Machine Learning Resit Exam

With Answers

11 June 2013

Question	*Max.*	*Pts*
Short answers	15	
Neural Nets and Gradient Descent	20	
Instance-Based Learning	14	
Combining Multiple Learners	15	
Comparison of ML algorithms	16	
Total	**80**	

The exam is **open book**: you can use Alpaydin's "Introduction to Machine Learning" (as a print of the PDF or as a proper book) as well as the lecture slides and any notes you've taken. You can use a calculator. The questions add up to *80 points* in total.

Good luck!

Answers in bold italics.

Questions

1. **Short answers, no justification required** *(3 Points for each question)*

   (a) (**True** or **False**): Pruning a decision tree reduces the likelihood that it fits noise in the training data.
   True

   (b) (**True** or **False**): Support Vector Machines can only classify linearly separable classes.
   False

   (c) (**True** or **False**): Adding hidden nodes in a multilayer perceptron increases the likelihood of overfitting.
   True

   (d) (**True** or **False**): Hierarchical clustering requires the user to define the expected number of clusters beforehand.
   False

(e) (**True** or **False**): Neural networks can only be used for regression, not for classification.

False

2. **Neural Nets and Gradient Descent**

(a) *(5 points)* In one extension of the basic gradient descent algorithm, the learning rate is decreased slowly while running the algorithm. Why would this be done?

By slowly letting the learning rate decrease as the algorithm runs, it is possible to ensure that gradient descent will converge to the minimum rather than oscillate around it.

(b) *(5 points)* When training perceptrons with gradient descent, one can add momentum: why and how does momentum help?

Momentum –as the name implies– adds momentum to the descent: the search doesn't stop at a minimum, but 'overshoots' it. This overshooting is hoped to get the descent out of local minima.

(c) *(5 points)* What would be the consequence of setting momentum too high? What would be the consequence of setting momentum too low?

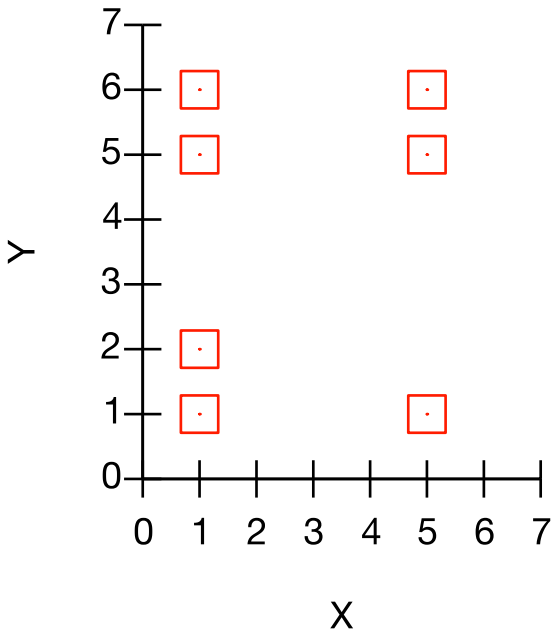
Too low: overshooting doesn't get us out of local minima; too high: we may overshoot the global minimum, possibly ending up in a local minimum further on, but at least it will take longer to converge as we go back and forth across the optimum (similar to setting learning rate too high).

(d) *(5 points)* Describe in one or two sentences a procedure to select the optimal value for momentum.

Any trial-and-error method that does not use the training set error to measure the effect of momentum will do. In class, we saw an example where n-fold cross validation was used.

3. **Instance-Based Learning**

The following picture shows a regression dataset with one real-valued input x and one real-valued output y. There are seven training points.



Suppose you are training using distance weighted nearest neighbour ("kernel regression" in the lecture) with some unspecified distance weighting (kernel) function. The only thing you know about the kernel function is that it is a monotonically decreasing function of distance that decays to zero at a distance of 3 units (and is strictly greater than zero at a distance of less than 3 units). *Note that the distance is in terms of the x value only.*

(a) *(3 points)* What is the predicted value of y when x = 1?
Answer: $\frac{1+2+5+6}{4} = 3.5$

(b) *(3 points)* What is the predicted value of y when x = 3?
Answer: $\frac{1+2+5+6+1+5+6}{7} = \frac{26}{7} \approx 3.71$

(c) *(3 points)* What is the predicted value of y when x = 4?
Answer: $\frac{1+5+6}{3} = 4$

(d) *(5 points)* What is the role of a kernel function and why does it allow instance-based learners to take all examples into account instead of only the k nearest neighbours?
The kernel function focuses the average around the instance that is queried: instances at increasing distance have decreasing influence. Thus, the neighbourhood size can become infite: the kernel will make sure that far-away instances have little or no influence.

4. **Combining Multiple Learners**

(a) *(5 points)* You are working with a considerably noisy dataset, in which classifiers tend to overfit on the training data. Which ensemble method would you propose to alleviate this problem? Why?

Bagging. Training a high number of classifiers on different samples of the training data each time means the noisy data instances will be less likely to be featured across datasets than instances correctly displaying the problem's features. Some classifiers may overfit, but their collective vote is likely to downplay the votes of overfit classifiers.

(b) *(5 points)* You want to use AdaBoost to improve the performance of classification of your dataset. You are going to use a Neural Network as the base learner. How big (in number of hidden layers and neurons) should your neural network be? Why?

We'd want a very small NN. Boosting works best with weak learners that focus at each step on the most salient features of the data currently under consideration, leaving the learning of the remaining features to other iterations.

(c) *(5 points)* Ensemble methods are primarily oriented towards the combination of multiple classifiers. Because label "A" is always label "A" across all classifiers in the ensemble, it is easy to carry out a vote and so combine classifiers. In the case of clustering, however, each clustering solution creates its own "labels".

Propose a way to support the creation of an ensemble of *clustering* solutions.

EM and k-Means each represent a cluster in a unique way (PDF, or centroid, respectively). A cluster's representation can provide the basis how to combine multiple, identical, such representations. In the case of EM, out of each PDF returned by an individual clusterer in the ensemble, we could produce the one that assigns greatest likelihood to the data instance under consideration. In k-Means, with each clusterer advocating a centroid for representing a data instance, the multiple answers could be combined though an averaging vote, that would return the centroid of all the advocated centroids.

5. **Comparison of ML algorithms**

Assume we have a set of data from thousands of patients who have visited VUmc hospital during the year 2012. A set of features (e.g., temperature, height) have been also extracted for each patient. Our

goal is to decide whether a new visiting patient has any of diabetes, heart disease, or Alzheimer (a patient can have one or more of these diseases).

(a) *(5 points)* Initially, we decide to try two ML algorithms to develop our classification models: naive Bayes and decision trees. Explain briefly how we should go about choosing between these methods to ensure that we obtain the best decisions.

Split data into training set and test set and use the test-set or use cross-validation to get an unbiased estimate of the methods' performances. Bonus point for those who mention that these performances should be compared using some statistical measure (e.g. a t-test).

(b) *(5 points)* Some patient features are expensive to collect (e.g., brain scans) whereas others are not (e.g., temperature). Therefore, we have decided to first ask our classification algorithm to predict whether a patient has a disease, and if the classifier is 80% confident that the patient has a disease, then we will do additional examinations to collect additional patient features. For this preliminary model, which classification methods do you recommend: neural networks, decision tree, or naive Bayes? Justify your answer in one or two sentences.

We expect students to explain how each of these learning techniques can be used to output a confidence value (any of these techniques can be modified to provide a confidence value). In addition, Naive Bayes is preferable to other cases since we can still use it for classification when the value of some of the features are unknown. Partial credits to those who mentioned neural network because of its non-linear decision boundary, or decision tree since it gives us an interpretable answer.

(c) In a different, much simpler case, we are dealing with samples x where x is a scalar value. We would like to test two alternative regression models:
1. $y = ax + c$
2. $y = ax + bx^2 + c$

   i. *(3 points)* Which of the two models is more likely to fit the *training* data better? Motivate your answer in 1-2 sentences.
      a. model 1
      b. model 2
      c. both will fit equally well
      d. impossible to tell

b. (model 2). Since it has more parameters it is likely to provide a better fit for the training data.

i. *(3 points)* Which of the two models is more likely to fit the *test* data better? Motivate your answer in 1-2 sentences.
   a. model 1
   b. model 2
   c. both will fit equally well
   d. impossible to tell

   d. It depends on the underlying model of the data and the amount of data available for training. If the data indeed comes from a linear model and we do not have a lot of data to train on model 2 will lead to overfitting and model 1 would do better. On the other hand if the data comes from an underlying quadratic model, model 2 would be better.