6.867 Machine Learning
Homework 2

0.1 LOGISTIC REGRESSION

Logistic Regression is a discriminative model used for classification. Given an input x, it finds the posterior of x belonging to one of the classes and then uses that probability to classify x. In the simplest case, it takes the dot product of x and w and uses that as an input to the sigmoid function, which outputs a number between 0 and 1. An advantage of logistic regressions is that they have few parameters, which allows them to be trained relatively quickly. One of the problems with logistic regressions is that they are very prone to overfitting to the training data. One way to prevent the overfitting is to add a regularization term, lambda, which penalizes the size of the weight vector. The size of the weight vector can be penalized using the L1 norm of the L2 norm. Here, we explore how different lambda values and the different norms affect several aspects of the logistic regression.

0.1.1 Optimizing with Gradient Descent

To investigate how regularization affected the logistic regression we tried lambda values of 0 and 1. We decided not to penalize the bias term in the weight vector. As expected, we found that with a lambda value of 1 the weight vector decreased in every iteration of the algorithm.

0.1.2 Section1

0.1.3 Section2

0.2 SUPPORT VECTOR MACHINES

Support Vector Machines are supervised learning models that work by finding a dividing hyperplane between the training data while maximizing the gap between the training data and the decision boundary. This is to help the classifier generalize better and makes it more robust to noise. Assuming the data is linearly separable, finding this dividing hyperplane amounts to solving the quadratic program

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 \\ \text{s. t.} \quad & y^i(w^T x^i + b) \geq 1, 1 \leq i \leq n \end{aligned} \tag{1}$$

where w is the vector perpendicular to the dividing hyperplane and $\frac{1}{||w||}$ is the size of the margin.

If the data is almost but not completely linearly separable, we can still model the data with an SVM by introducing slack variables when solving for a classifier. We allow the training points to be misclassified by some amount e and the goal is to maximize the margin while minimizing the slack. This formulation is called C-SVM and the separating hyperplane can be found by solving the quadratic program below.

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 + C\sum_i e_i \\ \text{s. t.} \quad & y^i(w^T x^i + b) \geq 1 - e_i, 1 \leq i \leq n \\ & e_i \geq 0, 1 \leq i \leq n \end{aligned} \tag{2}$$

0.2.1 Dual Formulation of C-SVM

This problem as stated above is difficult to implement with kernels functions. Fortunately, we convert the problem from the primal formulation stated above and instead implement its dual formuation.

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j [\phi(x_i)^T \phi(x_j)] - \sum_{t=1}^{n} \alpha_t \\ \text{s. t.} \quad & 0 \leq a_t \leq C, \sum_{t=1}^{n} \alpha_t y_t = 0 \end{aligned} \tag{3}$$

To solve this linear program and get the αs, we used the solver in a python library called cvxopt. The solver finds solutions to quadratic programs of the form:

$$\begin{aligned} \min \quad & \frac{1}{2}x^T P x + q^T x \\ \text{s. t.} \quad & Gx \leq h \\ & Ax = b \end{aligned} \tag{4}$$

In our case, our matrices are

$$P = Diag(\overrightarrow{y})XX^T Diag(\overrightarrow{y})$$
$$q = -1 * \overrightarrow{1}$$
$$G = [I| - I]$$
$$h = [C * \overrightarrow{1} \,|\, \overrightarrow{0}\,]$$
$$A = \overrightarrow{y}$$
$$b = 0$$

$Diag(\overrightarrow{y})$ is a diagonal matrix with $y^{(i)}$s on the diagonal.

This solver returns $\overrightarrow{x}$ which is our $\overrightarrow{\alpha}$ vector. Testing this implementation,

Do the stupid example!

0.2.2 2D Dataset Results

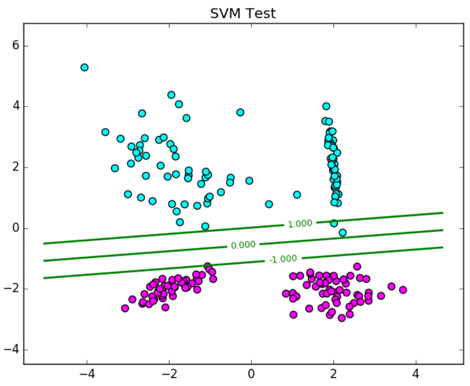
0.2.3 2D Dataset Results with Kernel Functions

0.3 SUPPORT VECTOR MACHINE WITH PEGASOS

0.3.1 Section1?

0.3.2 Section2?

0.4 HANDWRITTEN DIGIT RECOGNITION WITH MNIST

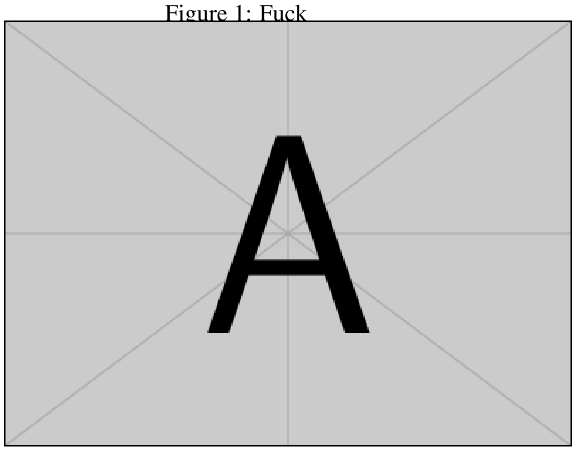


0.25

(a)

b

Figure 1: Fuck

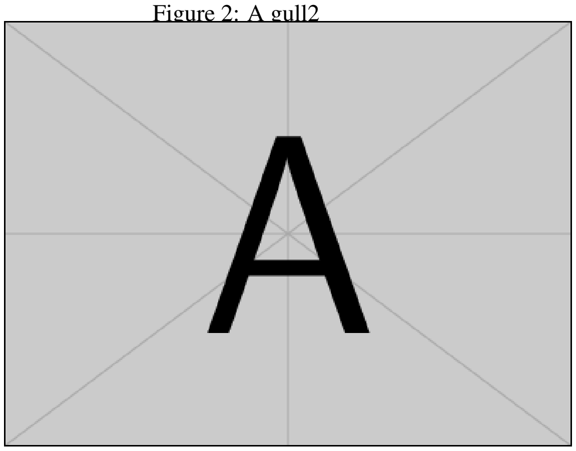


0.25

(a)

b

Figure 2: A gull2



0.25

(a)

b

Figure 3: A tiger