Machine Learning & Data Mining
Homework 2

Caltech CS/CNS/EE 155
January 12^{th}, 2017

This set is due 9pm, January 20^{th}, via Moodle. You are free to collaborate on all of the problems, subject to the collaboration policy stated in the syllabus. Please include any code with your submission.

1 Comparing Different Loss Functions

TAs: Jagriti Agrawal, Siddharth Murching
Relevant materials: Lecture 3

We've discussed three loss functions for linear models so far:

- Squared loss: $L_{\text{squared}} = (1 - y\mathbf{w}^T\mathbf{x})^2$
- Hinge loss: $L_{\text{hinge}} = \max(0, 1 - y\mathbf{w}^T\mathbf{x})$
- Log loss: $L_{\log} = \log(1 + e^{-y\mathbf{w}^T\mathbf{x}})$

where $\mathbf{w} \in \mathbb{R}^n$ are the model parameters, $y \in \{-1, 1\}$ is the class label for datapoint $\mathbf{x} \in \mathbb{R}^n$, and we're including a bias term in $\mathbf{x}$ and $\mathbf{w}$. The model classifies points according to $\text{sign}(\mathbf{w}^T\mathbf{x})$.

Performing gradient descent on any of these loss functions will train a model to classify more points correctly, but the choice of loss function has a significant impact on the model that is learned.

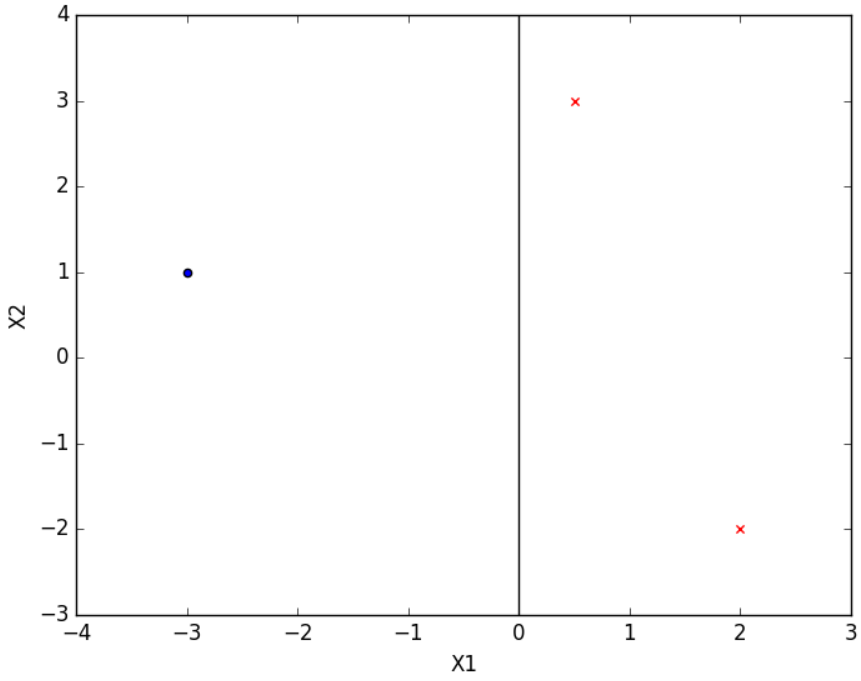
Question A: Squared loss is almost always a terrible choice of loss function to train on for classification problems. Why?

Question B: Leaving squared loss behind, let's focus on log loss and hinge loss.

Consider the set of points $S = \{(\frac{1}{2}, 3), (2, -2), (-3, 1)\}$ in 2D space, shown below, with labels $(1, 1, -1)$ respectively.

Given a linear model with weights $w_0 = 0, w_1 = 1, w_2 = 0$ (where w_0 corresponds to the bias term), compute the gradients $\nabla_w L_{\text{hinge}}$ and $\nabla_w L_{\log}$ of the hinge loss and log loss, and calculate their values for each point in S.

Compare the gradients resulting from log loss to those resulting from hinge loss. When (if ever) will these gradients converge to 0? Based on this answer, explain why for an SVM to be a "maximum margin" classifier, its learning objective must not be just to minimize L_{hinge}, but to minimize $L_{\text{hinge}} + \lambda\|w\|^2$ for some $\lambda > 0$.

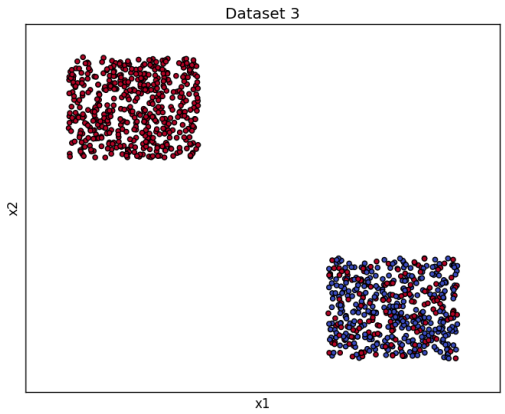


The example dataset and decision boundary described above. Positive instances are represented by red x's, while negative instances appear as blue dots.

Question C:

There are three 2-D datasets on the course website: problem1dataset1.txt, problem1dataset2, and problem1dataset3.txt. The first two columns in each represent x_1, x_2, and the last column represents the label, $y \in \{-1, +1\}$.

Each of the three datasets consists of two clusters of points, one of points primarily with positive labels, the other primarily of points with negative labels. The datasets vary as follows: dataset1 is noiseless and has no "outliers", meaning there are no points labeled +1 in the cluster of negative-labeled points, dataset2 has about 5 percent of the points as outliers, and dataset3 has about 15 percent of the points as outliers.



Plot of dataset3: points with positive labels in red, those with negative labels in blue

On each of the three datasets, train both a logistic regression model and an SVM model to classify the points. (In other words, on each dataset, train one linear classifier using L_{log} as the loss, and another linear classifier using L_{hinge} as the loss.) For this problem, you should use the logistic regression and SVM (a.k.a. SVC) implementations provided within scikit-learn (logistic regression documentation) (SVM documentation) instead of your own implementations. Leave all the parameters for these classifiers at their default values, *except change the kernel of the SVM to be linear*. I.e. include `kernel='linear'` when you instantiate your SVC class.

For each of the six combinations of dataset and loss function, plot the data points as a scatter plot and overlay them with the decision boundary defined by the weights of the trained linear classifier. Include your six plots in your submission. See the plot_decision_boundary.py file on the course website, which contains a helper function for producing plots given a trained classifier.

What differences do you see in the decision boundaries learned using the different loss functions? Explain them based on what you know about the proportion of outliers in each cluster and the gradients for SVM and logistic regression.

Question D: Now consider the case in which false negatives are significantly worse than false positives. This could occur, for example, when fitting a classifier to diagnose a disease.

Fit both logistic regression and SVM (with a linear kernel) to dataset2, but weight positive-labeled instances 5x more than negative-labeled instances in the loss calculation. (Hint: use the class_weight parameter.) For each fitted model, create a plot showing the data points and the model's decision boundary. You should end up producing two plots. How does this class-weighting scheme affect the models' decision boundaries?

2 Effects of Regularization

TAs: Jagriti Agrawal, Siddharth Murching
Relevant materials: Lecture 4

For this problem, you are required to implement everything yourself and submit code.

Question A : In order to prevent over-fitting in the least-squares linear regression problem, we add a regularization penalty term. Can adding the penalty term decrease the training (in-sample) error? Will adding a penalty term always decrease the out-of-sample errors? Please justify your answers. Think about the case when there is over-fitting while training the model.

Question B: ℓ_1 regularization is sometimes favored over ℓ_2 regularization due to its ability to generate a sparse w (more zero weights). In fact, ℓ_0 regularization (using ℓ_0 norm instead of ℓ_1 or ℓ_2 norm) can generate an even sparser w, which seems favorable in high-dimensional problems. However, it is rarely used. Why?

Implementation of ℓ_2 regularization:

We are going to experiment with regression for the Red Wine Quality Rating data set. The data set is uploaded on the course website, and you can read more about it here: https://archive.ics.uci.edu/ml/datasets/Wine. The data relate 13 different factors (last 13 columns) to wine type (the first column). Each column of data represents a different factor, and they are all continuous features. Note that the original data set has three classes, but one was removed to make this a binary classification problem.

Download the data for training and testing. There are two training sets, wine_training1.txt (100 data points) and wine_training2.txt (a proper subset of wine_training1.txt containing only 40 data points), and one test set, wine_testing.txt (30 data points). You will use the wine_testing.txt dataset to evaluate your models.

We will train a *ℓ_2-regularized logistic regression* model on these data. Recall that the unregularized logistic error (a.k.a. log loss) is

$$E = -\sum_{i=1}^{N} \log(p(y_i|\mathbf{x}_i))$$

where $p(y_i = -1|\mathbf{x}_i)$ is

$$\frac{1}{1+e^{\mathbf{w}^T\mathbf{x}_i}}$$

and $p(y_i = 1|\mathbf{x}_i)$ is

$$\frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}_i}},$$

where as usual we assume that all $\mathbf{x}_i$ contain a bias term. The ℓ_2-regularized logistic error is

$$E = -\sum_{i=1}^{N} \log(p(y_i|\mathbf{x}_i)) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w} = \lambda\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \log\left(\frac{1}{1+e^{-y_i\mathbf{w}^T\mathbf{x}_i}}\right).$$

Implement SGD to train a model that minimizes the ℓ_2-regularized logistic error, i.e. train an ℓ_2-regularized logistic regression model. Train the model with 15 different values of λ starting with $\lambda_0 = 0.0001$ and increasing by a factor of 5, i.e.

$$\lambda_0 = 0.0001, \lambda_1 = 0.0005, \lambda_2 = 0.0025, ..., \lambda_{14} = 610351.5625.$$

Some important notes: When doing stochastic gradient descent, you will have to figure out a stopping condition that works well by trial and error. Refer to the last HW for one idea, but you may need to use different values. You will also need to find a good learning rate to use. Ideally, you want the largest learning

rate value that will still properly converge, but a smaller learning rate will not hurt, it will just take longer to converge. You can start with a learning rate of 10^{-6}. You should also initialize your weights to small random numbers.

You may run into numerical instability issues (overflow or underflow). One way to deal with these issues is by normalizing the input data X. Given the column for the jth feature, $X_{:,j}$, you can normalize it by setting $X_{ij} = \frac{X_{ij} - \overline{X_{:,j}}}{\sigma(X_{:,j})}$ where $\sigma(X_{:,j})$ is the standard deviation of the jth column's entries, and $\overline{X_{:,j}}$ is the mean of the jth column's entries. Normalization may change the optimal choice of λ. If you normalize the input data, simply plot the enough choices of λ to see any trends.

Question C: Do the following for both training data sets (wine_training1.txt and wine_training2.txt) and attach your plots in the homework submission (use a log-scale on the horizontal axis):

i. Plot the training error (E_{in}) versus different λs.

ii. Plot the test error (E_{out}) versus different λs using wine_testing.txt as the test set.

iii. Plot the ℓ_2 norm of $\mathbf{w}$ versus different λs.

You should end up with three plots, with two series (one for wine_training1.txt and one for wine_training2.txt) on each plot. Note that the E_{in} and E_{out} values you plot should not include the regularization penalty — the penalty is only included when performing gradient descent.

Question D : Given that the data in wine_training2.txt is a subset of the data in wine_training1.txt, compare errors (training and test) resulting from training with wine_training1.txt (100 data points) versus wine_training2.txt (40 data points). Briefly explain the differences.

Question E : Briefly explain the qualitative behavior (i.e. over-fitting and under-fitting) of the training and test errors with different λs while training with data in wine_training1.txt.

Question F: Briefly explain the qualitative behavior of the ℓ_2 norm of $\mathbf{w}$ with different λs while training with the data in wine_training1.txt.

Question G: If the model were trained with wine_training2.txt, which λ would you choose to train your final model? Why?

3 Lasso (ℓ_1) vs. Ridge (ℓ_2) Regularization

TAs: Jagriti Agrawal, Siddharth Murching
Relevant materials: Lecture 3

For this problem, you may use the scikit-learn (or other Python package) implementation of Lasso and Ridge regression — you don't have to code it yourself.

The two most regularized regression models are Lasso (ℓ_1) regression and Ridge (ℓ_2) regression. Although both enforce "simplicity" in the models they learn, only Lasso regression results in sparse weight vectors. This problem compares the affect of the two methods on the learned model parameters.

Question A: The tab-delimited file problem3data.txt on the course website contains 1000 9-dimensional

datapoints. The first 9 columns contain $x_1, \ldots, x_9$, and the last column contains the target value y.

i. Train a linear regression model on the problem3data.txt data with Lasso regularization for various choices of regularization strength α. On a single plot, plot each of the model weights $w_1, ..., w_9$ (ignore the bias/intercept) as a function of α. Start with $\alpha = 0$ and increase α until all weights are small.

ii. Repeat **i.** but with Ridge regression.

iii. As the regularization parameter increases, what happens to the number of model weights that are exactly zero with Lasso regression? What happens to the number of model weights that are exactly zero with Ridge regression?

Question B:

i. Given a dataset containing N datapoints each with d features, solve for

$$\arg\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}^T\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|_1$$

where $\mathbf{X} \in \mathbb{R}^{N\times d}$ is the matrix of datapoints and $\mathbf{y} \in \mathbb{R}^N$ is the vector of all output values for these datapoints. Do so just for the case where $d = 1$ and $\lambda \geq 0$.

This is linear regression with Lasso regularization.

ii. Suppose that when $\lambda = 0$, $w_1 \neq 0$. Does there exist a value for λ such that $w_1 = 0$? If so, what is the smallest such value?

iii. Given a dataset containing N datapoints each with d features, solve for

$$\arg\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}^T\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|_2^2$$

where $\mathbf{X} \in \mathbb{R}^{N\times d}$ is the matrix of datapoints and $\mathbf{y} \in \mathbb{R}^N$ is the vector of all output values for these datapoints. Do so for arbitrary d and $\lambda \geq 0$.

This is linear regression with Ridge regularization.

iv. Suppose that when $\lambda = 0$, $w_i \neq 0$. Does there exist a value for $\lambda > 0$ such that $w_i = 0$? If so, what is the smallest such value?