Labs
Machine Learning Course
Fall 2016

EPFL
School of Computer and Communication Sciences
Martin Jaggi & Rüdiger Urbanke
mlo.epfl.ch/page-136795.html
epfmlcourse@gmail.com

Problem Set 5, Oct. 20, 2016 (Logistic Regression)

Goals. The goal of this exercise is to

- Do classification using linear regression.
- Implement and debug logistic regression using gradient descent.
- Compare it to linear regression.
- Implement Newton's method for logistic regression.

Setup, data and sample code. Obtain the folder `labs/ex05` of the course github repository

github.com/epfml/ML_course

We will use the dataset "height_weight_genders.csv" in this exercise, and we have provided sample code templates that already contain useful snippets of code required for this exercise.

1 Classification Using Linear Regression

We will try to use linear regression to do classification. Although this is not a good idea in general (as discussed in the class), it will work for simple data. We will use the height-weight data from the first exercise. For better visualization, we will randomly sub-select 200 data point from the data.

Exercise 1:

Classification using linear regression.

- Use least squares finished in previous exercise to compute a $\boldsymbol{w}$. Please COPY your previous implementation to the template `least_squares.py`.
- Visualize the data points and the decision boundary as Figure 1.

2 Logistic Regression

Exercise 2:

Implement logistic regression using gradient descent.

- Fill in the notebook function `sigmoid()`.
- Fill in two notebook functions `calculate_loss()` and `calculate_gradient()`. The first function should return negative of the value of log-likelihood, while the second function should return the corresponding gradient.
- Implement the gradient descent `learning_by_gradient_descent()` for logistic regression. You should calculate loss, gradient and update the weight $\boldsymbol{w}$, and the function should return loss and updated weight.

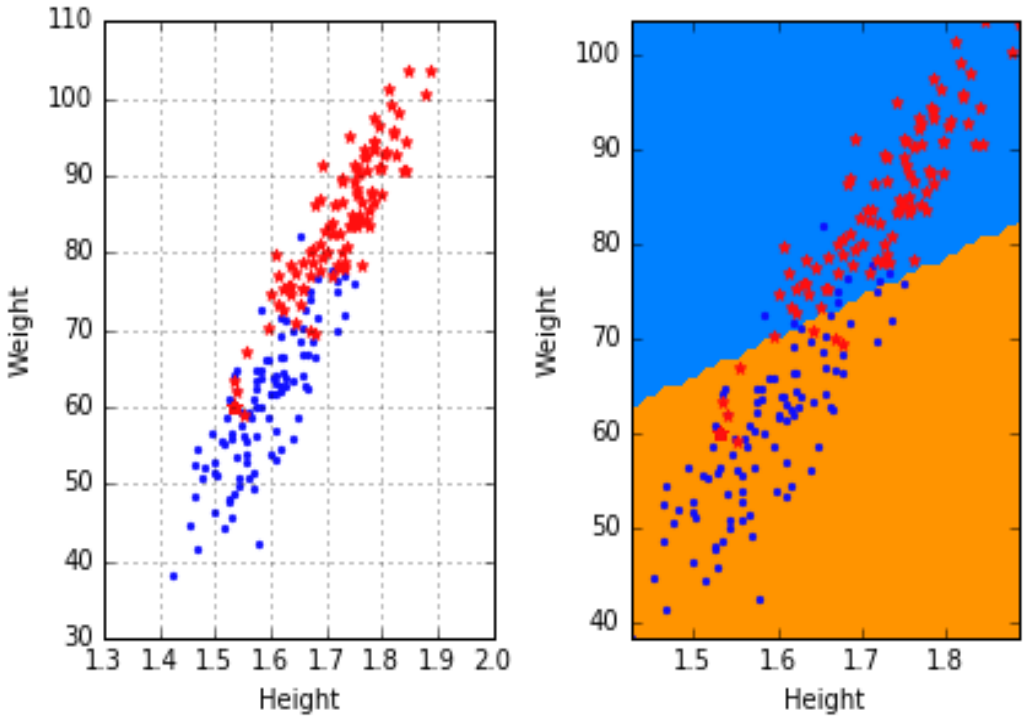

Figure 1: classification by least square.

- Plot predictions to get a visualization similar to the right one of Figure 1. Check if you get similar or different results.
- Do your results make sense? ***Yes, it's probably the best fit we can obtain with a linear regression. (a little below the linear_reg function, but same slope.***

Once you have gradient descent, it is also straightforward to implement Newton's method.

Exercise 3:

Newton's method for logistic regression.

- Fill in the notebook function `calculate_hessian()`. And then integrate your implementation, i.e., `calculate_loss()`, `calculate_gradient()` and `calculate_hessian()`, to the function `logistic_regression()` for future usage. The function should return the cost function, gradient, and Hessian altogether.
- Your gradient descent code can now be turned into a Newton's method algorithm. Please fill the notebook function `learning_by_newton_method()`. The function should return your loss and updated weight.

Exercise 4:

Penalized logistic regression.

- Fill in the notebook function `penalized_logistic_regression()`. Note that it can be done by adding the regularization term $\lambda \sum_{d=0}^{D} \boldsymbol{w}_d^2$. Set λ to a low value and check if it gives the same result. Once it is done, please fill in the notebook function `learning_by_penalized_gradient()`, and increase the value of λ and check whether $\boldsymbol{w}$ is shrinking or not.
- Check if this gives the same answer as gradient descent. To debug, print the function value and the norm of the gradient in every iteration. All of these values should decrease in every iteration.

Exercise 5:

BONUS: Implement IRLS. Follow the pseudo-code given in the leture notes.