Artificial Intelligence

COMS 4701 Section 2 – Fall 2014

http://www1.ccls.columbia.edu/~ansaf/4701/

Home Work n°3: Machine Learning

Due Wednesday November 19th, 2014 @11:55pm

Preliminaries

---

1. We recommend you follow the order in which the questions in each problem are posed.
2. Mandatory: Please comment **all important lines** in your code (except the obvious lines).
3. Please use Python in solving all problems and refrain from using code from the web.
4. You can use Python packages to solve the problems, including **numpy** and **matplotlib**, but **do not use scikit-learn** nor another ready to use implementation of linear regression in Problem 1 including **stats.linregress**. You can use **scikit-learn** in Problem 2.
5. Your submission package must be a zipped file including: the PDF containing all your responses, and figures as requested in png or pdf form and a separate code for each problem.
6. Name your submission as follows: firstname_lastname_uni.zip and submit on CourseWorks.

Problem 1: Regression

---

1 Linear regression with one feature

We are interested in studying the relationship between age and height (statures) in girls aged 2 to 8 years old. We think that this can be modeled with a linear regression model. We have examples (data points) for a population of 60 girls. Each example has one feature *Age* along with a numerical label *Height*. We will use the dataset **girls_train.csv** (derived from CDC growthchart data[1]). Your mission is to implement linear regression with one feature using gradient descent. You will plot the data, regression line, and cost function. You will finally make a prediction for a new example using your regression model and assess your out of sample mean square error.

1.1 Load & Plot

a) Load the dataset **girls.csv**.

b) Plot the distribution of the data.

1.2 Gradient descent

Now that your data is loaded and you know how it looks like, find a regression model of the form:

$$\text{Height} = \beta_0 + \beta_1 \times \text{Age}$$

a) Implement Gradient Descent to find the β's of the model. Remember you need to add the vector 1 ahead of your data matrix. Also, make sure you update the parameters β's *simultaneously*. Use a learning rate alpha $= 0.05$ and #iterations $= 1500$.

b) What is the mean square error of your regression model on the training set?

[1] http://www.cdc.gov/growthcharts/

1.3 Plot the regression line, and bowl function

Once you obtain your parameters:

a) Plot the regression line on top of your distribution.

c) Plot the cost function that has the bowl shape as we saw in class (may be a flat bowl though...)

1.4 Testing your model and Making a prediction for a new example

a) It's time now to try your model and make a prediction. Using your model, make a prediction for a 4.5 years old girl. What would be her predicted height?

b) Given a dataset of 20 girls that was kept on the side as a test set **girls_test.csv**, compute the mean square error for the test set and make a comparison to the training mean square error.

2 Linear regression with multiple features

In this problem, you will work on linear regression with multiple features using gradient descent and the normal equation. You will also study the relationship between the risk function, the convergence of gradient descent, and the learning rate. We will use the dataset **girls_age_weight_height_2_8.csv** (derived from CDC growthchart data).

2.1 Data Preparation & Normalization

Once you load your dataset, explore the content to identify each feature. Remember to add the vector 1 (intercept) ahead of your data matrix.

a) You will notice that the features are not on the same scale. They represent age (years), and weight (kilograms). Print the mean and standard deviation of each feature in your data. The last column is the label and represent the height (meters).

b) Scale each feature by its standard deviations and set its mean to zero. You do not need to scale the intercept. For the each feature x (a column in your data matrix), use the following formula:

$$x_{\text{scaled}} = \frac{x - \mu(x)}{stdev(x)}$$

2.2 Gradient Descent

As you did in the previous section, implement gradient descent but this time with two features. Initialize your β's to zero. We recall the empirical risk and gradient descent rule as follows:

$$R(\beta) = \frac{1}{2n}\sum_{i=0}^{n}(f(x_i) - y_i)^2$$

$$\forall j \qquad \beta_j = \beta_j - \alpha\frac{1}{n}\sum_{i=0}^{n}(f(x_i) - y_i)x_i$$

2.3 Plotting Risk function for different learning rates

a) We need to pick an appropriate learning rate $\alpha \in \{0.005, 0.001, 0.05, 0.1, 0.5, 1\}$. Run gradient descent and plot the Risk function with respect to the number of iterations for different values of α. Use the same figure to plot all curves. Choose #iterations=50.

b) Compare the convergence rate when α is small versus large.

c) Which α is best? Use this α to run gradient descent and print the β's.

d) Using the β vector, make a height prediction for a 5-year old girl weighting 20 kilos (don't forget to scale!).

2.4 Normal equation

Recall the second approach for linear regression using the Normal Equation. Its implementation is straightforward to find the β's. There is no need to scale the features this time.

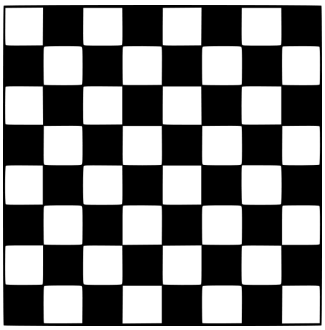
$$\beta = (X^t X)^{-1} X^T y$$

a) Compare the β vector you obtained with gradient descent to the one calculated with normal equation. Are they the same? Why?

b) Using the β vector, make a height prediction for a 5-year old girl weighting 20 kilos (don't forget to scale!).

c) Do gradient descent and Normal Equation lead to the same height prediction?

Problem 2: Classification with Support Vector Machines & more...



We will use Sklearn to learn a classification model for a chessboard-like dataset.

1. Open the dataset chessboard.csv in python.
2. Plot a scatter plot of the dataset showing the two classes with two different patterns.
3. We would like to use Support Vector Machines with different kernels to build a classifier. Make sure you split your data into training (60%) and testing (40%). Make sure you use a stratified sampling (same ratio positive to negative in the training and testing datasets). Use **cross validation** this time instead of a validation set (train-test splitting and cross validation are both functionalities are readily available in Sklearn).
   (a) Show the performance of the SVM for linear, polynomial and RBF kernels with different settings of the polynomial order, sigma and C value. Try to hunt for a good setting of these parameters to obtain high classification accuracy. Report all your results on crossvalidation and test data.
   (b) Plot the decision boundary for the best parameters for each kernel.
4. (Optional) Try other linear or non-linear classifiers on this dataset. You can pick one or many to try (e.g., logistic regression, random forests, multiple layer perceptron, etc.). Report the results and draw the decision boundary.