Assignment 4, Machine Learning Fall 2018

Cristian Mitroi, dmn470

18 December 2018

Contents

1

We observe that we have three events:

- event **C**: out of 10100 of observed passengers, 9600 show up for their flight
- event **B**: out of the passengers sampled in **C**, that out of 100 sampled passengers for a given flight, 100 show up
- event **A**: out of the passengers sampled in **C**, that out of 10000 sampled passengers, 9500 show up

We then have:

$$P(A \cap B) = P(A \cap B \cap C) = P(A \cap C) = P(C)P(A|C)$$

Since $P(C) \leq 1$, we have to calculate $P(A|C)$.

We can compute this numerically, keeping in my mind that we are sampling *without* replacement:

$$= \frac{9600}{10100} \cdot \frac{9599}{10099} \cdot ... \cdot \frac{9500}{10000} = 0.00607$$

2 Growth function

2.1

We know that $m_h(n) \leq 2^n$.

Since this is a *finite* hypotheses space, we cannot have more dichotomies than hypotheses. Thus: $m_h(n) \leq M$

So: $m_h(n) \leq min\{M, n^n\}$.

2.2

Since this is a *finite* hypotheses space, we have $d_{VC}(H) \leq log_2|H|$.

2.3

If $d_{VC}(H) = \infty$ we have

$$m_H(2n) = 2^{2n}$$

$$m_H(n)^2 = (2^n)^2$$

which are equal.

If $d_{VC}(H) \neq \infty$ there exists a break point k, such that:

$$m_H(n) = k$$

$$m_H(n)^2 = k^2$$

Then $m_H(2n) \leq k^2 = m_H(n)^2$ *q.e.d.*

2.4

We have the VC dimension of H is d ($d_{VC}(H) = d$)

From the *Learning from Data* ((Abu-Mostafa, Magdon-Ismail, and Lin 2012)) book we have the following:

$$m_H(n) \leq B(n, d+1)$$

(where d is the VC dimension of H and $d+1$ is the breakpoint of H, and the entire right-hand side is “the maximum nr of dichotomies on N points such that no subset of size k of the N points can be shattered”)

We also have that:

$$B(n, d+1) \leq \sum_{i=0}^{d} \binom{n}{i}$$

(which is also called **Sauer's lemma**)

We have that:

$$\sum_{i=0}^{d} \binom{n}{i} \leq n^d + 1$$

Thus, we can conclude that:

$$m_H(n) \leq \sum_{i=0}^{d} \leq n^d + 1$$

2.5

From the previous exercise we got the bound $m_H(n) \leq n^d + 1$

In the VC generalization bound we want to replace $m_H(2n)$ with this bound. We get that $m_H(2n) \leq 2n^d + 1$.

Thus we get the following expression for the VC gen. bound:

$$P(\exists h \in H : L(h) \geq \hat{L}(h, S) + \sqrt{\frac{8ln(2(2n^d+1)/\delta)}{n}}) \leq \delta$$

$$\implies P(\exists h \in H : L(h) \geq \hat{L}(h, S) + \sqrt{\frac{8ln(4n^d+2)/\delta)}{n}}) \leq \delta \tag{1}$$

2.6

d should be small such that $m_H(n) << 2^n$ ("much smaller than").

In order for the bound to be meaningful we need to have a small value of the square root expression. For this we need for N to be large, so that the denominator is large.

At the same time, d needs to be small. The smaller it is, the faster the square root expression convergences to 0 as N grows.

2.7

The bound in **Theorem 3.2** is tighter than the one we have derived in eq. 1. This is because the expression in **T 3.2** is divided by larger number ($2n$).

3 VC Dimension

3.1 Positive circles

To show that the VC dimension of the circle is ≥ 3 we have to show that the "positive circle" hypothesis can shatter 3 points. This means that the hypothesis can, for a given arrangement of 3 points, correctly classify all possible dichotomies ($m_h(N) = 2^N \implies m_h(3) = 8$)

We know that any 3 non-collinear points form a triangle. We can see that all the arrangements where the points form a triangle can be shattered. We simply draw the circle to include none, one, two, or three of the points, depending on the labeling.

Below you can see all the possible dichotomies of a specific arrangement of 3 points.[1]

[1] The arrangements and labelings were produced with `matplotlib` (see `ex3.py`), while the circles were drawn in Microsoft Paint. I have ensured that they are *perfect* circles by pressing the `Shift` key while drawing them (see https://www.wikihow.com/Draw-a-Perfect-Circle-on-Microsoft-Paint)

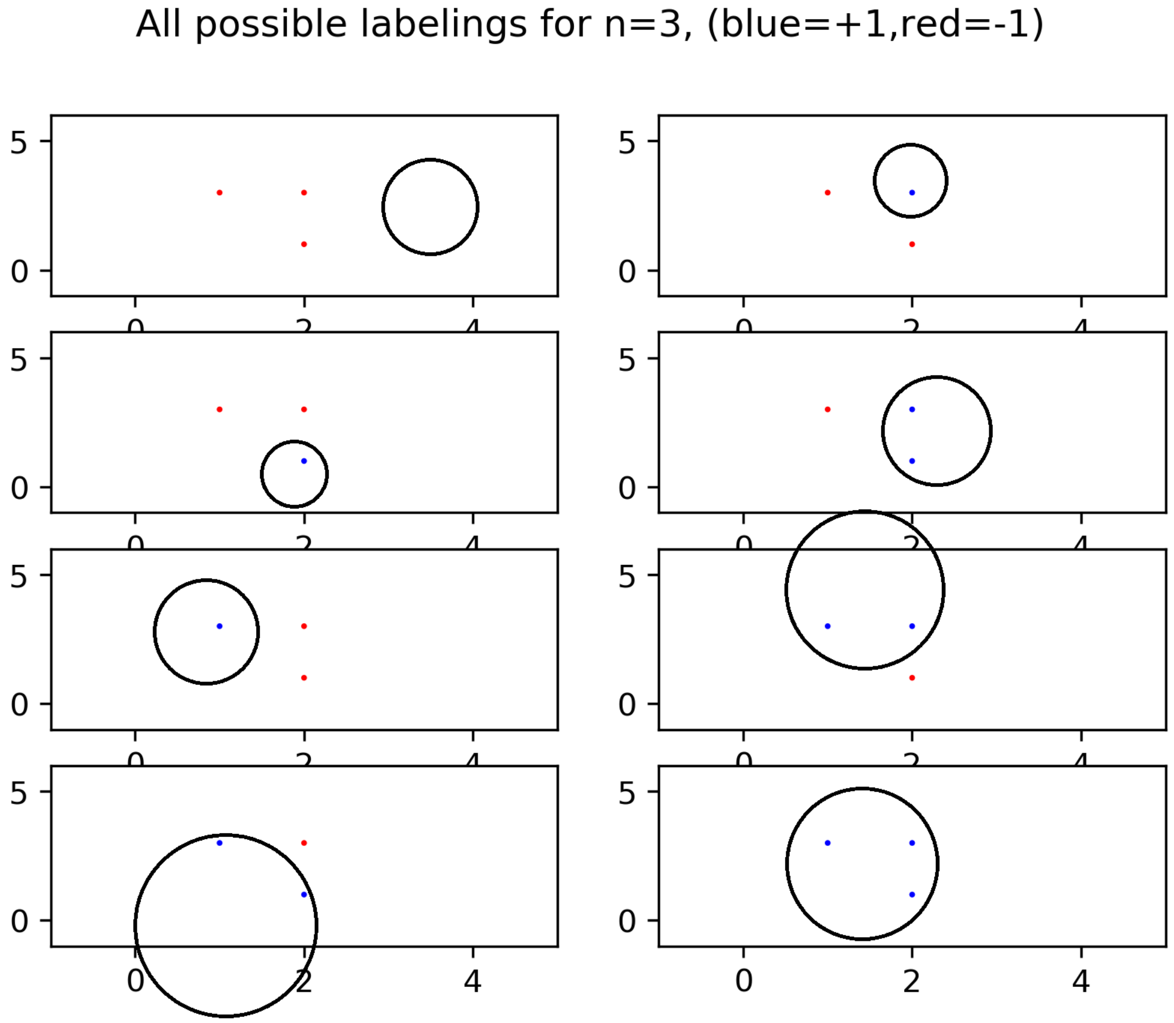

Figure 1: positive circles figures

We can thus conclude that the $d_{VC}(H_+) \geq 3$.

3.2 Positive and negative circles

We use a similar approach as in the previous exercise. We show that for a given arrangement of $n = 4$ points the hypothesis $H = H_+ \cup H_-$ can classify all possible dichotomies ($(m_h(N) = 2^N \implies m_h(4) = 16)$)

Since this is a union of two separate hypotheses spaces it is enough to prove that for a given dichotomy one of the two hypotheses (either "negative" or "positive" circles) can classify it.

Below you can see all the possible dichotomies of a specific arrangement of 4 points. The black circles are "positive", while the red ones are "negative".

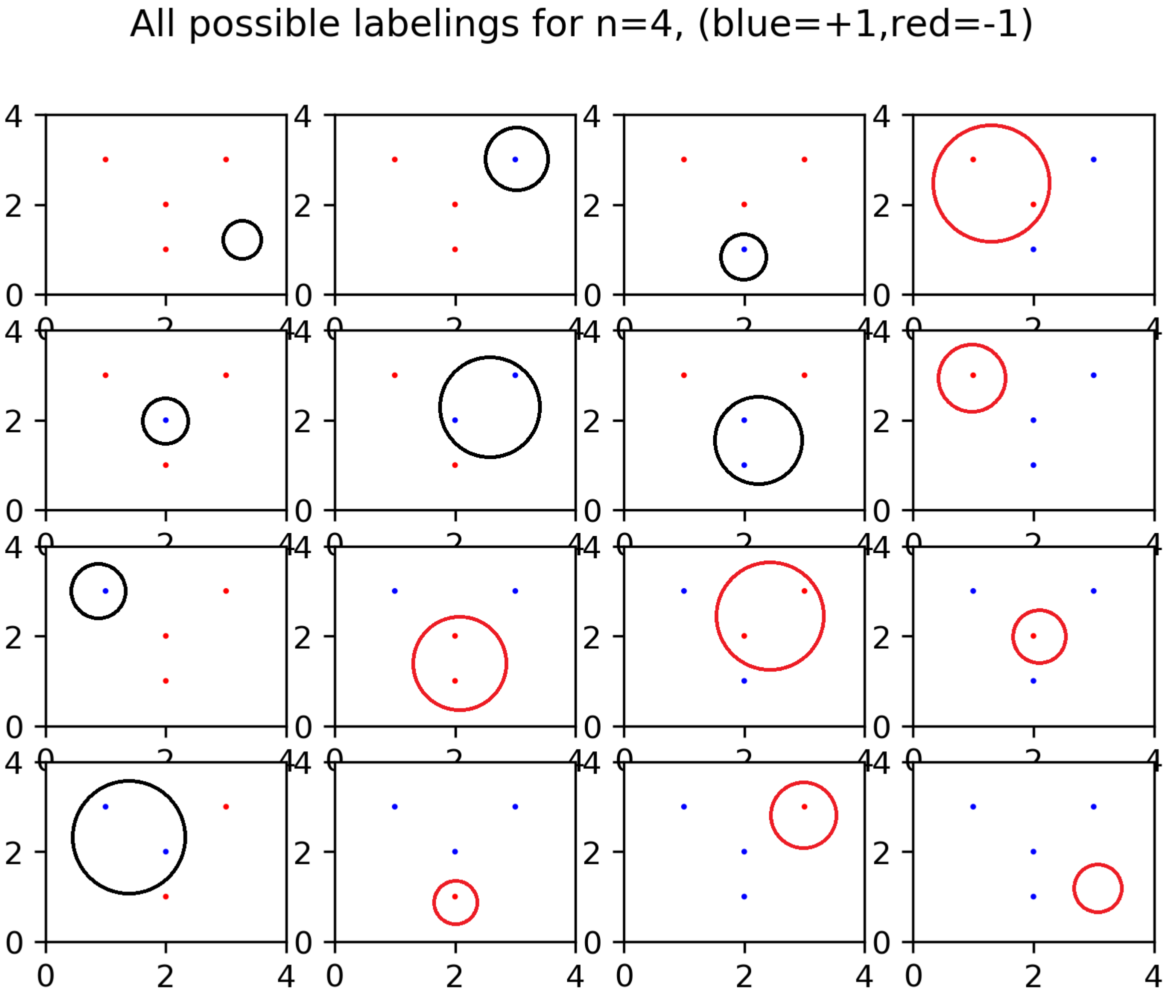

Figure 2: positive and negative circles

Thus we can conclude that $d_{VC}(H) \geq 4$.

4 SVMs

4.1 Normalization

Training data after normalization:

feature #	**mean**	**variance**
0	-0.00000	1.00000
1	0.00000	1.00000
2	-0.00000	1.00000

feature #	**mean**	**variance**
3	0.00000	1.00000
4	-0.00000	1.00000
5	0.00000	1.00000
6	0.00000	1.00000
7	-0.00000	1.00000
8	0.00000	1.00000
9	-0.00000	1.00000
10	0.00000	1.00000
11	0.00000	1.00000
12	0.00000	1.00000
13	0.00000	1.00000
14	0.00000	1.00000
15	0.00000	1.00000
16	-0.00000	1.00000
17	-0.00000	1.00000
18	-0.00000	1.00000
19	-0.00000	1.00000
20	0.00000	1.00000
21	0.00000	1.00000

Test data after normalization:

feature #	**mean**	**variance**
0	-0.07858	0.73219
1	-0.15804	0.71491
2	0.05562	0.79759
3	0.11318	1.99040
4	0.07157	1.66604
5	0.08691	2.13674
6	0.11567	1.92226
7	0.08702	2.13767
8	0.24898	1.77196
9	0.24519	1.82896
10	0.22957	1.71731
11	0.25089	1.77784
12	0.31661	2.19023
13	0.22960	1.71745
14	0.14906	2.66297
15	-0.05676	1.36090
16	0.07357	1.08263
17	0.08677	0.95131
18	0.15477	1.21651
19	0.31069	1.36280
20	0.08742	1.13352
21	0.16858	1.41470

4.2

For this part I used the `sklearn` library implementation of Support Vector Machines and Stratified K-Fold.

The SVM implementation from `sklearn` works by first taking a `C` parameter and a `gamma` (γ) parameter. It also requires a choice of kernel. In my case I opted to select the 'precomputed' option and calculate the gram matrix of the required kernel myself. In fact, the `gamma` parameter is not used by the SVM itself, but by the kernel function. So, I do not pass it to the `SVC` initializer.

For the k-fold validation process I use the `sklearn StratifiedKFold` class. It splits the data into `k` folds (in this case $k = 5$) while maintaining class distribution between them.

I then perform grid search between 7 options for the value of C and γ. I choose the values that yield the best average zero-one loss. In my case the values end up being $c = 10, \gamma = 0.1$. The lowest average zero-one loss during grid search was 0.0836.

I then train a new SVM with those parameters on the entire training set and test it on the test set. This yields a **test loss** of 0.0928. The **loss on the training set** was 0.0816.

4.3

A lower value of C will make the model more tolerant of missclassifications. On the other hand, a higher value of C will make the model less tolerant. Thus, I expect the edge cases to matter less for lower values, and the opposite for higher values.

I have taken the time to generate an artificial binary dataset and plot the SVM classification contour.

We can see the images below:

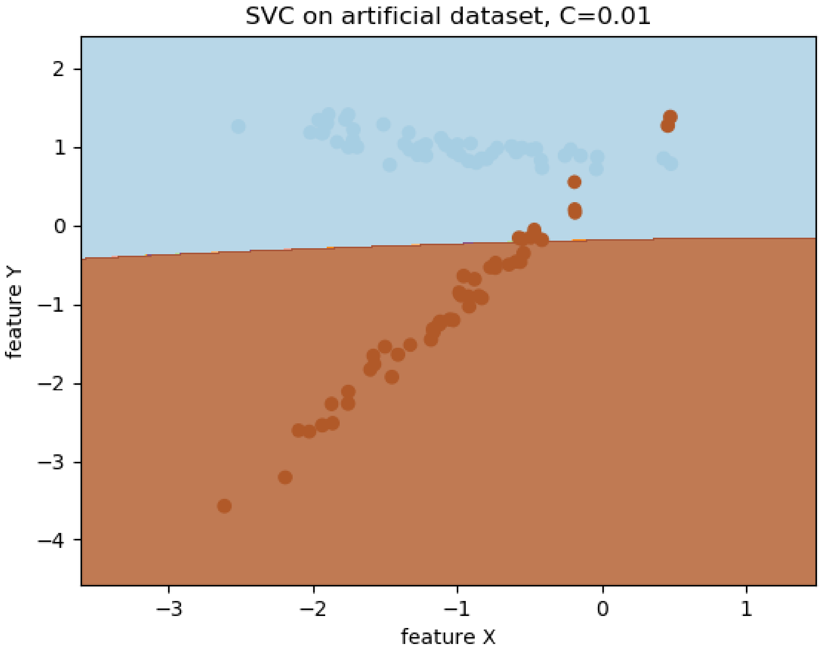

Figure 3: C = 0.01

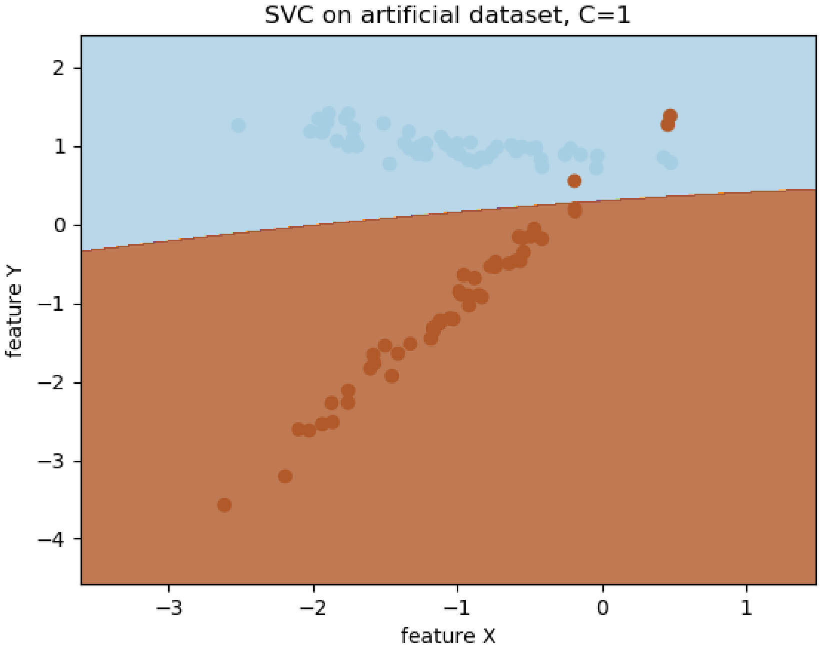

Figure 4: C = 1

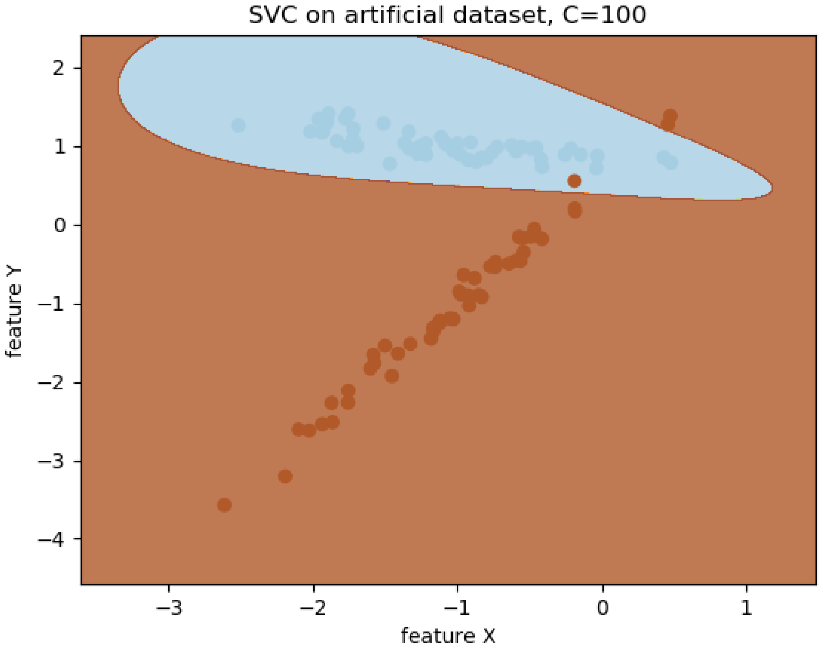

Figure 5: C = 100

My intuition is confirmed. Notice the samples that are initially misclassified for $C = 0.01$. The model adjusts itself to include these for the higher values of C. Thus, we can see that the number of bounded vectors is inversely proportional to the value of C.

References

Abu-Mostafa, Yaser S, Malik Magdon-Ismail, and Hsuan-Tien Lin. 2012. *Learning from Data.* Vol. 4. AMLBook New York, NY, USA: