Homework 2

SVM, Kernel methods, Ensemble learning, learning theory

CMU 10-701: Machine Learning (Fall 2016)
https://piazza.com/class/is95mzbrvpn63d
OUT: September 26th
DUE: October 10th, 11:59 PM

START HERE: Instructions

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 3.4"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only.
- **Late Submission Policy:** Late submissions will not receive full credit. Half credit will be awarded to correct solutions submitted within 48 hours of the original deadline. Otherwise, no credit will be given.
- **Submitting your work:** Non-programming parts of the assignment should be submitted as PDFs using Gradescope unless explicitly stated otherwise. Each derivation/proof should be completed on a separate page. Submissions can be handwritten, but should be labeled and clearly legible. If your writing is not legible, you will not be awarded marks. Alternatively, submissions can be written in LaTeX. Upon submission, label each question using the template provided.

Problem 1: SVM [30] (Brynn)

1.1 Soft-margin SVM

In real world applications, there are outliers in data. This can be dealt with using a soft margin, specified in a slightly different optimization problem as below (soft-margin SVM):

$$\min \frac{1}{2} w \cdot w + C \sum_{i}^{N} \xi_i \text{ where } \xi_i \geq 0$$

$$\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i$$

ξ_i represents the slack for each data point i, which allows misclassification of datapoints in the event that the data is not linearly seperable. SVM without the addition of slack terms is known as hard-margin SVM.

1. **[3 pt]** Intuitively, where does a data point lie relative to where the margin is when $\xi_i = 0$? Is this data point classified correctly?

2. **[4 pt]** Intuitively, where does a data point lie relative to where the margin is when $0 < \xi_i \leq 1$? Is this data point classified correctly?

3. **[3 pt]** Intuitively, where does a data point lie relative to where the margin is when $\xi_i > 1$? Is this data point classified correctly?

1.2 Kernel SVM

Support Vector Machines can be used to perform non-linear classification with a kernel trick. Recall the hard-margin SVM from class:

$$\min \frac{1}{2} w \cdot w$$

$$\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1$$

The dual of this primal problem can be specified as a procedure to learn the following linear classifier:

$$f(x) = \sum_{i}^{N} \alpha_i y_i (x_i^T x) + b$$

Note that now we can replace $x_i^T x$ with a kernel $k(x_i, x)$, and have a non-linear decision boundary.

In Figure 5, there are different SVMs with different shapes/patterns of decision boundaries. The training data is labeled as $y_i \in \{-1, 1\}$, represented as the shape of circles and squares respectively. Support vectors are drawn in solid circles. Match the scenarios described below to one of the 6 plots (note that one of the plots does not match to anything). Each scenario should be matched to a unique plot. Explain in less than two sentences why it is the case for each scenario.

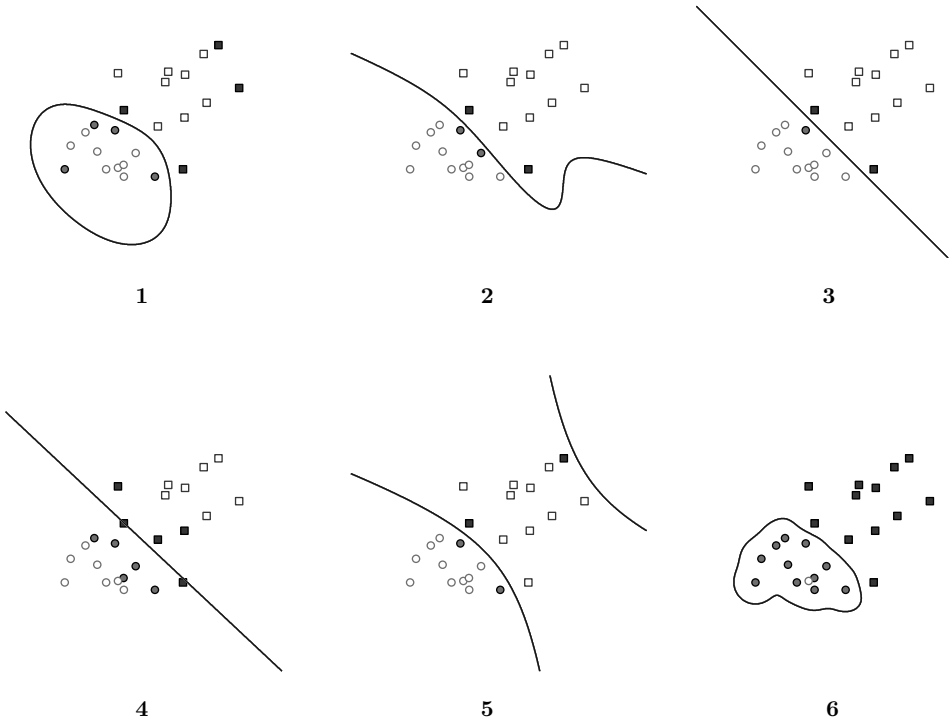

Figure 1: SVM boundaries

1. **[4 pt]** A soft-margin linear SVM with $C = 0.02$.
2. **[4 pt]** A soft-margin linear SVM with $C = 20$.
3. **[4 pt]** A hard-margin kernel SVM with $k(u, v) = u \cdot v + (u \cdot v)^2$
4. **[4 pt]** A hard-margin kernel SVM with $k(u, v) = exp(-5||u - v||^2)$
5. **[4 pt]** A hard-margin kernel SVM with $k(u, v) = exp(-\frac{1}{5}||u - v||^2)$

Problem 2: Feature Representation and Kernels [30] (Brynn)

2.1 Designing Transformations

In this problem, we will design some transformations of the original data points, i.e., derive features, to try to make a dataset linearly separable.
Note: for the following questions (1)–(5), if your answer is 'Yes', write out the expression for the transformation; if your answer is 'No', briefly explain why.

1. **[4 pt]** Consider the following 1-D dataset (as shown in Figure 2). Can you think of a 1-D transformation that will make the points linearly separable?

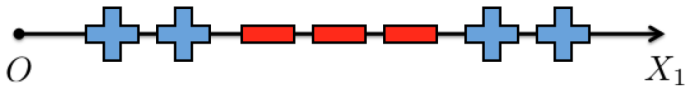

Figure 2: Dataset for question (a) & (b)

2. **[4 pt]** Still consider the above 1-D dataset (as shown in Figure 2). Can you come up with a 2-D transformation that makes the points linearly separable?
3. **[4 pt]** You may not always need to map to a higher dimensional space to make the data linearly separable. Consider the following 2-D dataset (as shown in Figure 3). Can you suggest a 1-D transformation that will make the data linearly separable?

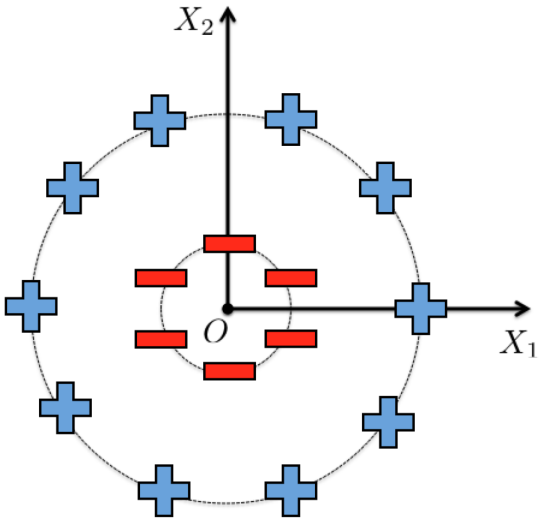

Figure 3: Dataset for question 3

4. **[4 pt]** Using ideas from the above two datasets, can you suggest a 2-D transformation of the following dataset (as shown in Figure 4) that makes it linearly separable?

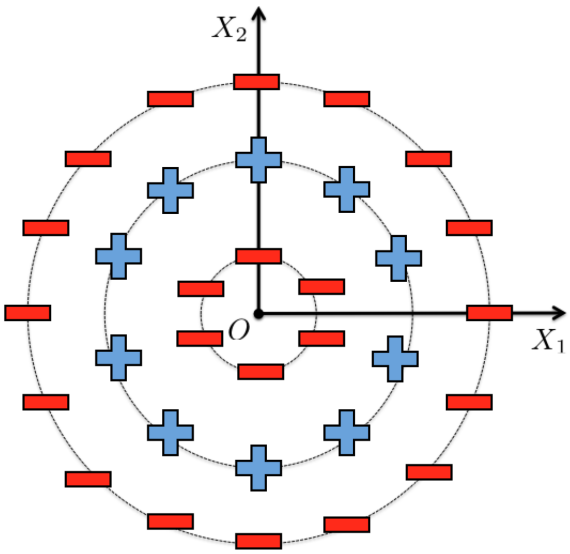

Figure 4: Dataset for question 4

5. **[2 pt]** What is the kernel corresponding to the transformation you designed in question (4)? Explicitly write out the expression for this kernel.

2.2 Kernel or Not?

For each of the following functions, prove or disprove that it is a valid kernel.

1. **[2 pt]** $k(x, z) = (xz + 1)^2$
2. **[2 pt]** $k(x, z) = (xz - 1)^3$

Problem 3: Ensemble Learning [40] (Hyun-Ah)

In this problem, we will explore various ensemble learning methods, and see their effects on the bias and variance.

3.1 Bias and variance decomposition [5] (Hyun-Ah)

In this problem, you will show that expected prediction error can be decomposed into bias, variance and noise.
Let's think about a (unobservable) true function $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Given a set of training dataset $D = \{(\mathbf{x}_i, y_i)\}$, we can fit a hypothesis $h(\mathbf{x}|D) = \mathbf{w}^T\mathbf{x} + b$ to the given data D to minimize squared error $\sum_i (y_i - h(\mathbf{x}_i|D))^2$ so that $h(\mathbf{x}|D)$ approximates the true function $f(\mathbf{x})$.
Given a new data point $\mathbf{x}^{'}$, and observed value $y^{'} = f(\mathbf{x}^{'}) + \epsilon$, we would like to minimize the expected prediction error $E_{D,\epsilon}[(h(\mathbf{x}^{'}|D) - y^{'})^2]$.

1. **[5 pt]** The bias of $\mathbf{x}^{'}$ is the difference between the expected value of hypotheses and the true value $f(\mathbf{x}^{'})$. The variance of $\mathbf{x}^{'}$ is how far a set of hypotheses learned on different Dataset D are spread out from their mean $E_D[h(\mathbf{x}^{'}|D)]$.
   Show that expected prediction error of $\mathbf{x}^{'}$ ($E_{D,\epsilon}[(h(\mathbf{x}^{'}|D) - y^{'})^2]$) can be decomposed into $(bias(\mathbf{x}^{'}))^2$, $variance(\mathbf{x}^{'})$, and $noise$, where $bias(\mathbf{x}^{'}) = E_D[h(\mathbf{x}^{'}|D)] - f(\mathbf{x}^{'})$, $variance(\mathbf{x}^{'}) = E_D[(h(\mathbf{x}^{'}|D) - E_D[h(\mathbf{x}^{'}|D)])^2]$, and noise=$\sigma^2$

You have just shown that the expected prediction error that we want to minimize can be decomposed into bias and variance. Now let's see how the ensemble methods effect these values and thus affect the expected prediction error.

3.2 Bootstrap aggregating (Bagging) [5] (Hyun-Ah)

Bootstrap aggregating or bagging is one of the ensemble methods. Below is the pseudocode for bagging method. Given a set of training samples, bootstrap size, and a learning algorithm, bagging method returns a set of hypothesis learned on each of the bootstrap samples h_b. For a test sample x_i, we can determine the label by weighted voting of the hypothesis: $y_i = \frac{\sum_b h_b(x_i)}{B}$.

Data: A set $\mathcal{S}$ of m labeled training samples: $\mathcal{S} = \{(x_i, y_i)\}, i = 1, \cdots, m$, where $y_i \in \mathbb{R}$ are a real-valued output from a function , bootstrap size B, a learning algorithm
Result: A set of hypothesis $\mathcal{H} = \{h_b, b = 1, \cdots, B\}$
for $b = 1, \cdots, B$ **do**
  Create a bootstrap sample S_b;
  (Randomly draw $|\mathcal{S}|$ samples from $\mathcal{S}$ with replacement);
  Learn a hypothesis h_b by applying a learning algorithm on data S_b
end

Algorithm 1: Bagging algorithm

1. **[3 pt]** Write down the bias and the variance for using the bagging method. By looking at the bias and the variance, is there any change in the expected prediction error compared to the case that does not use bagging method?

2. **[2 pt]** What kind of classifiers do you think will benefit from bagging method? Give a few examples of classifiers that will benefit from bagging method and explain why you think so.

We have briefly learned about one of the ensemble method. Now let's take a look at another ensemble method called 'Adaboost method'.

3.3 Adaboost [30] (Hyun-Ah)

Adaboost method is one of the commonly used ensemble method. Adaboost learns weights for both training samples and the classifiers. Unlike Bagging method, Adaboost method makes final decision by weighted voting of the classifiers. Every iteration, Adaboost updates the weights on training samples so that classifier can focus on learning samples that are difficult for classification. Adaboost reduces both bias and variance thus is effective in reducing the prediction error.

Data: A set $\mathcal{S}$ of m labeled training samples: $\mathcal{S} = \{(x_i, y_i), i = 1, \cdots, m\}$, where $y_i \in \{-1, 1\}$ are labels, number of iterations T, a set of hypothesis $\mathcal{H} = \{h_j\}$ from a learning algorithm
Result: A set of weights $\mathcal{A} = \{\alpha_t, t = 1, \cdots, T\}$ and classifiers selected at each iteration $\{h_t, t = 1, \cdots, T\}$
Initialize the weights $D_1(i) = \frac{1}{m}$;
for $t = 1, \cdots, T$ **do**
  Select a weak classifier with smallest weighted error $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j$, where $\epsilon_j = \sum_{i=1}^{m} D_t(i) \mathbb{1}_{\{y_i \neq h_j(x_i)\}}$, and let $\epsilon_t = \min_{h_j \in \mathcal{H}} \epsilon_j$;
  Compute the weight for the classifier $\alpha_t = \frac{1}{2} \log(\frac{1-\epsilon_t}{\epsilon_t})$;
  Update the weight $D_{t+1}(i) = D_t(i) \exp(-\alpha_t y_i h_t(x_i))$;
  Normalize the weight $D_{t+1}(i) = \frac{D_{t+1}(i)}{\sum_i D_{t+1}(i)}$;
end
Return weights α_t and classifiers h_t $t = 1, \cdots, T$.

Algorithm 2: Adaboost algorithm

1. **[8 pt]** In this problem, you will prove the error bound of the Adaboost. Let's denote H as the strong classifier. The decision of the strong classifier on new sample x^* is given as $H(x^*) = sign(\sum_t \alpha_t h_t(x))$. Recall that "weak" classifiers are defined as classifiers that do slightly better job than random guessing, $\epsilon_t = \frac{1}{2} - \gamma_t$, where $\gamma_t > 0$ is a small value.
   The upper bound of the training error of the strong classifier H is given as

$$err(H) \leq \prod_t^T 2\sqrt{\epsilon_t(1-\epsilon)} \tag{1}$$

$$= \prod_t^T \sqrt{1 - 4\gamma_t^2} \tag{2}$$

$$\leq \exp(-2 \sum_t^T \gamma_t^2) \tag{3}$$

   Now you will prove this bound step-by-step through (a)-(d).
   For notational simplicity, let $f(x) = \sum_t \alpha_t h_t(x)$, so that $H(x) = sign(f(x))$. Let Z_t denote a normalizing constant for weak classifier t: $Z_t = \sum_i D_{t+1}(i)$
   (a) **[2 pt]** First, show that $D_{T+1}(i) = \frac{1}{m} \frac{1}{\prod_t^T Z_t} \exp(-y_i f(x_i))$.

   (b) **[2 pt]** Show that error of the strong classifier H is upper bounded by the product of Z_t: $err(H) \leq \prod_t^T Z_t$.
   (hint 1: the error of the strong classifier H is given as: $err(H) = \frac{1}{m} \sum_i \mathbb{1}_{\{y_i \neq H(x_i)\}}$, hint 2: use the fact that 0-1 loss is upper bounded by exponential loss - recall that we are using 0-1 loss here by defining error as $\mathbb{1}_{\{y_i \neq H(x_i)\}}$).

(c) **[2 pt]** Now show that $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$
(hint 1: start from the $Z_t = \sum_i D_{t+1}(i) = \sum_i D_t(i)\exp(-\alpha_t y_i h_t(x_i))$*. hint 2: separate the* Z_t *expression for correctly classified cases and incorrectly classified cases. hint 3: express in terms of* ϵ_t*. hint 4: plug in* α_t*)*

(d) **[2 pt]** Combining (b) and (c), now we have upper bound on the error as $err(H) \leq \prod_t^T 2\sqrt{\epsilon_t(1-\epsilon_t)}$. Now show the following: $err(H) \leq \exp(-2\sum_t^T \gamma_t^2)$.
(hint 1: use the definition of weak classifier$\epsilon_t = \frac{1}{2} - \gamma_t$*, and plug in* ϵ_t *. hint 2: use the fact that* $1 - x \leq \exp^{-x}$*)*

2. **[22 pt]** Now let's implement the Adaboost method. Download the training and the test data [1]. In the folder, there are two csv files "train.txt" and "test.txt". "train.txt" has ten training samples ($m = 10$) of a two-dimensional vector $x_i \in \mathbb{R}^2$. Each row has three columns, where each column means (x_{i1}, x_{i2}, y_i) for each training sample. In Figure 5, the 10 training samples are plotted in blue * and red + for label -1 and label +1 respectively. "test.txt" has $10,201$ test samples. Each column is for (x^*_{i1}, x^*_{i2}) of each test sample.
For the learning algorithm, we will use decision stump that learns a vertical or horizontal hyperplane for classification. Construct a set of finite hypotheses $\mathcal{H}$ based on the training samples.
(tip: In Figure 6, an example of hypothesis is shown. In this example we have two training samples: $x_1 = (8, 13), x_2 = (12, 11)$. Given these two training samples, we can construct one decision boundary that separates these points for each dimension - vertical or horizontal. This gives us two hypotheses from given training samples. And there are two possibilities for assigning labels, -1 or +1 for the left part of the decision boundary. Therefore, we can construct four hypotheses from two training samples.

$$h_1(x_i) = 2 * (\mathbb{1}_{\{x_{i1}<=10\}} - 0.5)$$
$$h_2(x_i) = 2 * (\mathbb{1}_{\{x_{i1}>=10\}} - 0.5)$$
$$h_3(x_i) = 2 * (\mathbb{1}_{\{x_{i2}<=12\}} - 0.5)$$
$$h_4(x_i) = 2 * (\mathbb{1}_{\{x_{i2}>=12\}} - 0.5)$$

For constructing a set of hypothesis $\mathcal{H}$, you will need to store four information: (dimension - either vertical or horizontal, decision value, label assigned to the left part of the hyperplane, label assigned to the right part of the hyperplane))
Implement the Adaboost method as described in the pseudocode above. Let $T = 50$. Use the given training samples to learn a set of hypothesis $\{h_t, t = 1, \cdots, T\}$ and a set of weights for each hypothesis $\{\alpha_t, t = 1, \cdots, T\}$. Test the strong classifier for test sample x_i^* : $H(x_i^*) = sign(\sum_{t=1}^T \alpha_t h_t(x_i^*))$.
Plot the test samples of different labels with different colors and markers (e.g. magenta o for label +1 and green x for label -1), and also the training samples (e.g. red+ for label +1 and blue * for label -1). In the report, attach the plot. Submit your code to AutoLab.

[1] https://www.dropbox.com/sh/wek8499y5f0265d/AACz3wbA7bf6a-SD-2Yhemroa?dl=0

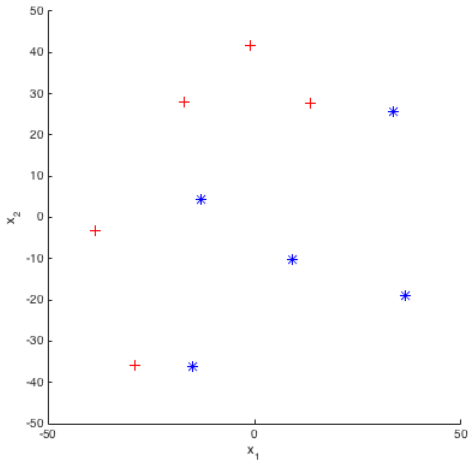

Figure 5: Training samples

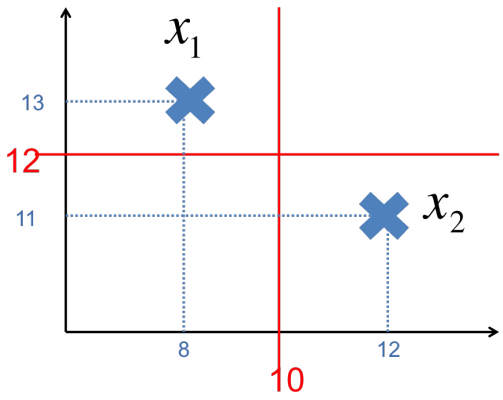

Figure 6: An example of hypothesis

Problem 4: Logistic Regression [20] (Devendra)

4.1 Multinomial Logistic Regression

Multinomial logistic regression is a classification method that generalizes logistic regression to multiclass problems. It has the form

$$p\left(y = c \mid \mathbf{x}, \mathbf{W}\right) = \frac{\exp\left(w_{c0} + \mathbf{w}_c^\top \mathbf{x}\right)}{\sum_{k=1}^{C} \exp\left(w_{k0} + \mathbf{w}_k^\top \mathbf{x}\right)}, \quad \forall c \in \{1, 2, ..., C\} \tag{4}$$

Here, C is the number of classes, and $\mathbf{W}$ is a $C \times (d+1)$ weight matrix, and d is the dimension of input vector $\mathbf{x}$. In other words, $\mathbf{W}$ is a matrix whose rows are the weight vectors for each class.

Given a set of training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, write the log likelihood of the data and derive the gradient ascent rule for Multinomial Logistic Regression.

4.2 Ordinal Logistic Regression

In many applications, the classes in the data will have an ordering, such as predicting the discrete integer rating given by an user among 1 to 5 stars. The multinomial logistic regression model doesn't utilize any information about the ordering of classes. The ordinal logistic regression model takes into account this ordering by modeling the cumulative probability of class labels as follows:

$$p\left(y \leq c \mid \mathbf{x}, \mathbf{w}\right) = \frac{\exp\left(\alpha_c - \mathbf{w}^\top \mathbf{x}\right)}{1 + \exp\left(\alpha_c - \mathbf{w}^\top \mathbf{x}\right)}, \quad \forall c \in \{1, 2, ..., C-1\} \tag{5}$$

Here, C is the number of classes, which are ordered from 1 to C, and $\mathbf{w}$ and $\mathbf{x}$ are vectors of dimension d.

Given a set of training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, write the log likelihood of the data and derive the gradient ascent rule for Ordinal Logistic Regression.

4.3 Comparison

1. Write one advantage and disadvantage of using ordinal logistic regression as compared to multinomial logistic regression.
2. How is ordinal logistic regression different from doing multiple binary logistic regressions, one for each class?

Problem 5: SVM Implementation - Document Classification [45] (Petar)

5.1 Dual of soft-margin SVM (20 points)

We learned about primal and dual forms of optimization problems. We are given the primal form of the SVM problem (soft margin version):

$$\min_{\beta,\beta_0,\xi} \frac{1}{2}||\beta||_2^2 + C\sum_{i=1}^{n}\xi_i \tag{6}$$

$$\text{s.t. } \xi_i \geq 0, \forall i \tag{7}$$

$$y_i(\mathbf{x_i}^T\beta + \beta_0) \geq 1 - \xi_i, \forall i \tag{8}$$

Where X is the data matrix with n rows corresponding to data points and d columns corresponding to features. β is a column vector of d predictors, and β_0 is a bias term. ξ_i correspond to slack variables, one or each of the n data points. $\mathbf{x_i}^T$ corresponds to i^{th} row in X.

Derive the dual problem.

Now, can you express the optimal weight vector in the dual formalism? (More specifically, if let's say α_i's are the Lagrange multipliers in the Lagrangian corresponding to the n constraints of the form: $y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i$, can you write the optimal weight vector in terms of the input points (and/or labels) and those α's?)

In order to make a prediction given a new test point (under the dual formalism), you will need the optimal bias term (β_0^*), in addition to the optimal weight vector. The optimal bias term (β_0^*) can be found by:

- Finding an i, such that $\alpha_i > 0$.
- Optimal β_0 then can be obtained as:

$$\beta_0^* = y_i - \sum_{j=1}^{n} y_j\alpha_j\mathbf{x_i}^T\mathbf{x_j} \tag{9}$$

5.2 Implementing soft-margin SVM (25 points)

We will now implement both the primal and the dual problem using the Quadprog function (in Matlab). We will use a subset of the "20 Newsgroups" dataset [2]. There are two topics of documents for classification taken from the talk.politics.misc and talk.religion.misc, for a total of 842 documents. X_{train} is a sparse matrix of training data, where each row is a document and each column is a feature (a word). $X_{ij} = 1$ corresponds to the document i containing the j-th word. Y_{train} is a vector of labels (1 or -1). X_{test} and Y_{test} follow the same pattern.

a. Use Quadprog to construct a solver for both the primal and the dual problems, using $C = 1$. For Quadprog in Matlab use the interior-point-convex algorithm. term in the objective. Please see: http://www.mathworks.com/help/optim/ug/quadprog.html. We will also look at R and Python as options and will post a reference on Piazza if we deem them appropriate.

b. Report the final objective values for both the primal and the dual problems.

c. Compare the β solutions for both the primal and the dual problem. Show a scatter plot of the two

[2] http://www.cs.cmu.edu/~mgormley/courses/10701-f16/homeworks/hw2/news.mat

vectors. What do you observe?

d. Calculate the mis-classification rate on the training and test data.

e. How many of the training points are: a) Missclassified?, b) Within the margin?. Report the dual variable for each of these points.

5.3 BONUS: optimal bias (5 points)

As a short bonus question, can you prove that the optimal bias in the dual problem, is actually given by the equation 9? (the points of this part are not included in the total of 45 for this question).