Homework Assignment 6

COGS 118A: Introduction to Machine Learning I

Due: 11:59pm, Sunday, December 2nd, 2018 (Pacific Time).

Instructions: Answer the questions below, attach your code, and insert figures to create a PDF file; submit your file via Gradescope. You may look up the information on the Internet, but you must write the final homework solutions by yourself.

Late Policy: 5% of the total points will be deducted on the first day past due. Every 10% of the total points will be deducted for every extra day past due.

Grade: ____ out of 100 points

1 (10 points) Multiple Choice

1. Which of the following statement(s) is/are **true** regarding the SVM classifier?

   A. The margin definition in the SVM formulation can be considered as a regularization term to prevent overfitting.

   B. Any function can be used a kernel function.

   C. Using a valid kernel, an SVM classifier can be trained without knowing the feature values for each sample.

   D. The so-called “support vectors” refer to the positive and negative planes.

2. Which of the following statement(s) is/are **true** regarding the decision tree classifier?

   A. When training a decision tree classifier, the depth of the tree goes linearly with respect to the number of training samples.

   B. The training objective function for a decision tree classifier has in general no analytic form to optimize for.

   C. Tree pruning can be used to prevent overfitting.

   D. In general, the deeper a decision tree is, the more complex the decision boundary is.

2 (10 points) Entropy and Conditional Entropy

Given a discrete random variable X with possible values $\{x_1, \dots, x_n\}$ and probability mass function $P(X)$, the formula to compute entropy is

$$H(X) = -\sum_{i}^{n} P(X = x_i) \ln P(X = x_i).$$

For Y with possible values $\{y_1, ..., y_m\}$, the conditional entropy of X conditioned on random variable Y is defined as

$$H(X|Y) = -\sum_{j=1}^{m}\sum_{i=1}^{n} P(X = x_i, Y = y_i) \ln \frac{P(X = x_i, Y = y_j)}{P(Y = y_j)}$$

The joint probability mass function of the random variables X and Y is given by the following table.

	$X = x_1$	$X = x_2$	$X = x_3$
$Y = y_1$	0.2	0.1	0.1
$Y = y_2$	0.3	0.2	0.1

1. Compute $H(X)$
2. Compute $H(X|Y)$
3. Compute $H(X|Y = y_1)$. Note that the entropy of X knowing $Y = y_1$ is defined as: $H(X|Y = y_1) = -\sum_i^n P(X = x_i|Y = y_1) \ln P(X = x_i|Y = y_1)$.

3 (15 points) Support Vector Machine

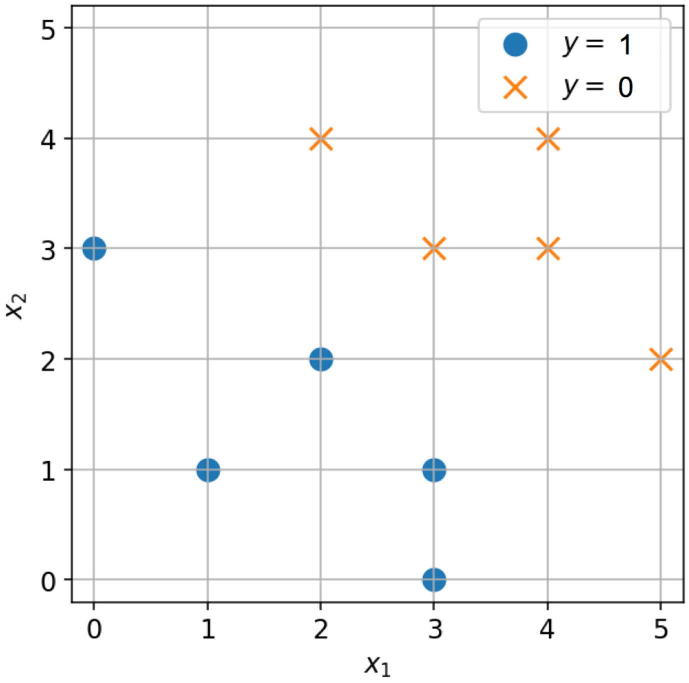
Consider a dataset $\{(\mathbf{x}^{(i)}, y^{(i)})\}$ where $\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}]^\top$ and $y^{(i)} \in \{0, 1\}$, which is shown in the figure below. Suppose we have trained a support vector machine (SVM) on the dataset, which has the **decision boundary** $\mathbf{w} \cdot \mathbf{x} + b = 0$, **positive plane** $\mathbf{w} \cdot \mathbf{x} + b = +1$ and **negative plane** $\mathbf{w} \cdot \mathbf{x} + b = -1$. The SVM is optimized as following:

$$\begin{aligned} \text{Find: } & \arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 \\ \text{Subject to: } & \mathbf{w} \cdot \mathbf{x}^{(i)} + b \geq +1, \text{ if } y^{(i)} = 1 \\ & \mathbf{w} \cdot \mathbf{x}^{(i)} + b \leq -1, \text{ if } y^{(i)} = 0. \end{aligned}$$

1) Please draw the **decision boundary**, **positive plane** and **negative plane** in the figure below.

2) Calculate the $\mathbf{w}$ and b from your drawn **positive plane** and **negative plane**.

3) Calculate the size of the margin.



4 (15 points) Ridge Regression

We are given a set of input training data $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$. Let $X = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)^\top$ be the entire input data matrix and $Y = (y_1, y_2, ..., y_n)^\top$ be the training labels. The objective function for the ridge regression is defined as follows:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \;\; b \times ||\mathbf{w}||^2 + \sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2$$

where

$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i.$$

Derive the closed form solution of $(\alpha_1, ..., \alpha_n)^\top$.

5 (20 points) Decision Tree

In this problem, you will implement decision tree algorithm to conduct the binary classification. We use the Ionosphere dataset that contains 351 data points. Each data point has a 34-dimensional feature vector and a binary label (either 0 or 1). Download the data `ionosphere.npy` from the course website. You may use `sklearn` for your implementation.

1) Load data from `ionosphere.npy` and shuffle the data points.

2) Randomly select 80% of the data points as your **training and validation set**. The rest 20% is regarded as your **test set**.

3) Train a decision tree. Use **entropy** criterion to measure the quality of a split. Use **5-fold** cross validation to select optimal maximum depth D from the set $\{1, 2, 3, 4, 5\}$.

4) Draw a heatmap for the result of grid search and find the optimal D that gives largest validation accuracy. Report the heatmap and optimal D.

5) Report test accuracy of trained decision tree with optimal D.

Hint: When trained properly, the test accuracy will be around 90%.

6 (30 points) k Nearest Neighbors

In this problem, you need to implement the k nearest neighbors (k-NN) and utilize it to conduct the binary classification. Here we still use the Ionosphere dataset. Please download the `ionosphere.npy` as data source and `HW6.ipynb` to fill the blanks. You are **NOT** allowed to use `sklearn.neighbors.KNeighborsClassifier()` in your code, but you can use it to validate your implementation.

1) Load data from `ionosphere.npy` and shuffle the data points.

2) Select 80% of the data points as your **training and validation set**. The rest 20% is regarded as your **test set**. Actually, in the cross-validation, the training and validation set can be called as "training set". However, in order to be consistent with the code, we still call it "training and validation set" here.

3) Implement the k-NN. For each feature vector to predict the label, you need to calculate the distances between **this feature vector** and **all the feature vectors in the training set**. Then sort all distances in ascending order and pick the labels for the k minimum distances. The mode of the k labels will be used as the predicted label for current feature vector. Here we assume **Euclidean distance** as the distance metric. For more details, please refer to the corresponding part in the slides.

4) Train the k-NN by implementing cross-validation to search for the optimal k. In k-NN, there is a parameter k which adjusts the number of nearest neighbors. You would need to use the grid search method to find the best parameter k^*. In fact, such grid search will utilize the cross-validation (3-fold) to get all the **average training accuracies** and **average validation accuracies** from the k-NN model with different parameter k on training and validation set. The parameter $k = k^*$ which maximizes the **average validation accuracy** will be selected as the best. In fact, here "average" means the average accuracy over the folds in cross-validation, not the average accuracy over the different parameter k.

Hint: You can perform grid search on the following list of k:

$$k \in \{1, 2, 3, 4, 5, 6\}$$

5) Draw heatmaps for the result of grid search and find the best k^* for average validation accuracy. Report the heatmaps and best k^*.

6) Use the the best k^* to train a k-NN on training and validation set. Then, use the trained classifier to calculate the accuracy on test set. Report the test accuracy.

7 (Bonus: 20 points) Support Vector Machine

In this problem, you need to implement the linear SVM using gradient descent by yourself. Same dataset in HW4 Q4 logistic regression (Iris dataset) is used here while only the range of $y^{(i)}$ is changed: $\{(\mathbf{x}^{(i)}, y^{(i)})\}$, $y^{(i)} \in \{-1, 1\}$ and $\mathbf{x}^{(i)} = [x_0^{(i)}, x_1^{(i)}, \ldots, x_K^{(i)}]^\top$ where $x_0 = 1$ is added as a bias. **To simplify the process, please only use first and third features to train your model.** Label -1 is used now for the data points originally labeled as 0. Your implementation of SVM should minimize the loss function:

$$\mathcal{L}(\theta) = ||\theta||^2 + \lambda \sum_i \max(0, 1 - y^{(i)} f(\mathbf{x}^{(i)}; \theta))$$

where $f(\mathbf{x}^{(i)}; \theta)) = \sum_{k=0}^{K} \theta_k x_k^{(i)}$ and $\lambda = 5$. You are **NOT** allowed to use `svm.SVC()` or any function that trains a SVM here. Train the linear SVM model and report the code with following results:

1) The optimal θ^*.

2) Training accuracy and test accuracy.

3) Plot of training data (first and third features) along with decision boundary.

4) Plot of test data (first and third features) along with decision boundary.

Note: You may need to change the learning rate, the number of iterations and the error threshold for θ in the code from HW4 Q4. As a bonus problem, we do not provide the details of the modification here.