

Contents

1 adaboost.py	2
2 ex4 tools.py	5
3 ex4merge.pdf	7

1 adaboost.py

```
1  """
2 =====
3   Introduction to Machine Learning (67577)
4 =====
5
6 Skeleton for the AdaBoost classifier.
7
8 """
9 import numpy as np
10 from ex4_tools import *
11
12
13 def train_and_test(train_samples_num, T, test_samples_num, noise_factor):
14     # q13 -----
15     x_train, y_train = generate_data(train_samples_num, noise_factor)
16     x_test, y_test = generate_data(test_samples_num, noise_factor)
17
18     adabost = AdaBoost(DecisionStump, T)
19     weight_d = adabost.train(x_train, y_train)
20     training_error, test_error = [], []
21
22     for t in range(1, T+1):
23         training_error.append(adabost.error(x_train, y_train, t))
24
25         test_error.append(adabost.error(x_test, y_test, t))
26         # print("train error " , training_error)
27         #
28         # print("test error " , test_error)
29
30
31     domain_x = np.arange(0, T)
32     plt.plot(domain_x, training_error, color="orange")
33     plt.plot(domain_x, test_error, color="blue")
34     # plt.plot(m, sum_mean_acc, color="green")
35     plt.title("train and test error difference with noise ratio = " + str(noise_factor))
36     plt.legend(["training error", "test error"], bbox_to_anchor=(1.05, 1), loc='upper left',
37                borderaxespad=0.)
38     plt.tight_layout()
39     plt.show()
40     # print("returnfro tst and test")
41
42     # q14 -----
43     t_set = [5, 10, 50, 100, 200, 500]
44     for plot_index, t in enumerate(t_set):
45         plt.subplot(2, 3, plot_index + 1)
46         decision_boundaries(adabost, x_test, y_test, t)
47     plt.show()
48
49     # q15 -----
50     arg_min_index = np.argmin(test_error)
51     print("test error list: ", test_error)
52     print("min index: ", arg_min_index, "arg min value = ", test_error[arg_min_index])
53     decision_boundaries(adabost, x_train, y_train, arg_min_index + 1)
54     plt.show()
55
56     # q16 -----
57     decision_boundaries(adabost, x_train, y_train, T, weight_d)
58     plt.show()
59     print("i6 start")
```

```

60     # print(weight_d)
61     # print("----")
62     weight_d = weight_d / np.max(weight_d) * 10
63     # print(weight_d)
64     decision_boundaries(adaboost, x_train, y_train, T, weight_d)
65     plt.show()
66     print("16 end")
67     return
68
69
70
71
72 class AdaBoost(object):
73
74     def __init__(self, WL, T):
75         """
76             Parameters
77             -----
78             WL : the class of the base weak learner
79             T : the number of base learners to learn
80         """
81         self.WL = WL
82         self.T = T
83         self.h = [None]*T      # list of base learners
84         self.w = np.zeros(T)   # weights
85
86     def train(self, X, y):
87         """
88             Parameters
89             -----
90             X : samples, shape=(num_samples, num_features)
91             y : labels, shape=(num_samples)
92             Train this classifier over the sample (X,y)
93             After finish the training return the weights of the samples in the last iteration.
94         """
95         # TODO complete this function
96         print(X.shape[0])
97         weighted_distirbut = np.ones(X.shape[0]) / X.shape[0]  # D1 = (1/m, ..., 1/m)
98         weak_learner = self.WL
99         # print("iteration 0: wd = ", weighted_distirbut)
100        for t in range(0, self.T):
101            # print("-----")
102            ht = weak_learner(weighted_distirbut, X, y)
103            ht_predict = ht.predict(X)
104            epsilon_t = np.sum(weighted_distirbut[(y != ht_predict)])
105            # print("epslit t =", epsilon_t)
106            wt = (0.5) * np.log((1/epsilon_t) - 1)
107            numerator = weighted_distirbut * np.exp(-wt * y * ht_predict)
108            # next_distirbut = (weighted_distirbut * np.exp(-wt * y * ht_predict)) / np.sum(weighted_distirbut * np.exp(-wt *
109            # weighted_distirbut = next_distirbut
110            weighted_distirbut = numerator / numerator.sum()
111            self.h[t] = ht
112            self.w[t] = wt
113        return weighted_distirbut
114
115    def predict(self, X, max_t):
116        """
117            Parameters
118            -----
119            X : samples, shape=(num_samples, num_features)
120            :param max_t: integer < self.T: the number of classifiers to use for the classification
121            :return: y_hat : a prediction vector for X. shape=(num_samples)
122            Predict only with max_t weak learners,
123        """
124        # TODO complete this function
125        y_predict = np.zeros((X.shape[0]))
126        for t in range(0, max_t):
127            y_predict += (self.w[t] * self.h[t].predict(X))

```

```

128     return np.sign(y_predict)
129
130     def error(self, X, y, max_t):
131         """
132             Parameters
133             -----
134             X : samples, shape=(num_samples, num_features)
135             y : labels, shape=(num_samples)
136             :param max_t: integer < self.T: the number of classifiers to use for the classification
137             :return: error : the ratio of the wrong predictions when predict only with max_t weak learners (float)
138             """
139             # TODO complete this function
140             y_predict = self.predict(X, max_t)
141             return (y_predict[(y_predict != y)].shape[0]) / X.shape[0]
142
143
144     if __name__ == '__main__':
145         # train_and_test(5000, 500, 200, 0)
146         # train_and_test(5000, 500, 200, 0.01)
147         train_and_test(5000, 500, 200, 0.4)

```

2 ex4 tools.py

```
1  """
2 =====
3   Introduction to Machine Learning (67577)
4 =====
5
6 This module provides some useful tools for Ex4.
7
8 """
9
10 import numpy as np
11 import matplotlib.pyplot as plt
12 from matplotlib.colors import ListedColormap
13 from itertools import product
14 from matplotlib.pyplot import imread
15 import os
16 from sklearn.model_selection import train_test_split
17
18
19 def find_threshold(D, X, y, sign, j):
20     """
21     Finds the best threshold.
22     D = distribution
23     S = (X, y) the data
24     """
25     # sort the data so that x1 <= x2 <= ... <= xm
26     sort_idx = np.argsort(X[:, j])
27     X, y, D = X[sort_idx], y[sort_idx], D[sort_idx]
28
29     thetas = np.concatenate([[-np.inf], (X[:, j] + X[:-1, j]) / 2, [np.inf]])
30     minimal_theta_loss = np.sum(D[y == sign]) # loss of the smallest possible theta
31     losses = np.append(minimal_theta_loss, minimal_theta_loss - np.cumsum(D * (y * sign)))
32     min_loss_idx = np.argmin(losses)
33     return losses[min_loss_idx], thetas[min_loss_idx]
34
35
36 class DecisionStump(object):
37     """
38     Decision stump classifier for 2D samples
39     """
40
41     def __init__(self, D, X, y):
42         self.theta = 0
43         self.j = 0
44         self.sign = 0
45         self.train(D, X, y)
46
47     def train(self, D, X, y):
48         """
49             Train the classifier over the sample (X,y) w.r.t. the weights D over X
50             Parameters
51             -----
52             D : weights over the sample
53             X : samples, shape=(num_samples, num_features)
54             y : labels, shape=(num_samples)
55             """
56         loss_star, theta_star = np.inf, np.inf
57         for sign, j in product([-1, 1], range(X.shape[1])):
58             loss, theta = find_threshold(D, X, y, sign, j)
59             if loss < loss_star:
```

```

60         self.sign, self.theta, self.j = sign, theta, j
61     loss_star = loss
62
63     def predict(self, X):
64         """
65             Parameters
66             -----
67             X : shape=(num_samples, num_features)
68             Returns
69             -----
70             y_hat : a prediction vector for X shape=(num_samples)
71             """
72     y_hat = self.sign * ((X[:, self.j] <= self.theta) * 2 - 1)
73     return y_hat
74
75
76     def decision_boundaries(classifier, X, y, num_classifiers=1, weights=None):
77         """
78             Plot the decision boundaries of a binary classifiers over X \subsetneq R^2
79
80             Parameters
81             -----
82             classifier : a binary classifier, implements classifier.predict(X)
83             X : samples, shape=(num_samples, 2)
84             y : labels, shape=(num_samples)
85             title_str : optional title
86             weights : weights for plotting X
87             """
88     cm = ListedColormap(['#AAAAFF', '#FFAAAA'])
89     cm_bright = ListedColormap(['#0000FF', '#FF0000'])
90     h = .003 # step size in the mesh
91     # Plot the decision boundary.
92     x_min, x_max = X[:, 0].min() - .2, X[:, 0].max() + .2
93     y_min, y_max = X[:, 1].min() - .2, X[:, 1].max() + .2
94     xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
95     Z = classifier.predict(np.c_[xx.ravel(), yy.ravel()])
96     Z = Z.reshape(xx.shape)
97     plt.pcolormesh(xx, yy, Z, cmap=cm)
98     # Plot also the training points
99     if weights is not None:
100         plt.scatter(X[:, 0], X[:, 1], c=y, s=weights, cmap=cm_bright)
101     else:
102         plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cm_bright)
103     plt.xlim(xx.min(), xx.max())
104     plt.ylim(yy.min(), yy.max())
105     plt.xticks([])
106     plt.yticks([])
107     plt.title(f'num classifiers = {num_classifiers}')
108     plt.draw()
109
110
111
112     def generate_data(num_samples, noise_ratio):
113         """
114             generate samples X with shape: (num_samples, 2) and labels y with shape (num_samples).
115             num_samples: the number of samples to generate
116             noise_ratio: invert the label for this ratio of the samples
117             """
118     X = np.random.rand(num_samples, 2) * 2 - 1
119     radius = 0.5 ** 2
120     in_circle = np.sum(X ** 2, axis=1) < radius
121     y = np.ones(num_samples)
122     y[in_circle] = -1
123     y[np.random.choice(num_samples, int(noise_ratio * num_samples))] *= -1
124
125     return X, y
126

```

IML ex4

(1) Ac. Learnability

• $\exists \delta' \in (0, \infty)$, $\forall N \in \mathbb{N}$, $\Pr_{S \sim D}[\text{err}(A(S)) > \delta'] \leq \epsilon$

$$(a) \quad \Pr_{S \sim D} [\text{err}(A(S)) > \delta] \leq \epsilon$$

$$\Pr_{S \sim D} [L_0(A(S)) \geq \delta] \geq 1 - \epsilon$$

$$(b) \quad \lim_{n \rightarrow \infty} \mathbb{E}_{S \sim D} [L_0(A(S))] = 0$$

$$\Rightarrow \lim_{n \rightarrow \infty} \mathbb{E}_{S \sim D} [L_0(A(S))] = 0 \Leftarrow \forall \epsilon > 0 : \mathbb{P}[L_0(A(S)) \geq \epsilon] \leq \epsilon$$

$$(c) \quad \lim_{n \rightarrow \infty} \mathbb{E}_{S \sim D} [L_0(A(S))] = 0 \Leftarrow \forall \epsilon > 0 : \mathbb{P}[L_0(A(S)) > \epsilon] \leq \epsilon$$

$$n = m(\epsilon) \quad \epsilon = \frac{\delta}{2m} \quad \Pr_{S \sim D} [L_0(A(S)) > \epsilon] \leq \Pr_{S \sim D} [L_0(A(S)) > \frac{\delta}{2}]$$

$$\Pr_{S \sim D} [L_0(A(S)) > \epsilon] \leq \Pr_{S \sim D} \left[\frac{L_0(A(S))}{\epsilon} \leq \frac{\delta}{\epsilon} \right] \leq \Pr_{S \sim D} [L_0(A(S)) \geq \frac{\delta}{2}]$$

$$\Pr_{S \sim D} [L_0(A(S)) > \epsilon] \leq \Pr_{S \sim D} [L_0(A(S)) \geq \frac{\delta}{2}] \leq 1 - \epsilon$$

$$(d) \Rightarrow \forall \epsilon > 0 : \Pr_{S \sim D} [L_0(A(S)) \geq \epsilon] \leq \epsilon$$

$$\text{def: } n = m_K(\epsilon) \geq \Pr_{S \sim D} [L_0(A(S)) \geq \epsilon] \leq \epsilon$$

$$\text{def: } X = L_0(A(S)) \quad \text{def: } \epsilon_a = \epsilon_a(\epsilon) = \frac{\epsilon}{2m_K(\epsilon)} \quad \Pr_{S \sim D} [X \geq \epsilon_a] \leq \epsilon$$

$$\mathbb{E}[X] = \int_0^{\epsilon_a} x f(x) dx = \int_0^{\epsilon_a} x f(x) dx + \int_{\epsilon_a}^{\epsilon} x f(x) dx \leq$$

$$\leq \int_0^{\epsilon_a} \epsilon_a f(x) dx + \int_{\epsilon_a}^{\epsilon} \epsilon_a f(x) dx = \epsilon_a \int_0^{\epsilon_a} f(x) dx + \int_{\epsilon_a}^{\epsilon} \epsilon_a f(x) dx = \epsilon_a \cdot \Pr(X \leq \epsilon_a) + \epsilon_a \cdot \Pr(X > \epsilon_a)$$

• $\Pr(X > \epsilon_a) \leq \epsilon$

$\in \mathbb{N}$

$$\text{Ea} P(X \leq \epsilon_0) = P(E \leq X) \leq \frac{\epsilon}{2} P(X \leq \frac{\epsilon}{2}) = P(X \geq \epsilon_0) \leq$$

$$\leq \frac{\epsilon}{2} \cdot P[L_n(A(\epsilon))] \leq \frac{\epsilon}{2} J = P[L_n(A(\epsilon)) > \frac{\epsilon}{2}] < \frac{\epsilon \cdot 1}{2} + \delta_a =$$

$$\frac{\epsilon}{2} - \frac{\epsilon}{2} = \frac{\epsilon}{2}$$

$$P[X > \frac{\epsilon}{2}] \leq \delta_a \quad \forall n \in \mathbb{N} \quad \text{and} \quad P[X > \epsilon] \leq \delta_a$$

$$\Rightarrow P(X > \epsilon) \leq \delta_a \quad \text{and} \quad P(X > \frac{\epsilon}{2}) \leq \delta_a$$

[2] Let $X = \mathbb{R}^2$, $\gamma \in \{0, 1\}$, $\partial I = \{h_i | i \in \mathbb{Z}\}$ where $h_i(x) = 1_{\{x \in V_i\}} \leq r$.
 (where V_i is a small ball around i)

Define $E = \cup_i \partial I$ is PAC learnable and its sample complexity bound by

$$m_{\alpha}(E) \leq \frac{\log(f)}{\epsilon}$$

: for all $w \in A$ consider x

$E(w) \cap B \neq \emptyset$ $\exists s \in A : s \rightarrow h_s$, $A \cap \partial I \neq \emptyset$ $\forall i \in \mathbb{Z}$

$\{h_i | i \in \mathbb{Z}\} = \{h_i | i \in \{0, 1\}\}$, $h_i := \{h_i | i \in \{0, 1\}\}$

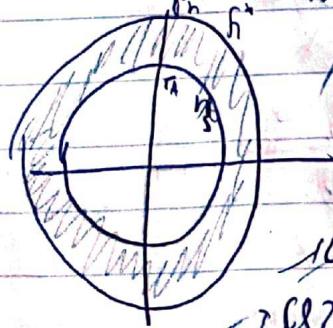
$\{h_i | i \in \{0, 1\}\}$, PAC $\Rightarrow m_{\alpha}(E) \leq \frac{\log(2)}{\epsilon}$

$\cap \{h_i | i \in \{0, 1\}\} = \emptyset$ $\Rightarrow h_i = 0$ for all $i \in \{0, 1\}$

$\Rightarrow h_0 = h_1 = 0$ $\Rightarrow h \in \{0, 1\}$ $\Rightarrow h \in E(w)$

$\therefore m_{\alpha}(E) \leq \min\{m_{\alpha}(B), m_{\alpha}(A)\}$ $\leq \min\{m_{\alpha}(V_0), m_{\alpha}(V_1)\}$ $\leq \min\{m_{\alpha}(V_0), m_{\alpha}(V_1)\}$

$m_{\alpha}(V_0) \leq m_{\alpha}(V_1) \leq m_{\alpha}(V_0) \leq m_{\alpha}(V_1) \leq m_{\alpha}(V_0) \leq m_{\alpha}(V_1)$



$\leftarrow (w)$

$L_{D,f}(A(S)) \geq \epsilon$ \Leftrightarrow $\Pr_{\sigma \sim D}[\sigma \in A(S)] \geq \frac{\epsilon}{1-\epsilon}$

$$L_{D,f}(A(S)) \geq \epsilon \quad \text{if and only if} \quad \Pr_{\sigma \sim D}[\sigma \in A(S)] \geq \frac{\epsilon}{1-\epsilon}$$

$\Pr_{\sigma \sim D}[\sigma \in A(S)] \geq \frac{\epsilon}{1-\epsilon} \Leftrightarrow \Pr_{\sigma \sim D}[\sigma \in A(S)] \geq \frac{\epsilon}{1-\epsilon}$

$D(\sigma) \leq \delta \Leftrightarrow \Pr_{\sigma' \sim D}[\sigma' \in A(S)] \geq \Pr_{\sigma \sim D}[\sigma \in A(S)] - \delta$

$\Pr_{\sigma \sim D}[\sigma \in A(S)] \geq \Pr_{\sigma' \sim D}[\sigma' \in A(S)] - \delta$

$\ln(1/(1-\epsilon)) \geq \ln((1-\delta)/(1-\epsilon))$

$$\ln(1/(1-\epsilon)) = \ln((1-\delta)/(1-\epsilon)) \Leftrightarrow \frac{\ln(1/(1-\epsilon))}{\ln((1-\delta)/(1-\epsilon))} = \frac{1}{1-\delta}$$

$$\ln(1/(1-\epsilon)) / \frac{1}{1-\delta} \Leftrightarrow \frac{\ln(1/(1-\epsilon))}{\ln((1-\delta)/(1-\epsilon))} = \frac{1}{1-\delta}$$

$$\Pr_{\sigma \sim D}[\sigma \in A(S)] \leq \epsilon \Leftrightarrow \Pr_{\sigma \sim D}[\sigma \in A(S)] \leq \epsilon \Leftrightarrow \Pr_{\sigma \sim D}[\sigma \in A(S)] \leq \epsilon$$

③ VC dimension

③

$$VCdim(H) \leq \lceil \log_2 |H| \rceil$$

הה H הוא קבוצה של היפרפליינר $\{h_i\}_{i=1}^n$

כל $x \in \mathbb{R}^d$ נקבע $c_i(x)$ על ידי $c_i(x) = \sum_{j=1}^n h_j(x) \cdot w_j$

כל $x \in \mathbb{R}^d$ נקבע $c_i(x) = \sum_{j=1}^n h_j(x) \cdot w_j$

אנו

$|H| = 2$ \Rightarrow $e \in H$, use $b, c \in G$ a coprime to n
 $b, c \in H$ $\Rightarrow b, c \in \langle e \rangle$ $\Rightarrow b, c \in \langle e^2 \rangle$
 $(c = ex, \dots, y_n)$ $\Rightarrow |H| = 2^2$ \Rightarrow $\text{Vcdm}(H) = 2$
 $|H| = 2^{2^n} \Leftrightarrow \log_2 |H| = 2^n \Leftrightarrow \text{Vcdm}(H) = \log_2 |H|$
 $\text{Vcdm}(H) \leq \log_2 |H|$

$$(4) \quad \exists J \subset [n] \text{ s.t. } Y = \{y_i\}_{i \in J} \quad \text{and} \quad X = \{x_i\}_{i \in J}$$

$$h_J(r) = (\sum_{i \in J} x_i) \bmod 2$$

$$H_{\text{range}} = \{ h_J \mid J \subset [n] \} \quad \text{such that} \quad h_J \in \text{VCDM}(H)$$

$H = \{h_{\text{range}}(x_1, \dots, x_n) \mid x_i \in \{0, 1\}\}$
 $x_i \in \{0, 1\} \Rightarrow h_{\text{range}}(x_1, \dots, x_n) \in \{0, 1\}$
 $\Rightarrow H \subseteq \{0, 1\}$
 $\Rightarrow H \subseteq \{0, 1\} \times \{0, 1\} \times \dots \times \{0, 1\} = \{0, 1\}^n$
 $\Rightarrow H \subseteq \{0, 1\}^n$
 $\Rightarrow H \subseteq \{0, 1\}^n \Rightarrow |H| = 2^n$
 $\Rightarrow |H| = 2^n \Rightarrow \log_2 |H| = n$

(w)

$\text{deg}(\text{lcm}(H)) = \lfloor \log_2 |H| \rfloor + O(\log n) \quad (\text{BPP}) \quad n \geq 1 \quad (3)$

$$M = \sum_{x \in m/2} h(x) = \left(\sum_{i \in I} x_i \right) m/2$$

ISCHIOPRIMUS (SIC)

$$h_n(x) = \begin{cases} 1 & \text{if } 2k+1 \leq n < 2k+e \\ 0 & \text{otherwise} \end{cases}$$

For $\lambda \in N$, $\lambda \neq 0$ \Rightarrow λ is a root of f_{λ} .
 λ is a root of f_{λ} if and only if λ is an eigenvalue of A .

$h_2(x) = \frac{1}{2}x^2 + 1$ is a convex function.

8581 200 XN-2) FDRZ-5) = P(?) (6) 200/R 6 x 61 +)

• $(V \otimes W)^{\otimes n} \cong V^{\otimes n} \otimes W^{\otimes n}$. \otimes is \otimes -distributive.

جـ ٢٠١٣/١٢/٢٧

$$\sum_{k=1}^n \binom{n}{k} = 2^n$$

2ⁿ ରେଗ୍ ମୋ ଡିପ୍ଲା ହା ରଜ୍ଯ ରଖି ଏକାନ୍ତରେ ଉପରେ

所以 $\lim_{n \rightarrow \infty} f_n(x) = f(x)$ 在 $[0, 1]$ 上几乎处处成立。

12) $\text{Vcdm}(H_1) \leq \text{Vcdm}(H_2)$ (P1), $\text{Vcdm}(H_2) \leq \text{Vcdm}(H_1)$ (P2), $\text{Vcdm}(H_1) = \text{Vcdm}(H_2)$

$$\text{or } \nabla_{\text{cdm}}(M_p) = [\log(M)]^{-1} \text{ per jor } (X \rightarrow t \text{ en})$$

(5) \rightarrow סט $\{[a_i, b_i]\}_{i=1}^k$ או קבוצה H כזו.
 $A = \bigcup_{i=1}^k [a_i, b_i]$ נס H פיזי \Rightarrow H קבוצה מותאמת H (אנו מודים $a_i < b_i$)
 $n_A(N) = \begin{cases} 0 & x \notin A \\ 2 & x \in A \end{cases}$

פ. סט H פיזי
 \rightarrow קבוצה H מותאמת H \Leftrightarrow $\exists N$
 $\forall n \in N \exists k \in H$ כך $n \in [a_k, b_k]$
 $\therefore \exists N \in \mathbb{N}$ $\forall n \in N$, $\exists k \in H$

4. $\{a_i, b_i\}_{i=1}^n$ מושתת \Rightarrow קבוצה H מותאמת
 $\therefore n_A(N) = 1 \text{ או } 2$

5. ענוקות \Rightarrow $A = \bigcup_{i=1}^k [a_i, b_i]$ מותאמת
 \rightarrow $\forall n \in N \exists k \in H$ $n \in [a_k, b_k]$
 $\therefore vcdm(H) = 2^n / 2^k = 2^{n-k}$

\rightarrow $\forall n \in N \exists k \in H$ $n \in [a_k, b_k]$
 $\therefore vcdm(H) = 2^{n-k}$
 \rightarrow $\forall n \in N \exists k \in H$ $n \in [a_k, b_k]$
 $\therefore vcdm(H) = 2^{n-k}$

ר' 3. מ' 2. \Rightarrow $vcdm(H) = 2^{n-k}$ \Rightarrow $vcdm(H) = 2^{n-k}$

ר' 3. מ' 2. \Rightarrow $vcdm(H) = 2^{n-k}$

\rightarrow $vcdm(H) = 2^{n-k}$ \Rightarrow $vcdm(H) = 2^{n-k}$

$vcdm(H) = 2^k = 2^{n-k}$ \Rightarrow $n = k + k$

\rightarrow $vcdm(H) = 2^k$ \Rightarrow $vcdm(H) = 2^k$

$\therefore vcdm(H) = 2^k$ \Rightarrow $vcdm(H) = 2^k$

$vcdm(H) = 2^k \Rightarrow \lim_{k \rightarrow \infty} vcdm(H) = \infty$

ולפ' 3)

(6) Boolean Conjunctions

Boolean Conjunctions

Octadyn R(1,1) variable cylind C'02 d N 0/1 N

$\bar{Y}_0 = 1 - Y_{1L} + Y_{1R}$ (10) is 2 CP X_{1L}, X_{1R} are 0 as
 23127 6 πN baryon conjugation is 0. It $\pi^+ N$ 2L (m) $\pi^- N$ 2R
 $\pi^+ N$ value is zero. 100% (2.5%) 2d N

Ex. 0. If x is even $\rightarrow L(x)$ is prime (P)
 $x_1 \bar{x}_1$ is odd.

$\text{dom}(f) \subseteq \mathbb{R}^n$ \Leftrightarrow f ist n -stellig

$\lim_{n \rightarrow \infty} \cos \pi n$

(2) $\forall (l) \exists p \forall s \in \omega \phi_n$

(B) 108 100 110 100 100 100 100 100 100 100

11. \bar{X} , Zeros , X , Zeros \rightarrow 11 kHz \rightarrow (1)

11. $\text{C}_2\text{H}_5\text{OH} + \text{K}_2\text{Cr}_2\text{O}_7 \rightarrow \text{CH}_3\text{COOC}_2\text{H}_5, \text{Cr}_2\text{O}_7^{2-}$

2. מילוי תבניות נורמלית ותבניות לא-נורמלית

$\text{rcdm}(U_0) \leq \partial_{\beta_2}(3^4) = 0$ (n2) \Rightarrow RON 70, $\Theta = |\Theta| = 3^{d+2}$ when
 $\log_2(3^n) \leq \log_2(3)$ 20%

$\alpha < \log(3)/2$, i.e. α is near zero but $\alpha \neq 0$.

Ex. 16. $\nabla \text{cdm}(H) = d$ (Ex) $\rightarrow P_1$

One possible form of ψ_0 is $\psi_0(x) = \text{rect}(x/a)$, where a is a positive constant.

$x_i \in \{1, \dots, n\}$ for all $i \in [d]$

λ (XN) \rightarrow λ (P3/C) (lambda) \rightarrow P1/P2/C | λ (XN) \rightarrow λ (XN) \rightarrow X (M, N)

$$C = \{e_1, e_2, \dots, e_d\} \quad \text{dim } \rightarrow P(D)$$

O 1,2 > R nM/G dP/D E. e.g. C. 20% FS, 10% UK

exel), 2) \Rightarrow $\forall n \in \mathbb{N} \exists x \in A, h(x) = y, \forall i \in \mathbb{N}, h(x_i) = y$

רשות מקרקעין מינהל ציבורי 2002

$$\text{vdm}(u_C) \in |u_C| = 2$$

111. $\text{M}_{\text{eff}} = 2^{\text{do}}$ ep 1c' 1rd op 1d C-200

$\Rightarrow x_{jN} \in \text{ker } f$ (because $x_{jN} = c'$)

$$h_i(x_j) = \begin{cases} 1 & i=j \\ 0 & i \neq j \end{cases} = x_{i,j}$$

Definition of linear function: $h_i(x) = \sum_{j=1}^n w_{ij}x_j + b_i$

Properties of linear functions:

- $h_i(x) = h_i(x_1, x_2, \dots, x_n)$
- $h_i(x) = h_i(x_1) + h_i(x_2)$
- $h_i(x) = h_i(x_1) + h_i(x_2) + \dots + h_i(x_n)$
- $h_i(x) = h_i(x_1) + h_i(x_2) + \dots + h_i(x_n) + b_i$

Agnostic PAC

(1)

$m_h^{(0)}: (0,1)^n \rightarrow \{0,1\}$ uniform distribution: $\Pr[h_i = 1] = 1/2$

$m_h(\epsilon, \delta) \leq m(\epsilon, \delta) \cdot n \cdot \ln(2n) / \epsilon^2$ proof: $\Pr[h_i = 1] = 1/2$

$D_{\text{err}}(m_h^{(0)}) = m_h^{(0)}((x, y)) = \Pr[h_i = 1 | x_i = 1]$

$x_i = 1 \in D, y = 1 \in D, \Pr[h_i = 1 | x_i = 1] = 1/2$

(2) $D^m(\{S \subseteq (X \times Y)^m \mid S \text{ is } \epsilon\text{-representable}\}) \geq 1 - \delta$

$(YR \in \mathcal{F}, D \in \mathcal{M}) \Rightarrow \exists S \subseteq (X \times Y)^m \text{ such that } S \text{ is } \epsilon\text{-representable}$

(3) $\forall h \in H \quad |L_S(h) - L_D(h)| < \epsilon$

$h_S := \arg \min_{h \in H} L_S(h) / \Pr[h \in H]$ S contains n points $\epsilon \leq \frac{\epsilon}{2}$

$\Pr[h \in H] = \Pr[\text{at least one point in } S \text{ is correctly classified}]$

(4) $|L_S(h_S) - L_D(h_S)| < \epsilon \Rightarrow L_S(h_S) \leq L_D(h_S) + \epsilon$

$m \geq m_h^{(0)}(\epsilon, \delta) \cdot \ln(2n) / \epsilon^2$ proof: $\Pr[h \in H] \geq 1 - \delta$

$\Pr[h \in H] = \Pr[\text{at least one point in } S \text{ is correctly classified}] \geq 1 - \delta$

$\Pr[h \in H] = \Pr[\text{at least one point in } S \text{ is correctly classified}] \geq 1 - \delta$

$P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 If $L_D(h_s) > L_D(h^*) + \epsilon$, then $\hat{m}_{\pi^D}(h_s) < \hat{m}_{\pi^D}(h^*)$
 $\hat{m}_{\pi^D}(h_s) = \frac{1}{n} \sum_{i=1}^n \hat{\pi}_i^D(h_s)$
 $\hat{m}_{\pi^D}(h^*) = \frac{1}{n} \sum_{i=1}^n \hat{\pi}_i^D(h^*)$

$\hat{\pi}_i^D(h_s) = \frac{1}{n} \sum_{j=1}^n \hat{\pi}_{ij}^D(h_s)$
 If $\hat{\pi}_{ij}^D(h_s) > \hat{\pi}_{ij}^D(h^*)$, then $\hat{\pi}_{ij}^D(h_s) = 1$
 If $\hat{\pi}_{ij}^D(h_s) \leq \hat{\pi}_{ij}^D(h^*)$, then $\hat{\pi}_{ij}^D(h_s) = 0$

$P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 If $\hat{\pi}_{ij}^D(h_s) = 1$, then $\hat{\pi}_{ij}^D(h^*) = 0$
 If $\hat{\pi}_{ij}^D(h_s) = 0$, then $\hat{\pi}_{ij}^D(h^*) = 1$
 $\hat{\pi}_{ij}^D(h_s) = \frac{1}{n} \sum_{k=1}^n \hat{\pi}_{ijk}^D(h_s)$
 $\hat{\pi}_{ijk}^D(h_s) = \begin{cases} 1 & \text{if } h_s(k) = h^*(k) \\ 0 & \text{otherwise} \end{cases}$
 $\hat{\pi}_{ijk}^D(h^*) = \begin{cases} 1 & \text{if } h^*(k) = h^*(k) \\ 0 & \text{otherwise} \end{cases}$

$P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 $P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 $P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 $P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$

$P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 $P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 $P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$
 $P_{S \rightarrow D} [L_D(h_s) \leq L_D(h^*) + \epsilon] \geq 1 - \delta$

Monotonicity

(9) PACAns (ϵ) $\in \mathcal{C}(\Omega)$ \Rightarrow $f(\omega) > f(\omega')$ $\forall \omega, \omega' \in \Omega$
 If ω, ω' are near ϵ then $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$
 $\Rightarrow f(\omega) \approx f(\omega')$ $\Rightarrow f(\omega) > f(\omega')$
 $\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$

Since $f(\omega) > f(\omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

Since $f(\omega, 1) \in \mathcal{C}(\Omega)$, $(\forall \omega, m \approx m_\mu(\epsilon, \omega)) \Rightarrow A(\omega) \rightarrow h_f(\omega) \in \mathcal{C}(\Omega)$
 $0 < \epsilon, \leq \epsilon \ll \sigma(\omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$\forall \omega, \omega' \in \Omega$ $m_\mu(\epsilon, \omega) \approx m_\mu(\epsilon, \omega')$ $\Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

$m \approx m_\mu(\epsilon, \omega) \Rightarrow f(\omega) \in \mathcal{C}(\Omega)$

QED

(b) $H_1 \subset H_2$ of $\text{VC}(H_1) < \text{VC}(H_2)$
 $\text{VC-dm}(H_1) \leq \text{VC-dm}(H_2) + 1$

Proof: $\text{VC}(H_1) = n$ given $H_1 \subseteq H_2$ $\text{VC}(H_2) \geq n$

Construction: $\exists N$ $\forall h \in H_2$ $\exists l \in H_1$ s.t. $h \in \text{supp}(l)$

H_1 is closed under $\text{VC}(H_1) = n$ $\text{VC}(H_2) \geq n$

Goal: $\text{VC}(H_2) \geq n+1$
 $\exists N$ $\forall h \in H_2$ $\exists l \in H_1$ s.t. $h \in \text{supp}(l)$

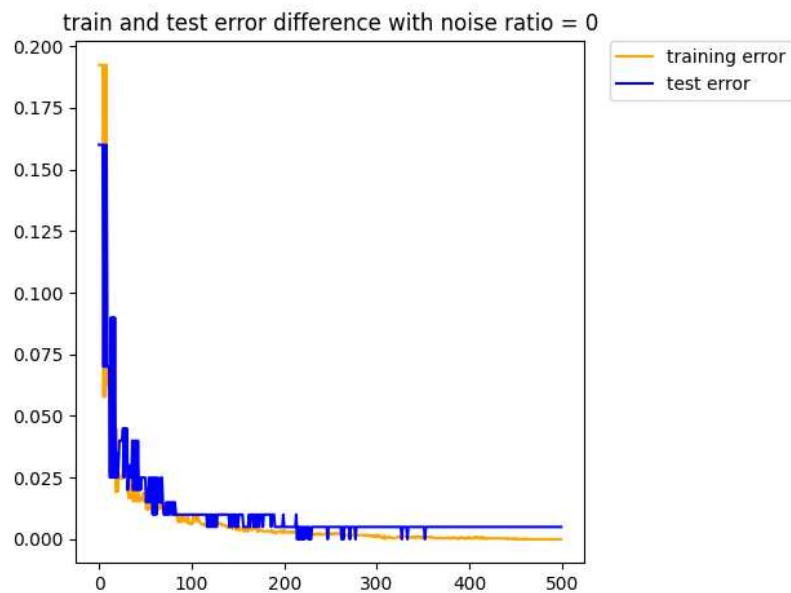
$\exists N$ $\forall h \in H_2$ $\exists l \in H_1$ s.t. $h \in \text{supp}(l)$

$H_1 \subseteq H_2 \Rightarrow \text{VC-dm}(H_1) \leq \text{VC-dm}(H_2)$

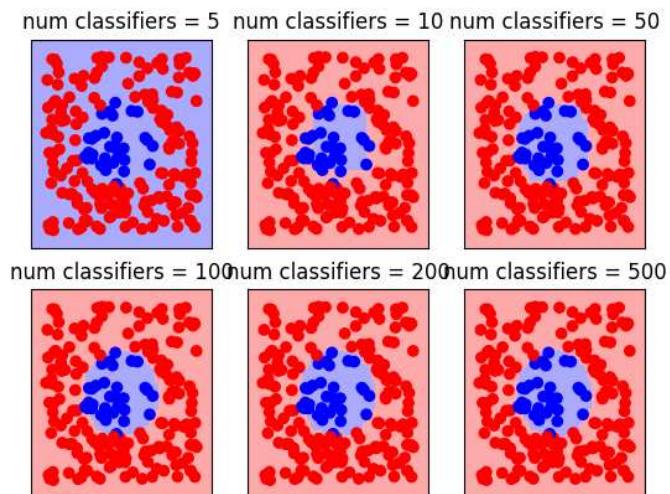
IML ex4 part2

אביב אוחיון 313410458

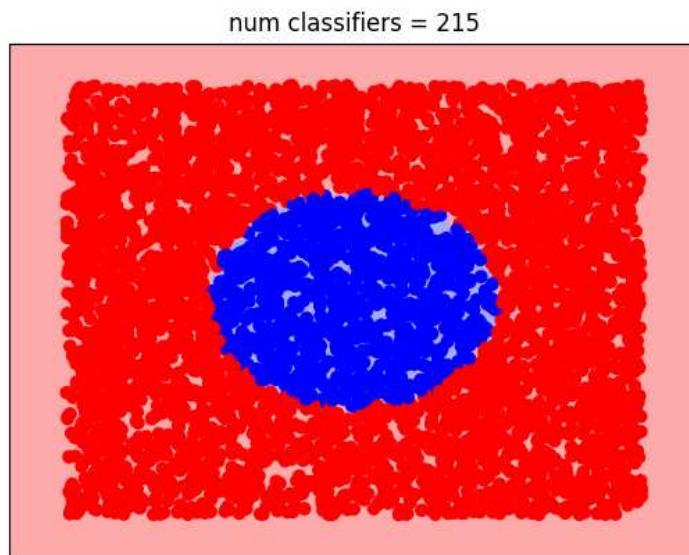
שאלה 13



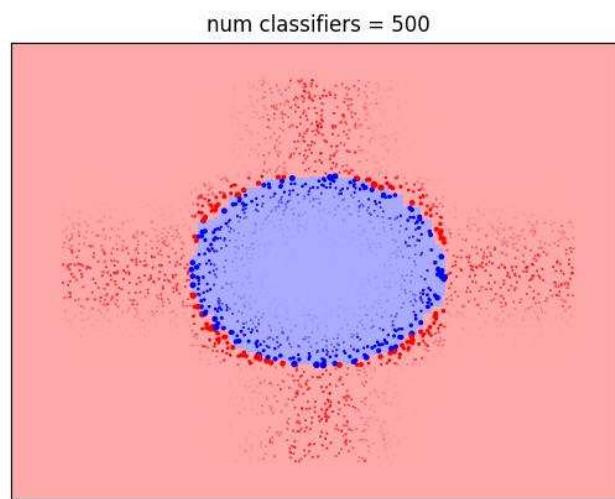
שאלה 14



שאלה 15



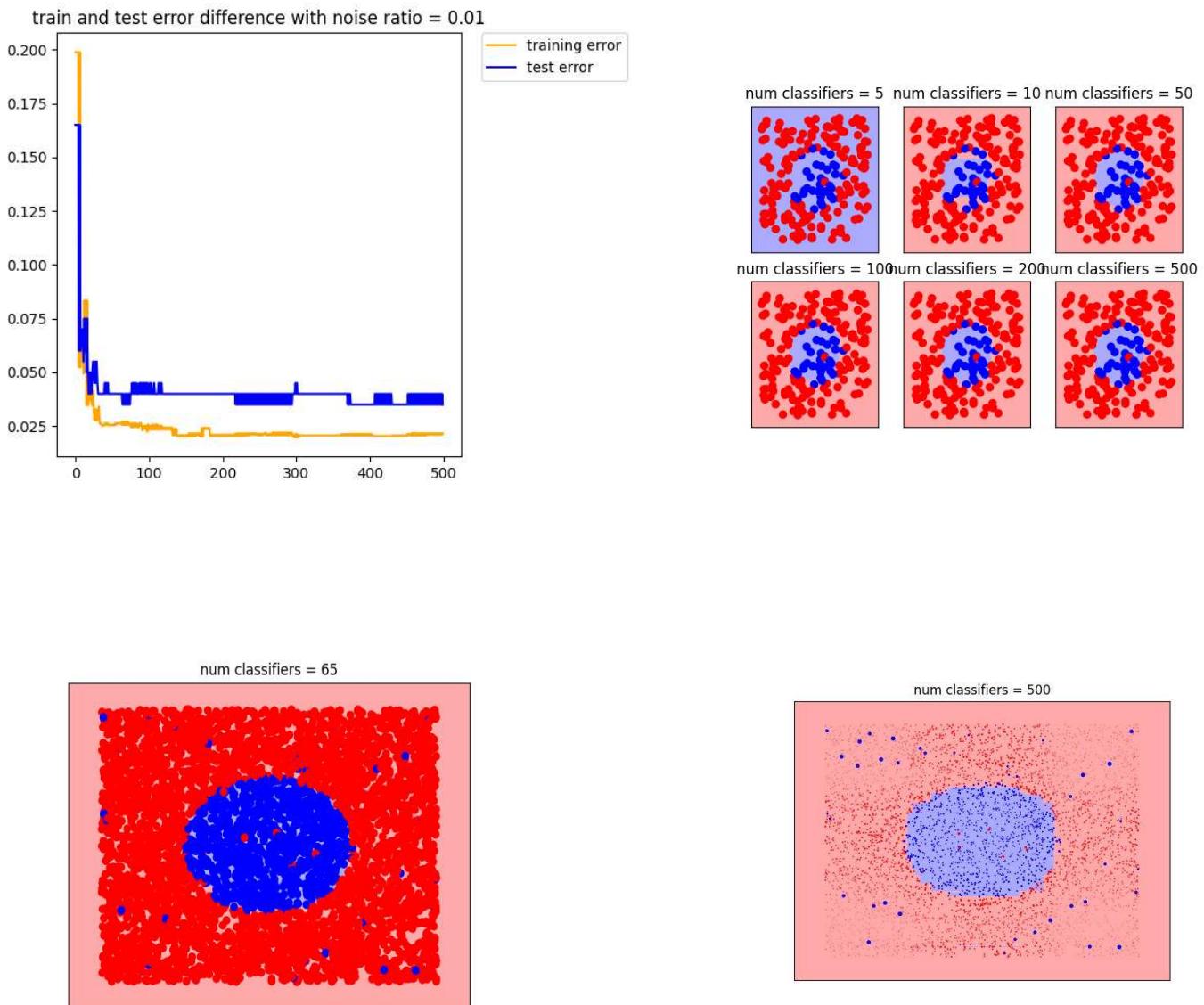
שאלה 16

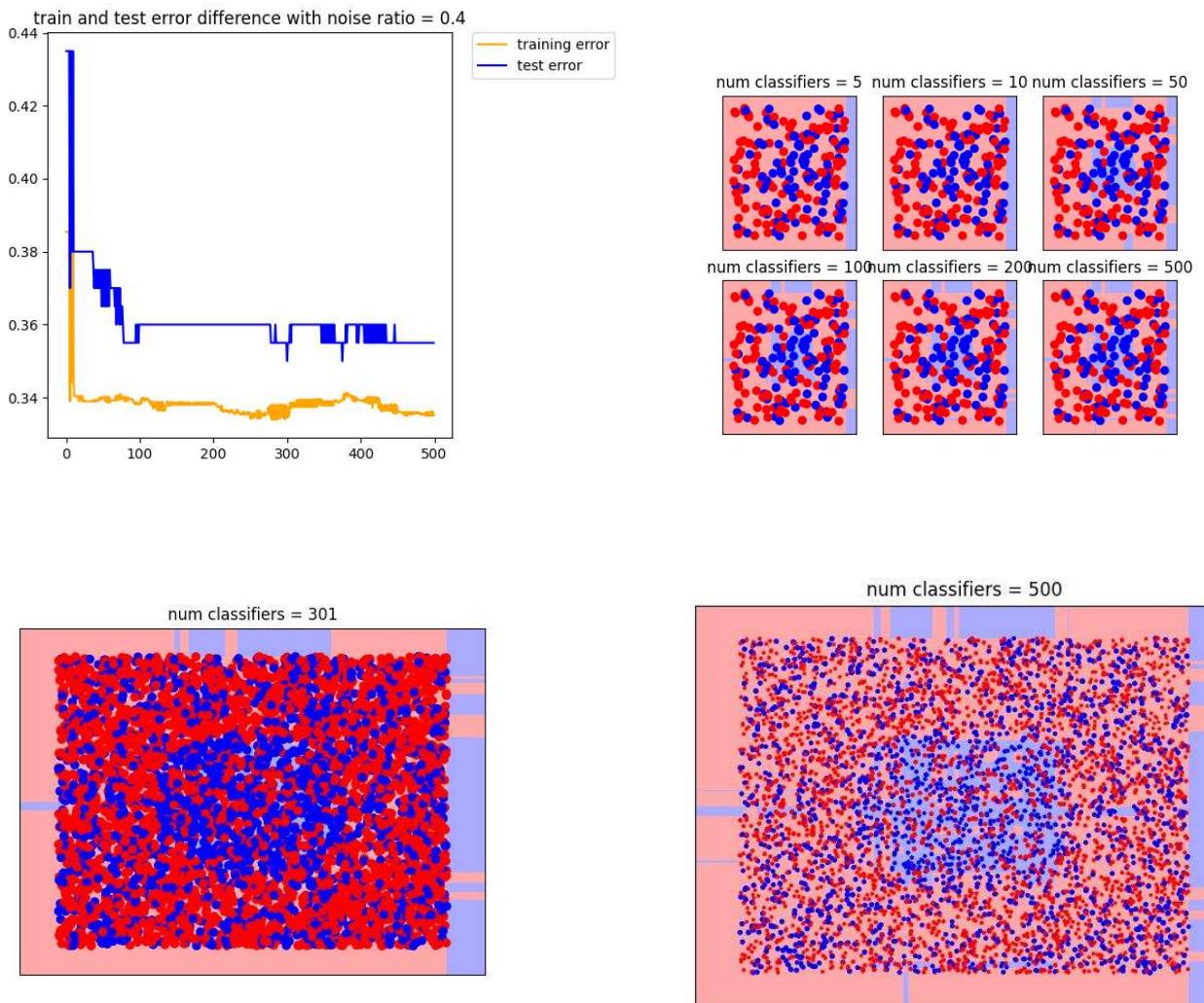


הסביר לגרף: כאשר עשינו adaboost לדאטא שלנו השתמשו בלומד חלש מסוג stump (decision stump).
שזהו עץ בעל דרגת פיצול 2, כלומר, בכל איטרציה הוא חילק את המרחב שלנו ל 2 חלקיים בלבד.
הדאטה אשר קיבלנו הוא דאטא ללא רוש התפלג נורמלי כפי שניתן לראות בגרף הקודם ולכן
המסה המרכזית של הנקודות הכהולות היא במרכז והאידומות מסביב. בכל איטרציה של
adaboost ייצורנו מרכיב מדגם אשר נותן משקלות גדולות יותר על תיוגים שהקלספירים
הקודמים טעו בו, את נקודות אשר הלומד החלש טעה בהן ניתן לראות כנקודות בגדים שונים
בגרף. מהගраф ניתן לראות כי מרבית החלוקות בהם הקלספירים טועו יהיו באיזור מרכז הגראף.

מה שיצור מעין "פלוס" אשר יציג איך כל מסוג סיווג את האיזור הזה בדאטא (כל חלוקה תהיה קו ישר במקביל לצירים המתאימים בעלי מסה גדולה יותר של נקודות כחולות במרכז ואדומות בצדדים). ניתן לראות מהגרף הנ"ל כי האיזורים הורודים הם איזורים שביהם רוב הקלאסיפירים הסכימו ולכן הנקודות נראהות "מרוחות" בצורה אחידה, עם זאת באיזור של "הפלוס" כאמור ישן כמות גבוהה יותר של נקודות בהם הקלאסיפירים לא "הסכימו" באופן גורף, הדבר ההגוני כי כפי שניתן לראות, סיבוב העיגול יש נקודות שקשה יותר לטייג אותם בצורה נכונה ולכן גם אנחנו רואים את הנקודות בגודל גדול יותר מהשאר ולכן `adaboost` נותן לאיזורים אלו משקלות גדולות יותר בכל איטרציה.

שאלה 17





הספ לתוכאות הגרפים עקב הוספה רעש בעוצמה $0.01 + 0.4$.
 ראשית נשים לב שככה שהעלינו את הרעש, ככה גם השגיאה עלתה כאשר עבר חוסר רעש קיבלנו 0, עם רעש קטן 0.01 קיבילנו שגיאה של 0.05 ועם רעש יחסית גדול קיבלנו שגיאה של 0.36.
 בהקשר של bias variance tradeoff הסיבות לכך הם שכך שביצענו יותר איטרציות אימון ככה גם קיבלנו מסווג מרכיב יותר, עם זאת המרכיבות הללו תלויות בדאטא עליו אימנו מה שמצד אחד צורך bias נמוך על הסAMPLים ומצד שני שונות גובהה, מה שיוביל שאנו אכן בוחנים את המשוגע על test samples אנחנו נצליח פחת לנבא אותו בצורה הנכונה, כלומר עשינו overfitting לדאטא שלנו.

כמו כן נשים לב שכבר אחרי 100-50 איטרציות כלומר 100-50 מסווגים בוידעה עבור רעש של 0.01 וכבר על ידי 300-200 מסווגים בוידעה עבור רעש של 0.4 הגענו למצב של התוצאות כאשר התוצאות של "הכרעת הרוב" לא מועלות בחיזוי נאמן יותר של התוצאות על הדאטא.
 ניתן לראות זאת על ידי שakan החומר argmin()

בדיקות בטוחים הללו עברו כל עצמה מסוימת של רעש ($T=301$ ו- $T=65$ לרעש החלש והחזק בהתאם).

בנוסף לכך, בשונה מהניסיונות הראשונים בו לא היה לנו רעש, הפעם הפיזור של השגיאות של הקלטפסיירים הוא רחב יותר, עברו רעש נמוך של 0.01 עדין נוצר מעין "פלוס" כמו במקרה בלי הרעש, אך כיוון שיש לנו רעש מסוים גם "קפצו" כל מיני נקודות כחולות באקריות על המרחב מה שגם להרבה קלטפסיירים לתמיג בצורה לא נכונה. עם זאת כיוון שהרעש היה יחסית קטן עדין רוב השגיאות של المسؤولים היו סביר אותו ה"פלוס" ובפרט רוב התוצאות של הנקודות הכהולות אכן היו במרכז כפי שהיינו מצפים.

במקרה של רעש גדול 0.4, הרעש הזה, אינטיאוטיבית משנה את הדאטה להיות כמעט ומרקורי (אם לדוגמא היה רעש של 0.5 אז לנסות לבבא איפה הדאטה המקורי נפל היה כמעט כמו לנחש מה ערכו בהטלת מטבח) וכך ניתן לראות שקלטפסיירם טעו בצורה גורפת כמעט על כל מרחב המדגם, ככלומר, כל מרחב המדגם מלא בנקודות כחולות ואדומות גדולות יחסית וכל נק צואן מתארת שגיאה של ועידת הקלטפסיירם הזו עברו תיוגם. על כן אף על פי שאנו יודעים שהדאטה הגיע מההתפלגות הנורמלית וכי הנקודות הכהולות במקור ללא הרוש היו במרכז עדין המונע נקודות כחולות תיוגו באיזורים אקרים שהם לא בעיגול המרכזי והפור לגבי האדומים.

(הערה, ניתן לראות שהוינידה "קבעה" שמרבית הנקודות הכהולות אכן צריכות להיות מתיוגות באופןן, אך עם לא בצורה "נחרצת" כמו במקרים הקודמים.)