regression in solving spatial autocorrelation problem[2]

The Moran Coefficient spatial autocorrelation index can be decomposed into orthogonal map pattern components. This decomposition relates it directly to standard linear regression, in which corresponding eigenvectors can be used as predictors. This paper reports comparative results between these linear regressions and their auto-Gaussian counterparts for the following georeferenced data sets: Columbus (Ohio) crime, Ottawa-Hull median family income, Toronto population density, southwest Ohio unemployment, Syracuse pediatric lead poisoning, and Glasgow standard mortality rates, and a small remotely sensed image of the High Peak district. This methodology is extended to auto-logistic and auto-Poisson situations, with selected data analyses including percentage of urban population across Puerto Rico, and the frequency of SIDs cases across North Carolina. These data analytic results suggest that this approach to georeferenced data analysis offers considerable promise.

Logistic regression in survival analysis[3]

In this paper, we could learn that standard logistic regression technique can be used to estimate hazard rates and survival curves from censored data. These techniques allow the statistician to use parametric regression modeling on censored data in a flexible way that provides both estimates and standard errors. An example is given that demonstrates the increased structure that can be seen in a parametric analysis, as compared with the nonparametric Kaplan-Meier survival curves. In fact, the logistic regression estimates are closely related to Kaplan-Meier curves, and approach the Kaplan-Meier estimate as the number of parameters grows large.

References

[1]Zhang T , Oles F J . Text Categorization Based on Regularized Linear Classification Methods[J]. Information Retrieval, 2001, 4(1):5-31.

[2] Griffith D A . A linear regression solution to the spatial autocorrelation problem[J]. Journal of Geographical Systems, 2000, 2(2):141-156.

[3] Bradley, Efron. Logistic Regression, Survival Analysis, and the Kaplan-Meier Curve[J].