1. **Generalization and risk decomposition.** Govinda is designing a classifier to determine whether an email should be sent to the user's spam folder. He collects a dataset of 1000 emails, pays human experts as to annotate them with labels (spam or not spam), and splits the dataset into a training set, validation set and test set.

   He first decides to use the 10000 most common words of English as features for this classifier (let's denote this classifier g_{10000}). Later, his manager asks him to use a smaller number of features, so he also trains another classifier, g_{100}, which uses only the 100 most common words of English.

   (a) (2 points) Which is more likely to overfit?

   A. g_{100}

   B. g_{10000}

   C. They are both equally likely to overfit

   (b) (3 points) Is is approximation error higher when Govinda uses 100 features, when he uses 10000 features, or is it equal in both cases? **Briefly** explain using the definition of approximation error.

(c) (2 points) Is the Bayes error higher when Govinda uses 100 features, when he uses 10000 features, or is it equal in both cases? Briefly explain using the definition of the Bayes error.

(d) (3 points) As the deadline approaches, Govinda decreases the size of the training set to speed up training. Do you expect the approximation error to increase, decrease, or stay the same? What about the estimation error? Explain briefly.

(e) (2 points) Is the approximation error a random variable? What about the estimation error? Briefly explain what it means to say that a given quantity is a random variable.

(f) (3 points) Suppose efficiency isn't a concern anymore, but Govinda is now intrigued and would like to determine empirically how many features he should use in his classifier. He experiments with all g_i for $100 \leq i \leq 10000$; again, the feature set would be the i most common words of English. How should he choose the best value of i?

2. **Optimization.** Suppose our loss function $L(w)$ has a single global minimum w^*. Our goal is to start from an arbitrary w_0 and find w using gradient descent, with learning rate η. Let $w^{(t)}$ be our parameter vector after t gradient descent iterations, and let $\nabla_w f(w_t)$ be the gradient of f evaluated at w_t.

   (a) (3 points) Write out the expression for the update rule computing $w^{(t)}$ from $w^{(t-1)}$.

(b) (3 points) Is this procedure guaranteed to converge? If not, why?

(c) (3 points) If L is convex and this procedure converges, is it guaranteed to converge to w^*? If not, why?

Suppose we are now working with a large dataset and choose to use stochastic gradient descent instead (i.e. with a minibatch size of one sample point). Let $L_i(w)$ be the loss corresponding to a sample point X_i.

(d) (3 points) What is now the update rule that computes $w^{(t)}$ based on $w^{(t-1)}$?

(e) (3 points) We can interpolate between gradient descent and stochastic gradient descent by varying the size of the minibatch. Describe the tradeoff that is represented by this continuum.

3. **Regularization.**

(a) (5 points) Suppose you are trying to choose a good subset of features for a least-squares linear regression model. Let Algorithm A be forward stepwise selection, where we start with zero features and at each step we add the new feature that most decreases validation error, stopping only when validation error starts increasing. Let Algorithm B be similar, but at each step we include the new feature that most decreases training error (measured by the usual cost function, mean squared error), stopping only when training error starts increasing. What is the relationship between the number of features that the two algorithms will end up selecting?

Bingzhi is updating some legacy code. She decides to switch one of the data analysis procedures from ordinary least-square linear regression to ridge regression. Are the following statements true or false?

(b) (1 point) Training error can decrease.

A. True

B. False

(c) (1 point) Training error can increase.

A. True

B. False

(d) (1 point) Training error is likely to decrease.

A. True

B. False

(e) (1 point) Validation error can decrease.

A. True

B. False

(f) (1 point) Validation error can increase.

A. True

B. False

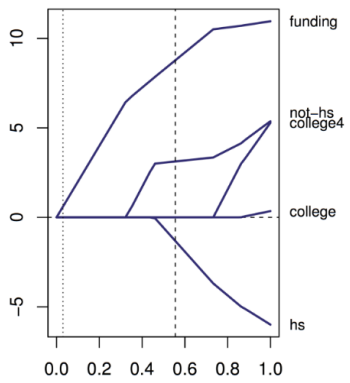
(g) (1 point) Validation error is likely to decrease.

A. True

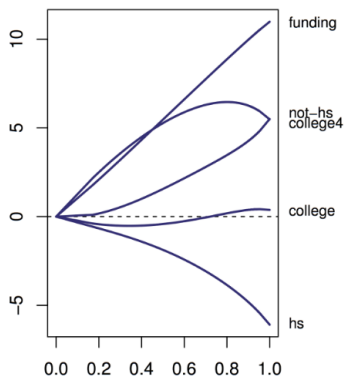
B. False

Compare the following regularization path plots:

(a)



(b)



(h) (4 points) Which plot corresponds to the lasso and which correponds to ridge regression? Explain your answer.

4. **Support Vector Machines.** Recall the (soft-margin) SVM primal problem:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}||w||^2 + \frac{c}{n}\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & -\xi_i \leq 0 \quad \text{for } i = 1, \ldots, n \\ & \left(1 - y_i\left[w^T x_i + b\right]\right) - \xi_i \leq 0 \quad \text{for } i = 1, \ldots, n \end{aligned}$$

and its dual problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_j^T x_i \\ \text{subject to} \quad & \sum_{i=1}^{n}\alpha_i y_i = 0 \\ & \alpha_i \in \left[0, \frac{c}{n}\right] \quad \text{for } i = 1, \ldots, n \end{aligned}$$

(a) (1 point) The primal objective is

A. convex

B. concave

C. neither convex nor concave

(b) (2 points) The dual objective is

A. convex

B. concave

C. neither convex nor concave

(c) (3 points) c is an important hyperparameter to tune when solving SVMs. If you find that the training error is too large, to reduce training error, you should

A. increase c

B. decrease c

C. do something else because c is irrelevant

(d) Recall that given the dual solution α_i^*'s, the primal solution is given by $w^* = \sum_{i=1}^{n}\alpha_i^* y_i x_i$, and the support vectors are defined to be x_i's where $\alpha_i > 0$. The figure below shows a toy dataset and the SVM decision boundary (the black line) with the corresponding margin (indicated by the two gray lines).

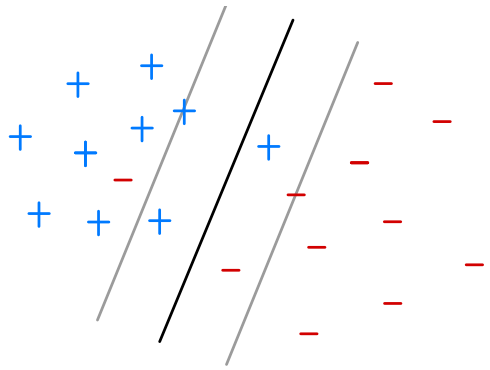
i. (2 points) For x_i "on the margin" (i.e. on the gray lines), which of the following is/are true about the corresponding ξ_i? (select all correct answers)

A. $\xi_i > 0$

B. $\xi_i = 0$

C. $\xi_i > 1$

D. $\xi_i = 1$



ii. (3 points) Circle all points that **must** be support vectors. (no partial credits)

(e) Assuming the data is not linearly separable, increasing c is likely to result in (circle the right answer)

i. (2 points) smaller / larger geometric margin

ii. (2 points) fewer / more support vectors

5. **Kernelized perceptron.** Recall the perceptron algorithm:

Initialize $w \leftarrow 0$

While not converged

For $(x_i, y_i) \in \mathcal{D}$

If $y_i w^T x_i \leq 0$, then

update $w \leftarrow w + y_i x_i$

return w

(a) (3 points) Show that the solution returned by the perceptron algorithm is in the span of the data, i.e. $w = \sum_{i=1}^{n} \alpha_i x_i$ where $\alpha_i \in \mathbb{R}$.

(b) (3 points) Given a weight vector w, to make an prediction for an input x we need to compute $w \cdot x$. Using the above result, express $w \cdot x$ in terms of α_i and inner products between x and the training examples x_i: $\langle x_i, x \rangle$.

(c) (5 points) During training, upon making a mistake on x_i, we update w by $w \leftarrow w + y_i x_i$. Give an equivalent update rule for α_i (so that we don't need to access w and x directly).

(d) (4 points) Suppose we use the RBF kernel for both SVM and perceptron. Which one is likely to have faster inference time and why? [Hint: For both algorithms, we need to compute $w \cdot x$ using inner products between the test example and the training examples. How many inner products do we need to compute for each?]

Congratulations! You have reached the end of the exam.