Introduction to Machine Learning (67577)

Exercise 3
PAC, Regularization, Ensemble Methods & Cross Validation

Second Semester, 2024

Contents

1 Submission Instructions

Please make sure to follow the general submission instructions available on the course website. In addition, for the following assignment, submit a single `ex3_ID.tar` file containing:

- An `Answers.pdf` file with the answers for all theoretical and practical questions (include plotted graphs *in* the PDF file).
- The following python files (without any directories): `decision_stump.py`, `adaboost.py`, `adaboost_scenario.py`, `loss_functions.py`

The `ex3_ID.tar` file must be submitted in the designated Moodle activity prior to the date specified *in the activity*.

- Late submissions will not be accepted and result in a zero mark.
- Plots included as separate files will be considered as not provided.

2 Theoretical Part

Based on Lectures 3,4 and Recitation 6

2.1 PAC Learnability

1. Let $\mathcal{X} := \mathbb{R}^2$, $\mathcal{Y} := \{0,1\}$ and let $\mathcal{H}$ be the class of concentric circles in the plane, i.e.,

$$\mathcal{H} := \{h_r : r \in \mathbb{R}_+\} \quad \text{where} \quad h_r(\mathbf{x}) = \mathbb{1}_{[||\mathbf{x}||_2 \leq r]}$$

Prove that $\mathcal{H}$ is PAC-learnable and its sample complexity is bounded by

$$m_{\mathcal{H}}(\varepsilon, \delta) \leq \frac{\log(1/\delta)}{\varepsilon}$$

When proving, *do not* use a VC-Dimension argument. Instead, prove the claim directly from the PAC learnability definition by showing a specific algorithm and analyzing its sample complexity. Derive the sample complexity explicitly.
Hint: Remember that for every $\varepsilon > 0$ it holds that $1 - \varepsilon \leq e^{-\varepsilon}$
Hint 2: The main idea is similar to section 1.2 in recitation 6

2.2 VC-Dimension

2. Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two classes for binary classification, such that $\mathcal{H}_1 \subseteq \mathcal{H}_2$. Show that $VC-dim(\mathcal{H}_1) \leq VC-dim(\mathcal{H}_2)$.
3. Let $\mathcal{X} = \{0,1\}^d$ and $\mathcal{Y} = \{0,1\}$, and assume $d \geq 2$. Each sample $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ consists of an assignment to d boolean variables ($\mathbf{x}$) and a label (y). For each boolean variable x_k, $k \in [d]$, there are two literals: x_k and $\bar{x}_k = 1 - x_k$. The class $\mathcal{H}_{\text{con}}$ is defined by boolean conjunctions over any subset of these $2d$ literals. Compute the VC dimension of $\mathcal{H}_{\text{con}}$ and prove your answer.
Hint: Let's start by understanding how Boolean Conjunctions class work. For example: let $d = 5$ and consider the hypothesis that labels $\mathbf{x}$ according to the following conjunction

$$x_1 \wedge x_2 \wedge \bar{x}_3$$

For $\mathbf{x} = (0,1,1,1,1)$ the label would be 0, and for $\mathbf{x} = (1,1,0,0,0)$ the label would be 1. The conjunction over the empty set is defined as the 'all positive hypothesis' ($h^+(\mathbf{x}) = 1$ for all $\mathbf{x}$), and any conjunction that contains $x_k \wedge \bar{x}_k$ for some k gives the 'all negative hypothesis' ($h^-(\mathbf{x}) = 0$ for all $\mathbf{x}$).

2.3 Sample Complexity

4. Let $\mathcal{H}$ be a hypothesis class for a binary classification task. Suppose that $\mathcal{H}$ is PAC learnable and its sample complexity is given by $m_{\mathcal{H}}(\cdot,\cdot)$. Show that $m_{\mathcal{H}}$ is monotonically non-increasing in each of its parameters. That is:
   - Show that given $\delta \in (0,1)$, and given $0 < \varepsilon_1 \leq \varepsilon_2 < 1$, we have that $m_{\mathcal{H}}(\varepsilon_1, \delta) \geq m_{\mathcal{H}}(\varepsilon_2, \delta)$.
   - Similarly, show that given $\varepsilon \in (0,1)$, and given $0 < \delta_1 \leq \delta_2 < 1$, we have that $m_{\mathcal{H}}(\varepsilon, \delta_1) \geq m_{\mathcal{H}}(\varepsilon, \delta_2)$.

2.4 Agnostic-PAC

5. Let $\mathcal{H}$ be a hypothesis class over a domain $\mathcal{Z} = \mathcal{X} \times \{\pm 1\}$, and consider the 0-1 loss function. Assume that there exists a function $m_{\mathcal{H}}$, for which it holds that for every distribution $\mathcal{D}$ over $\mathcal{Z}$ there is an algorithm $\mathcal{A}$ with the following property: when running $\mathcal{A}$ on $m \geq m_{\mathcal{H}}$ i.i.d. examples drawn from $\mathcal{D}$, it is guaranteed to return, with probability at least $1-\delta$, a hypothesis $h_S : \mathcal{X} \to \{\pm 1\}$ with $L_{\mathcal{D}}(h_S) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \varepsilon$. Is $\mathcal{H}$ agnostic PAC learnable? Prove or show a counter example.

3 Practical Part

You will implement a decision stump (decision tree of depth 1) and the AdaBoost algorithm.

3.1 Boosting

Based on Lecture 5 and Recitation 7

Implement the following according to their documentation:

- `misclassification_error` function in `loss_functions.py`
- `decision_stump.py`
- `adaboost.py`
- Write code that runs the experiments detailed below in `adaboost_scenario.py`

Note:

- `decision_stump.py` class selects a feature of the design matrix X and a corresponding threshold value that achieve the best classification result when thresholding only with regard to that feature.
- `adaboost.py` class is parameterized by a weak learner ("wl" like decision stump above) and the number of weak learners to use ("iterations"). Its solution is described by a list of fitted weak learner models ("models_"), their weights ("weights_"), and the derived distribution over samples ("D_", enhanced for samples we are wrong about). Note that fitting a classifier on samples with $\mathcal{D}$ distribution is equivalent to weighing each sample's label y_i by the its probability $\mathcal{D}_i$. Also, notice that `predict`/`loss` are specific cases of `partial_predict`/`partial_loss` that return prediction/loss evaluation when using all learners.

Packages:

- You can use the packages `numpy, plotly, matplotlib, itertools`
- You can use functions imported in the provided code
- Usage of any other package is not allowed!

Then, answer the following in `adaboost_scenario.py`:

1. Use the provided `generate_data` function to generate 5000 train samples and 500 test samples, with no noise (i.e. `noise_ratio = 0`). Train an Adaboost ensemble of size 250 (i.e passing 250 as the number of iterations) using your implementation of the `DecisionStump` as weak learner. Plot, in a single figure, the training- and test errors as a function of the number of fitted learners (i.e. use the `partial_loss` function for $t = 1, \ldots, 250$). Explain your results.

2. Using the previously fitted ensemble, plot the decision boundary obtained by using the the ensemble up to iteration 5, 50, 100 and 250. Use your implementation of the `partial_predict` to obtain the predictions of the ensemble up to the specified size. In each of these plots also add the test set (colored and/or shaped by the actual labels). Explain your results.

   You can use the `decision_surface` function in `utils.py`. Revisit lab 05 of comparing

classifiers to see how to create such plots.

3. Using the test evaluation scores per number of learners, which ensemble size achieved the lowest test error? Plot the decision surface of that ensemble as well as the test set data points. In the plot's title provide the ensemble size and its accuracy.

4. Using the previously fitted ensemble, use the weights of the last iteration (i.e. $\mathcal{D}$ at T) to plot the *training* set with a point size proportional to its weight and color (and/or shape) indicating its label.
   - As previously, plot the decision surface (using the full ensemble).
   - As the weights are of very small numbers normalize and transform as follow: `D = D/np.max(D) * 5`

   Explain your results, and specifically explain which samples are "easier" and which are more "challenging" for the classifier.

5. Repeat the steps above (while avoiding code repetition) for train- and test sets generated with noise levels of 0.4. Show graphs as in question (1) and question (4). Explain the results. In your answer explain what is seen in the plot of the loss in terms of the bias-variance tradeoff.