Introduction to Machine Learning (67577)

Exercise 2
PAC, Regularization, Ensemble Methods & Cross Validation

Second Semester, 2025

Contents

1 Theoretical Part

Based on Lectures 2,3 and Recitations 3,4

1.1 Hard- & Soft-SVM

In class we saw the Hard-SVM classification model.

1. Prove that the following Hard-SVM optimization problem is a Quadratic Programming problem:

$$\underset{(\mathbf{w},b)}{\operatorname{argmin}} ||\mathbf{w}||^2 \quad \text{s.t.} \quad \forall i\; y_i\left(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\right) \geq 1 \tag{1}$$

That is, find matrices Q and A and vectors $\mathbf{a}$ and $\mathbf{d}$ such that the above problem can be written in the following format

$$\underset{\mathbf{v}\in\mathbb{R}^n}{\operatorname{argmin}} \tfrac{1}{2}\mathbf{v}^\top Q\mathbf{v} + \mathbf{a}^\top\mathbf{v} \quad \text{s.t.} \quad A\mathbf{v} \leq \mathbf{d} \tag{2}$$

Hint: Observe that $||\mathbf{w}||^2 = \mathbf{w}^\top\mathbf{I}\mathbf{w}$

1.2 PAC Learnability

1. In this question, we will prove the following:

Theorem 1.1 — All finite hypothesis classes are PAC-Learnable. $\forall\mathcal{H}$ such that $|\mathcal{H}| < \infty$ is a finite hypothesis class, $\exists\mathcal{A}$ an algorithm such that $\forall f \in \mathcal{H}$ and $\forall\mathcal{D}$, $\forall\varepsilon,\delta \in (0,1)^2$, $\forall m \geq \frac{\log(|\mathcal{H}|)+\log(\frac{1}{\delta})}{\varepsilon}$,

$$\mathbb{P}_{\mathcal{D}}(L_{\mathcal{D},f}(\mathcal{A}(S_m)) < \varepsilon) > 1-\delta$$

(a) Prove the following lemma: Let D be a distribution over $\mathcal{X}$. Let S_m be m i.i.d. samples over $\mathcal{X}$. Then for any $\mathcal{X}' \subseteq \mathcal{X}$ such that $\mathbb{P}_D[X \in \mathcal{X}'] > \varepsilon$, $\mathbb{P}_{\mathcal{D}}[S_m \cap \mathcal{X}' = \emptyset] \leq e^{-m\varepsilon}$.
Hint: Use Bernoulli's Inequality

(b) Let S be some training sample with m samples. Recall that we are in the realizable setup, meaning we can choose $\hat{f}_S$ such that

$$L_{D,f}(\hat{f}_S, S) = 0$$

(i.e. we can choose $\hat{f}_S(x)$ which makes no mistakes over the training set S).
Let $\mathcal{A}$ be an algorithm that chooses such function. Bound the probability

$$\mathbb{P}_D[L_{D,f}(\hat{f}_S) > \varepsilon]$$

using the above lemma.
Hint 1: define a set $Z_{\hat{f}_S}$ the set of all the cases where $\hat{f}_S$ is wrong, How can you express the event $\{L_{D,f}(\hat{f}_S) > \varepsilon\}$ using this set?

Hint 2: Since $\hat{f}_S$ makes no mistakes on the training set S, what can you say about the intersection between the training set S and the set $Z_{\hat{f}_S}$ (the set where $\hat{f}_S$ is wrong)? How can you then use the lemma?

(c) Given the above, obtain $\mathbb{P}_{\mathcal{D}}(L_{\mathcal{D},f}(\mathcal{A}(S_m)) < \varepsilon) > 1-\delta$.
Hint: Union bound

2. 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.
Hint: Remember that for every $\varepsilon > 0$ it holds that $1-\varepsilon \leq e^{-\varepsilon}$

3. Prove that if $\mathcal{H}$ has the uniform convergence property with function $m_{\mathcal{H}}^{UC} : (0,1)^2 \to \mathbb{N}$, then $\mathcal{H}$ is Agnostic-PAC learnable with sample complexity $m_{\mathcal{H}}(\varepsilon,\delta) \leq m^{UC}(\varepsilon/2,\delta)$.

Definition 1.1 — Uniform Convergence of Function Sequences. A sequence of functions $f_n : X \to \mathbb{R}$ converges uniformly to $f : X \to \mathbb{R}$ if and only if

$$\forall \varepsilon > 0 \quad \exists m_0 \in \mathbb{N}$$

such that

$$\forall x \in X \quad |f_n(x) - f(x)| < \varepsilon$$

Definition 1.2 — ε-representative. A training sample S is called ε-representative for $\mathcal{D},\mathcal{H},\ell$ if and only if

$$\forall h \in \mathcal{H} \quad |L_S(h) - L_{\mathcal{D}}(h)| < \varepsilon$$

Definition 1.3 — Uniform Convergence Property. A hypothesis class $\mathcal{H}$ is said to have the uniform convergence property if and only if there exists a function $m_{\mathcal{H}}^{UC} : (0,1)^2 \to \mathbb{N}$ such that for every $\varepsilon,\delta \in (0,1)$ and every distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$

$$\mathcal{D}^m\left(\{S \in (\mathcal{X}\times\mathcal{Y})^m \mid S \text{ is } \varepsilon\text{-representative}\}\right) \geq 1-\delta$$

1.3 VC-Dimension

1. Let $\mathcal{X} = \{0,1\}^n$ and $\mathcal{Y} = \{0,1\}$, for each $I \subseteq [n]$ define the parity function:

$$h_I(\mathbf{x}) = \left(\sum_{i\in I} x_i\right) \bmod 2.$$

What is the VC-dimension of the class $\mathcal{H}_{parity} = \{h_I \mid I \subseteq [n]\}$? Prove your answer.
Hint: what is the size of the hypothesis class?

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 $VC-dim(\mathcal{H}_1) \leq VC-dim(\mathcal{H}_2)$.

2 Practical Part

In this exercise, you will explore and visualize various classification methods. The answers and plots will be in submitted in `Answers.pdf` file, and the code will be submitted in `solution.py`.

Implementation details:

- You may use the packages `scikit-learn`, `matplotlib`, `numpy`, `seaborn` (Optional). No other packages are allowed.
- Running the functions `practical_1_runner`, `practical_2_runner` should generate all the plots for the questions in sections 3.1, 3.2 respectively.
- These functions accept a single argument: a path. If it's None, the plots will be shown using `plt.show()` else they will be saved in the path. You may not change the signature of the functions.
- You must keep the function signature unchanged, but you may add helper functions if needed.

2.1 Support Vector Machines (SVM)

Data Generation: You will generate a dataset based on the Gaussian distribution with mean $(0,0)$ and covariance matrix

$$\begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

You will assign each sample with a label according to the function

$$f(x) = \text{sign}\big(\langle x, [-0.6, 0.4] \rangle\big)$$

Classification: For each $m \in \{5, 10, 20, 100\}$ and regularization $C \in \{0.1, 1, 5, 10, 100\}$, run a soft-SVM classification experiment over m samples with coefficient C.

Questions

1. Provide the plots for each of m, C values listed above. In each plot, distinguish the true labels using different colors, and plot both the decision boundary created by f and the SVM hypothesis.
2. Describe how changing C impacts the results.
3. Describe how changing m impacts the results in terms of PAC learning. Use terms of sample complexity m_H and accuracy ε.

2.2 Classification Boundaries and Model Comparison

Data Generation: Generate 200 samples of the following datasets:

- **Moons:** Generate moons dataset using `scikit-learn`'s function `make_moons`. Set `noise=0.2`.
- **Circles**: Generate circle dataset using `scikit-learn`'s function `make_circles`. Set `noise=0.1`.
- **Two Guassians**: Generate two Gaussian distributions with means $(-1,-1), (1,1)$ and covariance matrix

$$\begin{bmatrix} 0.5 & 0.2 \\ 0.2 & 0.5 \end{bmatrix}$$

- Split the datasets to train (0.8) and test (0.2)

Classifiers: Apply the following classifiers over these train datasets:

1. SVM with $\lambda = 5$
2. Decision Tree with depth 7
3. KNN with $k = 5$

Boundary Plots: Visualize the data points, using distinct colors to represent different labels. Additionally, display the decision boundary of a classifier by coloring the background. For each coordinate on the plot, color it according to the class the classifier would predict, using slightly lighter shades of the corresponding data point colors.

Questions:

1. For each classifier and each dataset, provide a decision boundary plot. Include the accuracy over the test dataset and model in the title of each plot.
2. Write a comparison of how the classifiers perform on different types of data distributions, and explain why each classifier is or is not a good fit for the dataset.
3. Explain how each algorithm shapes the decision boundary.