CS 5350/6350: Machine Learining Spring 2015

Homework 5 Solution

Handed out: Mar 31, 2015
Due date: Apr 13, 2015

1 Kernel Perceptron

In the class, we have seen the idea of kernels applied to support vector machines. In this question, you will derive the kernel version of the Perceptron learner. For the purpose of this question, we will assume that instances are classified using a feature representation $\phi(x)$. That is, classification using Perceptron is the sign of the dot product of the weight vector and the example $w^T\phi(x)$. The goal is to, instead, represent this using a kernel $K(w, x)$.

Note that, with this feature representation, the perceptron update is $w \leftarrow w + ry\phi(x)$, whenever there is a mistake (i.e. $yw^T\phi(x) < 0$). For this problem you may assume the learning rate $r = 1$.

1. [10 points] Assume that the initial weight vector is the zero vector. Then, show that $w = \sum_{(x_i,y_i)\in M} y_i\,\phi(x_i)$. Here $M = \{(x_i, y_i)\}$ is the set of examples on which the learning algorithm made mistakes.

Solution:

Let M_k be the set of the first k mistakes the algorithm will make. Let w_k be the weight vector after the kth mistake. If w_0 is defined as 0 and the perceptron update gives $w_{k+1} = w_k + y_{k+1}\,\phi(x_{k+1})$ then w_k can be written as $w_k = \sum_{(x_i,y_i)\in M_k} y_i\,\phi(x_i)$. The final weight vector can be written as $w = \sum_{(x_i,y_i)\in M} y_i\,\phi(x_i)$.

2. [10 points] Let K be a kernel function defined as $K(u, v) = \phi(u)^T\phi(v)$.

   Using the fact that the weight vector can be written as in the previous question, write the classification step $y = sgn\left(w^T\phi(x)\right)$ using the kernel function $K(u, v)$ instead of using the explicit feature representation $\phi(u)$.

Solution:

Vector multiplication is distributive over addition, so $\phi(x)$ can be pulled inside the sum in w.

$$w^T\phi(x) = \sum_{(x_i,y_i)\in M} y_i\,\phi(x_i)^T\phi(x)$$

$$sgn\left(w^T\phi(x)\right) = sgn\left(\sum_{(x_i,y_i)\in M} y_i\, K(x_i,x)\right)$$

3. [5 points] In pseudo code, write the full kernel perceptron algorithm.

   (Hint: Instead of storing a weight vector, you will store a list of examples the algorithm made a mistake on.)

Solution:

```
define C(M, x) = sum(map((xi, yi) => yi * K(xi, x), M))

var mistakes = List.empty
for t in 1 to T
    for (xi, yi) in Data {
        if sgn(C(mistakes, xi)) != sgn(yi)
            mistakes += (xi, yi)
    }

define classify(x) = sgn(C(mistakes, x))
```

2 Naïve Bayes

Consider the Boolean function $f_{TH(3,7)}$. This is a threshold function defined on the 7 dimensional Boolean cube as follows: given an instance x, $f_{TH(3,7)}(x) = 1$ if and only if 3 or more of x's components are 1.

1. [5 points] Show that $f_{TH(3,7)}$ has a linear decision surface over the 7 dimensional Boolean cube.

Solution:

Any linear decision surface can be expressed by a weight vector and an offset as $w^T x \geq b$. The following inequality of this form captures the function.

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \geq 3$$

2. [10 points] Assume that you are given data sampled according to the uniform distribution over the Boolean cube $\{0,1\}^7$ and labeled according to $f_{TH(3,7)}$. Use naïve Bayes to learn a hypothesis that predicts these labels. What is the hypothesis generated by the naïve Bayes algorithm? (You do not have to implement the algorithm here. You may assume that you have seen all the data required to get accurate estimates of the probabilities).

Solution:

The prediction step is $\text{argmax}_y P(X=x|Y=y)P(Y=y)$. To compute this we the following set of probabilities.

Probability	Computation	Value
$P(y=1)$	$1-P(y=0)$	$\frac{99}{128}=0.773$
$P(y=0)$	$(1+\binom{7}{1}+\binom{7}{2})/2^7$	$\frac{29}{128}=0.227$
$P(x_i=1\|y=1)$	$(2^6-\binom{6}{0}-\binom{6}{1})/99$	$\frac{57}{99}=0.576$
$P(x_i=0\|y=1)$	$1-P(x_i=1\|y=1)$	$\frac{42}{99}=0.424$
$P(x_i=1\|y=0)$	$(\binom{6}{0}+\binom{6}{1})/29$	$\frac{7}{29}=0.241$
$P(x_i=0\|y=0)$	$1-P(x_i=1\|y=0)$	$\frac{22}{29}=0.759$

3. [5 points] Show that the hypothesis produced in (2) does not represent this function.

Solution:

For any input x with exactly two ones in it classification is performed as follows.

$$P(X=x|Y=1)P(Y=1)=(0.576)^2(0.424)^5(0.773)=0.0035$$

$$P(X=x|Y=0)P(Y=0)=(0.241)^2(0.759)^5(0.227)=0.0033$$

The probability is maximized for $Y=1$, but the correct classification is $Y=0$.

4. [5 points] Are the naïve Bayes assumptions satisfied by $f_{TH(3,7)}$? Justify your answer.

Solution:

The function is the *count* of the number of variables. This does not satisfy conditional independence for thresholds greater than 1.

3 Ensembles of decision trees

You will train on the new badges data available on the class web page.

1. Implement SVM using SGD as a training algorithm.

2. Using the provided features, run 10-fold cross validation to find the best values for the hyper-parameters ρ_0 and C. Try out all combinations of $\rho_0 \in \{0.001, 0.01, 0.1, 1\}$ and $C \in \{0.1, 1, 10, 100, 1000\}$. Feel free to expand the set of parameters if you have enough time.

   Show a table including the 5 best parameters with three columns: ρ_0, C and the average cross validation accuracy for that choice of hyper-parameters.

   (Since all you care about here is the relative accuracy to other training runs it is not necessary for the weight vector to converge. To make cross validation faster, only run 10 epochs of SGD.)

   Use the best value of ρ_0 and C to train a classifier on the entire training set. Run at least 30 epochs of SGD. Report the performance of this classifier on the test set.

3. Next, you will use the ID3 algorithm to train decision trees on sub-samples of the data. You can use your own implementation of decision trees from the earlier homework for this part. The one important change to the tree growing algorithm is a depth parameter. This is a number that restricts the tree from growing beyond a certain depth. If the depth of the tree reaches the limit, your algorithm should create a leaf with the majority label.

   From the full training set, randomly sample half the examples. Grow a decision tree of maximum depth 4 on this sample. This decision tree can now predict a $+$ or $-$ for any example.

   Train hundred different decision trees of maximum depth four on the entire training set. *Note: To get a hundred* **different** *decision stumps, you need to repeatedly sample 50% of the training set and train a decision tree on the sub-sample.*

   At this point, you should have 100 trees, each of which can predict $+$ or $-$ for any example. These trees will create your new feature set. Report the mean and variance of the accuracy of these trees on the test set.

   Create a new 100-dimensional dataset using the hundred decision trees as follows: For each example in the data set, the value of the i^{th} feature will be the prediction of the i^{th} decision tree. This will give you a new feature representation for the data. Using this new feature set, train a linear separator with the SGD algorithm and evaluate with 10-fold cross-validation as in part 2.

   Remember: Make sure that you only sample from the training set to generate the decision stumps, otherwise you might contaminate the training set with examples from the test set and this will skew your results.

4. Repeat this with depth limits of 8 and 20.

3.1 Evaluation and report

You should compare seven models: SGD for SVM on the original feature set, three depths of decision trees, and SVM over an ensemble of decision trees with depth limits four, eight and twenty. Of course, this is just the minimum. Feel free to experiment with more parameter combinations (decision tree depth, learning rate for the SGD and fraction of the data used to train the trees.) In each setting, you should use cross-validation to get the best SVM hyper-parameters.

Report the following for each case: The best cross validation accuracy and the test set performance. For decision trees show the variance instead of cross validation accuracy. Rank the different methods in terms of their performance. In the end, your conclusion will be that a particular algorithm (or set of algorithms) performed the best. Briefly state the assumptions that this conclusion is based on.

As mentioned in previous homeworks, you may use any programming language for your implementation. Upload your code along with a script so the TAs can run your solution in the CADE environment.

Solution:

Accuracies in this section depend, sometimes heavily, on random processes outside our control, so the values listed below are examples of a sane set of results. Cross validation was done with 10 epochs of SGD, and the reported accuracies were trained for 30 epochs using the best parameters from cross validation.

Algorithm	Accuracy	Variance	Cross Validation Accuracy	C	ρ_0
SVM Raw	0.689	-	0.609	1	1e-3
Decision Tree 4	0.758	0.231	-		
Ensemble 4	0.824	-	0.691	1	1e-5
Decision Tree 8	0.817	0.189	-		
Ensemble 8	0.811	-	0.645	1	1e-5
Decision Tree 20	0.845	0.082	-		
Ensemble 20	0.824	-	0.650	1	1e-2