EE 5239 Nonlinear Optimization Homework 5 Cover Sheet

Instructor name: Mingyi Hong

Student name: Graham Droge

- Date assigned: Thursday 10/1/2019
- Date due: Tuesday 10/13/2019, at midnight
- This cover sheet must be signed and submitted along with the homework answers on additional sheets.
- By submitting this homework with my name affixed above,
  - I understand that late homework will not be accepted,
  - I acknowledge that I am aware of the University's policy concerning academic misconduct (appended below),
  - I attest that the work I am submitting for this homework assignment is solely my own, and
  - I understand that suspiciously similar homework submitted by multiple individuals will be reported to the Dean of Students Office for investigation.
- Academic Misconduct in any form is in violation of the University's Disciplinary Regulations and will not be tolerated. This includes, but is not limited to: copying or sharing answers on tests or assignments, plagiarism, having someone else do your academic work or working with someone on homework when not permitted to do so by the instructor. Depending on the act, a student could receive an F grade on the test/assignment, F grade for the course, and could be suspended or expelled from the University.

4.3.1

The problem is to solve the following equation

$$\min(x-a)^2+(y-b)^2+xy$$

$$s.t \quad x-1\leq 0 \quad -x\leq 0 \quad y-1\leq 0 \quad -y\leq 0$$

The first thing to do is write the Lagrangian with the included inequality constrain multipliers

$$L(x,y,\bar{\lambda})=(x-a)^2+(y-b)^2+xy-\lambda_1(x-1)+\lambda_2(-x)+\lambda_3(y-1)+\lambda_4(-y)$$

Now using the KKT conditions we can write the equality conditions below

$$2(x-a)+y=\lambda_2-\lambda_1 \tag{1}$$
$$2(y-b)+x=\lambda_4-\lambda_3 \tag{2}$$
$$\lambda_1(x-1)=0 \tag{3}$$
$$\lambda_2(-x)=0 \tag{4}$$
$$\lambda_3(y-1)=0 \tag{5}$$
$$\lambda_4(-y)=0 \tag{6}$$
$$\lambda_1,\lambda_2,\lambda_3,\lambda_4\geq 0 \tag{7}$$

Now we start analyzing the possible solutions for the problem. For one we notice that λ_2,λ_1 cannot both be non-zero and this applies for λ_3,λ_4.

Case 1 $\lambda_1,\lambda_3\neq 0$

$$x,y=1\longrightarrow\lambda_1=2a-3,\lambda_3=2b-3$$

For the previous equations to satisfy the non-negative constraints of $\bar{\lambda}$ we have

$$a\geq\frac{3}{2},b\geq\frac{3}{2}$$

Case 2 $\lambda_1,\lambda_4\neq 0$

$$x=1,y=0\longrightarrow\lambda_1=2a-2,\lambda_4=1-2b$$
$$a\geq 1,b\leq 1$$

Case 3 $\lambda_2,\lambda_3\neq 0$

$$x=0,x=1\longrightarrow\lambda_2=1-2a,\lambda_3=2b-2$$
$$a\leq 1,b\geq 1$$

Case 4 $\lambda_2, \lambda_4 \neq 0$

$$x, y = 0 \longrightarrow \lambda_2 = -2a, \lambda_4 = -2b$$
$$a \leq 0, b \leq 0$$

Thus our final solution depends on the values on a, b and can written as the following

$$(a \geq \frac{3}{2}), (b \geq \frac{3}{2}) \longrightarrow x = 1, y = 1$$
$$(a \geq 1), (b \leq 1) \longrightarrow x = 1, y = 0$$
$$(a \leq 1), (b \geq 1) \longrightarrow x = 0, y = 1$$
$$(a \leq 0), (b \leq 0) \longrightarrow x = 0, y = 0$$

4.3.3

The original problem is the following

$$\max \quad x_N + \sum_{i=0}^{N-1}(1 - u_i)x_i$$
$$s.t \quad u_i - 1 \leq 0 \quad - u_i \leq 0$$

we consider the case when $N = 2$. Let's put it in terms of one variable x_0 and analyze the left term first

$$\max \quad x_2 + \sum_{i=0}^{N-1}(1 - u_i)x_i$$
$$x_2 = x_1 + wu_1x_1 \quad x_1 = x_0 + wu_0x_0$$
$$x_2 = x_0 + wu_0x_0 + wu_1x_0 + w^2u_1u_0x_0 \quad (1)$$

Now analyze the right term

$$\sum_{i=0}^{1}(1 - u_i)x_i = 2x_0 - x_0u_0 - x_0u_1 + wu_0x_0 - 2u_0u_1x_0 \quad (2)$$

Now form the KKT conditions

$$wx_0 + w^2u_1x_0 - x_0 + wx_0 - 2u_1x_0 = \lambda_1 - \lambda_3 \tag{8}$$
$$wx_0 + w^2u_0x_0 - x_0 - wu_0x_0 = \lambda_2 - \lambda_4 \tag{9}$$
$$\lambda_1(u_0 - 1) = 0 \tag{10}$$
$$\lambda_2(-u_1 - 1) = 0 \tag{11}$$
$$\lambda_3(-u_0) = 0 \tag{12}$$
$$\lambda_4(-u_1) = 0 \tag{13}$$
$$\lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0 \tag{14}$$

We see that λ_1, λ_3 cannot both be non-active and the same for λ_2, λ_4.

Case 1 $\lambda_1, \lambda_2 \neq 0$

$$u_0, u_1 = 1$$
$$(8) \longrightarrow 2wx_0 + w^2x_0 - x_0 - wx_0 = \lambda_1 \longrightarrow (w^2 + w - 1)x_0 = \lambda_1$$
$$(9) \longrightarrow w^2x_0 - -x_0 = \lambda_2 \longrightarrow (w^2 - 1)x_0 = \lambda_2$$

For these two equations to satisfy the constraints on λ as well as being strictly positive we get

$$w > 1 \longrightarrow u_i = 1$$

Case 2 $\lambda_1, \lambda_4 \neq 0$

$$u_0 = 1, u_1 = 0$$
$$(8) \longrightarrow (2w - 1)x_0 = \lambda_1$$
$$(9) \longrightarrow (w^2 - 1)x_0 = -\lambda_4$$
$$\frac{1}{2} \leq w \leq 1 \longrightarrow u_0 = 1, u_1 = 0$$

Case 3 $\lambda_3, \lambda_2 \neq 0$

$$u_0 = 0, u_1 = 1$$
$$(8) \longrightarrow (w^2 + w - 1)x_0 = -\lambda_3$$
$$(9) \longrightarrow (w - 1)x_0 = \lambda_2$$

Case 3 cannot possible due to the two requirements for w contradict each other so it is not a possible case.

Case 4 $\lambda_3, \lambda_4 \neq 0$

$$u_0 = 1, u_1 = 1$$
$$(8) \longrightarrow (w^2 + w - 1)x_0 = -\lambda_3$$
$$(9) \longrightarrow (w - 1)x_0 = -\lambda_4$$

To satisfy the constraints on λ and the fact that $w > 0$

$$0 < w < \frac{1}{2} \longrightarrow u_i = 0$$

4.4.1

The problem is the following

$$\min \sum_{i=0}^{n} p_i s_i$$
$$s.t \quad Q - \sum d_i \leq 0, \quad s_i - d_i \leq 0, \quad -s_i \leq 0$$

We can also infer that to minimize the previous relationship the following relationship should hold $\sum_i s_i - Q = 0$. Form the Lagrangian equation and form the KKT conditions

$$L(s_i, \lambda, \mu) = -p_i + \lambda_2 - \lambda_3 + \mu_1 = 0$$
$$\lambda_1(Q - \sum d_i) = 0 \quad \lambda_2(s_i - d_i) = 0, \quad \lambda_3(-s_i) = 0, \quad \sum s_i - Q = 0$$

λ_2, λ_3 cannot both be non-zero. Let's go over the possible cases.

Case 1 $\lambda_3 = 0, \lambda_2 \neq 0$

$$p_i = \lambda_2 + \mu_1, \quad s_i = d_i, \quad Q \neq \sum d_i$$

Case 2 $\lambda_2 = 0, \lambda_3 \neq 0$

$$p_i = \mu_1 - \lambda_3, \quad s_i = 0, \quad Q = \sum d_i$$

Since λ is constrained to be greater or equal to zero we can see that we can mark μ_1 as y. If $p_i > y$ we get $s_i = d_i$ and if $p_i < y$ we get $s_i = 0$. Lets analyze the case where $p_i = y$

Case 3 $\lambda_2, \lambda_3 = 0$

$$p_i = \mu_1, \quad s_i \neq 0, \quad s_i \neq d_i, \quad \sum s_i = Q$$

4.4.3

The first duality proof we have is

$$\min_{A^{'}x \geq b} c^{'}x \Longleftrightarrow \max_{A\mu = c, \mu \geq 0} b^{'}\mu$$

Start by writing the Lagrangian for the left term

$$L(x, \mu) = c^{'}x + \mu(b - A^{'}x)$$

$$L(\lambda) = \inf_x \Big((c - A\mu)^{'}x + b^{'}\mu\Big)$$

From this we can see that since x is unconstrained we can approach $-\infty$ whether $(c - A\mu)$ is positive or negative so our maximization dual problem becomes the following with an equality constraint $(c - A\mu) = 0$ and the constraint $\mu \geq 0$ which is required for inequality constraints.

$$\max_{A\mu = c, \mu \geq 0} b^{'}\mu$$

Now lets prove that the dual equation for this form is the original primal problem. Writing our Lagrangian again with some new variables, λ and with respect to the variable μ as well as specifying it as a minimum by factoring in a negative sign.

$$L(\mu, \lambda) = (-b^{'}\mu) + \lambda(A\mu - c)$$

$$L(\lambda) = \inf_{\mu \geq 0} \Big((-b + A^{'}\lambda)\mu - c^{'}\lambda\Big)$$

Looking at the previous equation we now have a constrained variable we are minimizing over where if $(-b + A^{'}\lambda) < 0$ then $x = \infty$ for a minimum and if $(-b + A^{'}\lambda) > 0$ the minimum occurs at $x = 0$ so we only consider the latter case. Note too that since we have an equality constraint λ, it is unconstrained and can take positive and negative values. Thus we can write the dual form as the following where we can replace $\lambda = x$ if we wanted and factoring a negative sign to change it to a minimum. This is the same as the primal problem.

$$\min_{A^{'}\lambda \geq b} c^{'}\lambda$$

The next duality proof we have is

$$\min_{A^{'}x \geq b, x \geq 0} c^{'}x \iff \max_{A\mu \leq c, \mu \geq 0} b^{'}\mu$$

Write the Lagrangian for the left hand term

$$L(x,\mu) = c^{'}x + \mu(A^{'}x - b) \quad x \geq 0$$

$$L(\mu) = \inf_{x \geq 0} \Big((c - A\mu)x + b^{'}\mu\Big)$$

We can use the same observations as before to conclude that we need to consider the constraint $(c - A\mu) \leq 0$, thus using the fact that μ is constrained to being greater than or equal to 0 we can write

$$\max_{A\mu \leq c, \mu \geq 0} b^{'}\mu$$

Now we will take the dual of this equation to show that it is equal to the primal. The Lagrangian can be written as a minimization

$$L(\mu,\lambda) = (-b^{'}\mu) + \lambda(A\mu - c) \quad \mu \geq 0$$

$$L(\lambda) = \inf_{\mu \geq 0} \Big((-b + A^{'}\lambda)\mu - c^{'}\lambda\Big)$$

Once again we see that the constraint we need to consider is $(c - A\mu) \geq 0$ along with the constraint that $\lambda \geq 0$ from the inequality constraint. We can write the new equation below and we can see it equals the primal problem with a variable substitution $\lambda = x$.

$$\max_{A^{'}\lambda \geq b, \lambda \geq 0} -c^{'}\lambda \quad = \quad \min_{A^{'}\lambda \geq b, \lambda \geq 0} c^{'}\lambda$$

Coding Problems

SVM with Stochastic Sub-Gradient Descent

The SVM model is constructed so that not only is the loss minimized over the data points but the distance between the data points and the decision boundary are maximized. This is done to better generalize to unseen data so that the decision boundary doesn't separate the data in a non-optimal way. The first algorithm that I designed to solve the SVM problem was the stochastic sub-gradient descent algorithm. This algorithm has similar properties to steepest descent but with some modifications that better conform to the objective function. The objective function for the SVM problem can be written as

$$\min_{\mathbf{w}} f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \max(0, 1 - y_i\mathbf{w}^T x_i)$$

The sum over the max is a non-smooth function so it is not differential at the point 0. To better remedy this the gradient first checks to see if the max function is greater than 0, which if it is the gradient is just the gradient of $\sum 1 - y_i\mathbf{w}^T x_i$. If it is not greater than 0 than the gradient is obviously 0. The other modification that is made is the stochastic selection of a point to evaluate the gradient at. You can see that the gradient is a sum over the entire data set, which is undesirable when the data set is very large as this will significantly slow down the speed of the algorithm. At each iteration a random point is chosen to evaluate the gradient at. This has the benefit of a faster algorithm (but possibly worse solution) and with this approach the algorithm has a better opportunity to escape non-optimal local mins and possibly find more optimal points. The figure below shows the decision boundary that was computed for the stochastic sub-gradient algorithm. You can see that the decision boundary does a decent job at separating the data points though better separations are possible. The accuracy obtained on the test data set is %94.78 with $C = 5$ which very close to the accuracies obtained for the linear and logistic regression models.

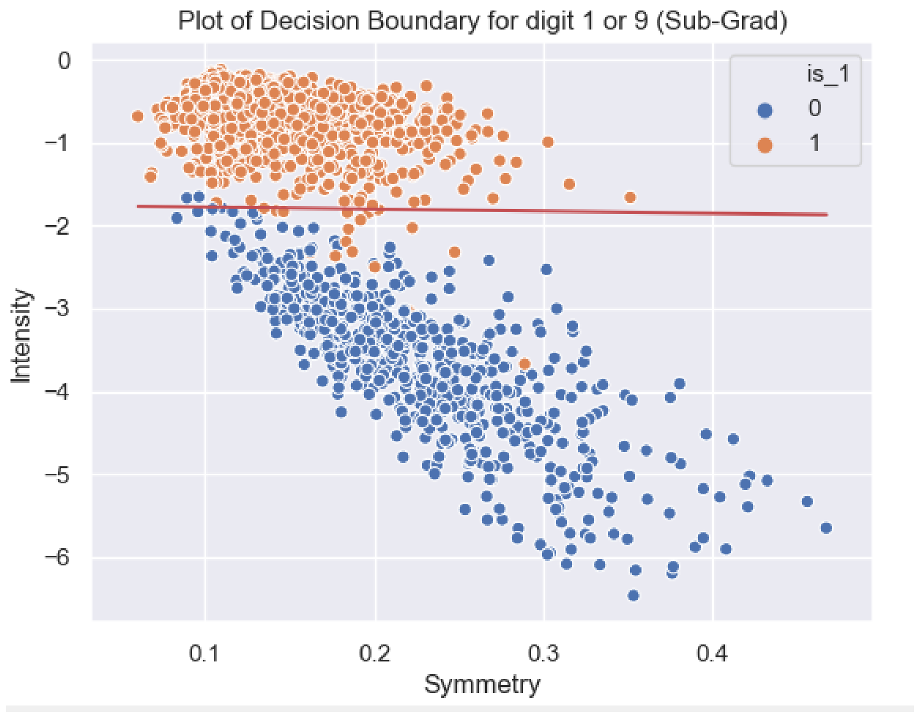


Figure 1: Feature data for digit 1 and 9 with decision boundary from fitting linear model with train data

SVM with Coordinate Descent on the Dual

The other algorithm that I designed is a coordinate descent algorithm that iterates over coordinates sequentially and updates one at a time. The objective function that I am minimizing over changes from the primal form to the dual form

$$\min_{\alpha} f(\alpha) = \frac{1}{2}\alpha^T \bar{Q}\alpha + e^T\alpha$$

$$s.t \quad 0 \leq \alpha_i \leq C$$

where $\bar{Q} = Q + D$, $Q_{ij} = y_i y_j x_i^T x_j, x \in R^d$. This algorithm is based on the paper that was mentioned on the slides. With the objective function in this form we can iterate over α and simultaneous solve for x from the primal and dual form relationships. The algorithm loops through each coordinate of α and computes a projection using the $min(max())$ procedure. The α are then updated as well as the x vector. After the fitting of the algorithm has run through enough steps we calculate the bias term by looking at the support vectors that were calculated for the non-zero α's. So the steps are to form the Gram Matrix Q_{ij}, loop through coordinates and calculate projection gradient, update the

α and variable together, and lastly calculate the bias based on the support vectors. After running the algorithm on the given data set I get a decision boundary that is shown in the figure below. This decision boundary seems to be more robust to generalization error. Testing the accuracy on the test data gives an accuracy of around %95.91 with $C = 10$ which is very good.

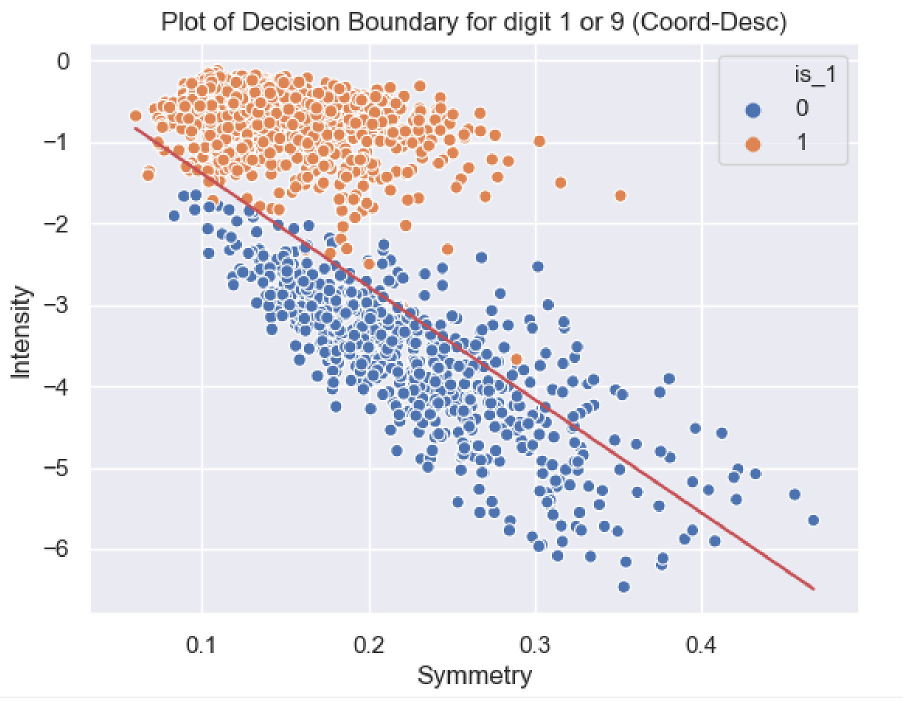
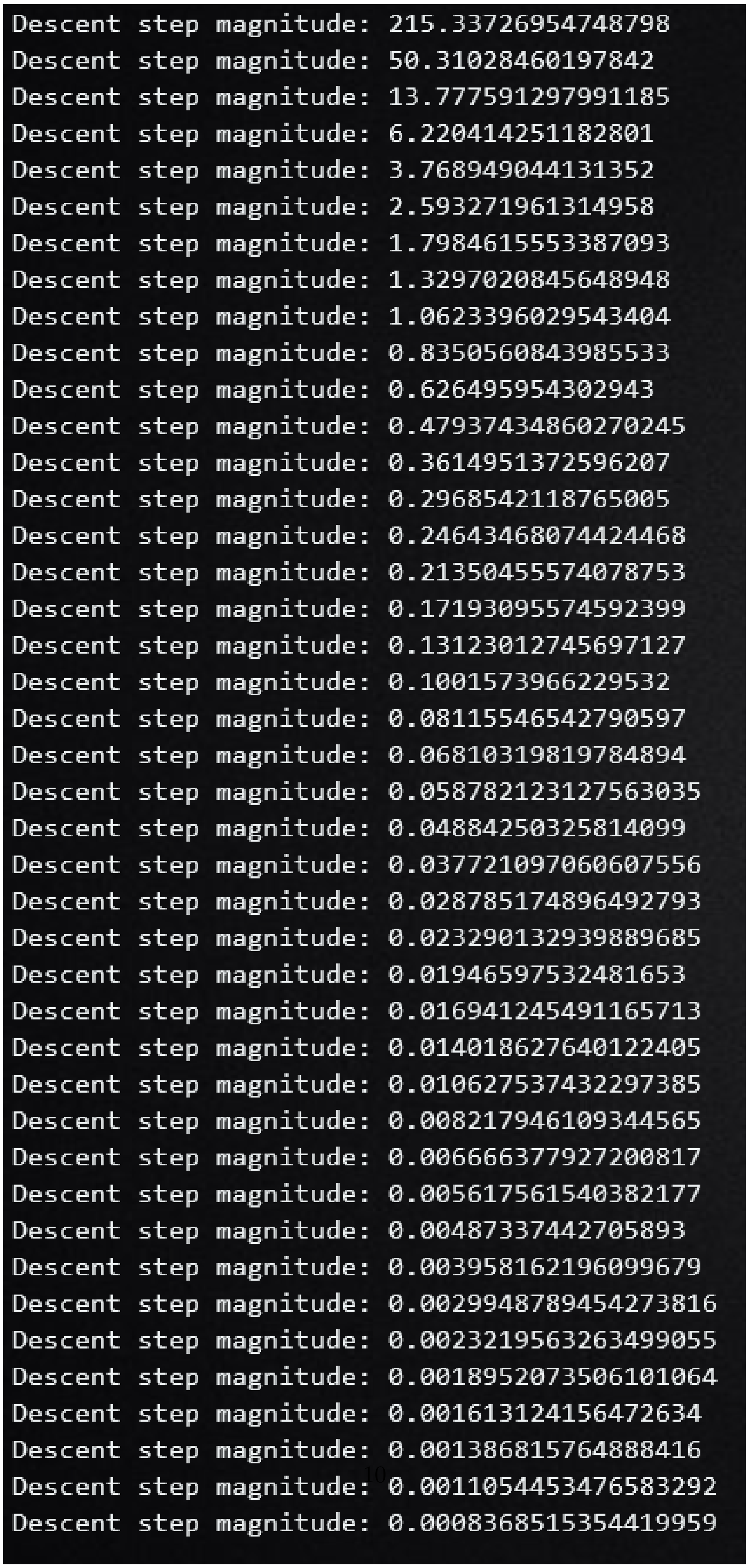


Figure 2: Feature data for digit 1 and 9 with decision boundary from fitting linear model with train data

Option 2

For option 2 we need to run the previous problem of one of the algorithms on the full data feature set with 256 features. My task before was to split the digits 1 and 9 but when i looked at the raw data set it looks like the digits stop at 8 so I changed the task to separating the digits 1 and 8. I ran the the previous developed SVM model with the coordinate descent algorithm on the dual formulation for the full feature set. This was chosen due to the fact that from the previous analysis it would generalize better to other data points and is a more robust model. I modified the previous model to include the kernel trick for pushing the feature set into a higher dimensional space. This higher dimensional space gives a better representation to find linear boundaries between instances while at the same time keeping the computational time low by using the kernel trick. I used the radial basis function for the choice of kernel. I used the simple dot product for the previous SVM model since we were only considering two features. The results that were obtained for this model and algorithm were a very good %99.84 on the training set but a disappointing %61.31 on the test data set. Where do we loose our classification accuracy? This loss in accuracy stems from the added dimensionality of the data set. With 256 features and only a total of 649 training instances we don't have enough data to build a robust model in higher dimensions. So while the accuracy is very good on our training set it doesn't scale well to the test data. I believe this issue deals with the so called "curse of dimensionality". When increasing the feature space to higher dimensions the search space to search for an optimal boundary exponentially grows. With such a small number of observation set the model. cannot search this space robustly to build a model. Since plotting the features space is difficult, the figure below shows the descent step magnitudes as the algorithm operates.



```
Descent step magnitude: 215.33726954748798
Descent step magnitude: 50.31028460197842
Descent step magnitude: 13.777591297991185
Descent step magnitude: 6.220414251182801
Descent step magnitude: 3.768949044131352
Descent step magnitude: 2.593271961314958
Descent step magnitude: 1.7984615553387093
Descent step magnitude: 1.3297020845648948
Descent step magnitude: 1.0623396029543404
Descent step magnitude: 0.8350560843985533
Descent step magnitude: 0.626495954302943
Descent step magnitude: 0.47937434860270245
Descent step magnitude: 0.3614951372596207
Descent step magnitude: 0.2968542118765005
Descent step magnitude: 0.24643468074424468
Descent step magnitude: 0.21350455574078753
Descent step magnitude: 0.17193095574592399
Descent step magnitude: 0.13123012745697127
Descent step magnitude: 0.1001573966229532
Descent step magnitude: 0.08115546542790597
Descent step magnitude: 0.06810319819784894
Descent step magnitude: 0.058782123127563035
Descent step magnitude: 0.04884250325814099
Descent step magnitude: 0.037721097060607556
Descent step magnitude: 0.028785174896492793
Descent step magnitude: 0.023290132939889685
Descent step magnitude: 0.01946597532481653
Descent step magnitude: 0.016941245491165713
Descent step magnitude: 0.014018627640122405
Descent step magnitude: 0.010627537432297385
Descent step magnitude: 0.008217946109344565
Descent step magnitude: 0.006666377927200817
Descent step magnitude: 0.005617561540382177
Descent step magnitude: 0.00487337442705893
Descent step magnitude: 0.003958162196099679
Descent step magnitude: 0.0029948789454273816
Descent step magnitude: 0.0023219563263499055
Descent step magnitude: 0.0018952073506101064
Descent step magnitude: 0.001613124156472634
Descent step magnitude: 0.001386815764888416
Descent step magnitude: 0.0011054453476583292
Descent step magnitude: 0.0008368515354419959
```

Figure 3: Descent magnitude values for the coordinate descent algorithm