ECE M146 Introduction to Machine Learning

Prof. Lara Dolecek

ECE Department, UCLA

Today’s Lecture

Recap:

- Unsupervised Learning

New topic:

- Ensemble methods: bagging and boosting
- AdaBoost

Today's Lecture

Recap:

- Unsupervised Learning

New topic:

- Ensemble methods: bagging and boosting
- AdaBoost

Recap: Unsupervised learning

- Clustering:
  - K-means for hard membership assignment
  - EM for soft membership assignment

  - Both methods are iterative and they iterate between two steps: cluster membership and parameter refitting.

- PCA for dimensionality reduction
  - Formulate and solve as an optimization problem. Arrive at the dimension specified by the largest eigenvalue. Generalize using SVD.

Today's Lecture

Recap:

- Unsupervised Learning

New topic:

- Ensemble methods: bagging and boosting
- AdaBoost

Ensemble methods

- Ensemble of ML algorithms, e.g., classifiers, is a collection of these algorithms whose individual decisions are combined in some way such that the performance of the ensemble is better in the aggregate than that of one of its constituents.

- Let’s consider ensemble of classifiers.

- How to make classifiers non-identical?
- How to combine their results ?

Let’s see some choices

- For example, we can add randomness in different initial conditions, and take the average of these classifiers.
- We have touched upon this approach before.

- Other, principled choices are:

1) Bagging (parallel)
2) Boosting (serial)

Bagging

- Bagging stands for **bootstrap aggregation**

- Idea: train K classifiers in parallel, each on re-sampled data, and combine.

Bagging – how

- Suppose we have N training data points. Call this set $\mathcal{D}$.
- Create K new data sets by sampling with replacement from $\mathcal{D}$.
- Call each of these sets $\mathcal{D}_k$, for $1 \leq k \leq K$.

- It is entirely feasible that a given data point appears more than once in one of these sets; it is also entirely feasible that a given data point does not appear in one of these sets.
- What is also feasible is that a data point does not appear in any set!

Bagging – intuition

- What bagging does is it suppresses sensitivity to one specific data point, i.e., it reduces the impact of outliers and in turn the variance.

Example

- Suppose we have 11 binary classifiers, each with individual misclassification rate of 0.3.
- These are relatively weak classifiers.
- We are also going to take an idealistic assumption that they are independent.
- What is the overall misclassification probability, based on the majority vote?

Upshot

- All we needed was for individual classifiers to have own misclassification rate less than 0.5!
  - Which is, really, just being slightly better than random choice.

- Then, as the number of classifiers increases, the overall misclassification rate decreases to zero.

Boosting

- Boosting is a serial approach: base classifiers are trained in a sequential order.
- Each classifier is trained on weighted data, where the weighting factor of a given data point depends on the performance of the preceding classifier on that data point.
- Suppose we have N training data points. Call this set $\mathcal{D}$. In essence, we sequentially create sets $\mathcal{D}_k$, for $1 \leq k \leq K$, where each of these sets is the weighted version of the base set.
- More emphasis is given to more difficult points (i.e., misclassified points).

Today's Lecture

Recap:

- Unsupervised Learning

New topic:

- Ensemble methods: bagging and boosting
- AdaBoost

AdaBoost for Binary Classification

- AdaBoost = Adaptive Boosting.
- Suppose we have N data points x_n, $1 \leq n \leq N$, each with a label t_n in {-1,1}.
- Suppose we have K binary classifiers.

AdaBoost for Binary Classification

Initialization

- Initialize weights $w_n^{(1)} = 1/N$ i.e., each data point is initially given the same importance.

Fit K classifiers

- For $1 \leq k \leq K$, fit a classifier $y_k(x)$ by minimizing the function J_k

Make a prediction

- For a new data point x, compute:

How to fit an individual classifier ?

- We seek to minimize the function J_k.
- If the classifier is simple, can fit it exhaustively.
- For example, consider a decision tree that makes just one query.

Update of the weighting coefficients

- Recall that each classifier k has its own weighting coefficients, one per data point, that are derived based on the performance of the preceding classifier.

- First, capture accuracy:

- Second, scaling:

Update of the weighting coefficients

- Third, combine to get the weighting coefficients of the next classifier, and renormalize.

Illustrative example

- Consider the following:

- Clearly no linear classifier alone can perfectly separate this data.
- But, combining simple linear classifiers may work!

Illustrative example

- Step 1: Initialize the weights:
- Step 2: Fit the first classifier to minimize J_1.
- What are the choices ?

Illustrative example

- Fit the first classifier, ctd.
- At this point, the best we can do is to have one misclassified point.

- Compute error (capture accuracy):

- Compute scaling:

Illustrative example

- Step 2: Fit the first classifier to minimize J_1 (ctd)
- Compute weightings and then renormalize:
- This is how we build the first stage.

Illustrative example

- Stage 2: Fit the second classifier to minimize J_2.
- Compute accuracy:
- Compute scaling:

Illustrative example

- Currently, the overall classifier gives us:

Illustrative example

- Stage 2: Fit the second classifier to minimize J_2.

- Compute weightings:

Illustrative example

- Stage 2: Fit the third classifier to minimize J_3.

- Compute accuracy:
- Compute scaling:

Illustrative example

- Currently, the overall classifier gives us:

Comments

- Necessary conditions on the parameters

Mathematical justification

- The previous algorithm minimizes exponential loss
- Expression:

- Picture:

Set-up

- Consider labeled data:
- We are going to minimize this expression:
- We keep track of the following quantity:
- Can be interpreted as the running weighted average.

Sequential minimization

- The idea is to minimize E with respect to both scaling coefficients and classifiers. We are going to do this sequentially.
- Mathematical expansion:

Sequential minimization, ctd.

- Math, ctd.

Sequential minimization, ctd.

- Let T_m be the set of indices of points correctly classified by the classifier y_m.

- Let M_m be the set of indices incorrectly classified by the classifier y_m.

Sequential minimization, ctd.

- Decouple the overall error as follows:

Minimization with respect to the last base classifier

Minimization with respect to the scaling parameter

Minimization with respect to the scaling parameter

Minimization with respect to the scaling parameter

More on the scaling coefficients and weights

Bottom line

- AdaBoost can be interpreted as the minimization of the exponential error in a sequential minimization framework.

- Boosting is a very powerful technique. Boosted Decision Trees are some of the state of the art ML methods for large-scale data.