

Homework #3

3.1 Growth Function and VC Dimension

- (1) List the patterns with n points, n equal 1 to 3, as Table 3.1(1)-1 :

$n = 1$	$n = 2$	$n = 3$
+	++	+++
-	+ -	++ -
	- +	+ - +
	--	- + +
		-- +
		+ --

$L(1) = 2$	$L(2) = 4$	$L(3) = 7$

Table 3.1(1)-1

We can figure out the pattern $+ - +$ cannot exist in $N = 3$, so the VC dimension $D = 2$. The general way to say is that we cannot accept the pattern with negative inside two positive signs. Therefore, all positive sign should be continuous, we consider the situation in Table 3.1(1)-2 :

$n = N$
With 0 + : 1 kind
With 1 + : N kinds
With 2 + : N-1 kinds
With 3 + : N-2 kinds
...
With N + : 1 kind
$L(N) = \text{sum up all}$

Table 3.1(1)-2

$L(N)$ for the interval learning model can represent by the formula:

$$L_{g[\alpha, \beta]}(N) = \frac{N(N+1)}{2} + 1 \quad (3.1.1)$$

Now we verify the inequality:

$$\frac{N(N+1)}{2} + 1 \leq N^2 + 1 \quad (3.1.2)$$

By transpose N^2 term and multiply 2 on both sides, we can get the new form of inequality:

$$\left(N - \frac{1}{2}\right)^2 - \frac{1}{4} \geq 0 \quad (3.1.3)$$

It is true for all positive integers N .

- (2) By put 7 points on a circle, we can reduce patterns from 128 to 18 kinds. We would illustrate some kinds of patterns (in Table 3.1(2)):

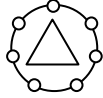
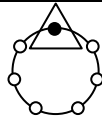
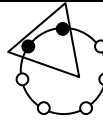
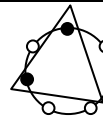

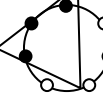

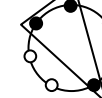
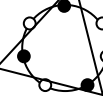
0 +	1 +	2 +			3 +			
								

Table 3.1(2)

The black point means inside triangle (+) and the white point means outside triangle. The reason we only show 0 ~ 4 positive sign is because 5 ~ 7 positive's graph is the same as 0 ~ 4 positive sign with inverse color. In fact, we don't need to illustrate all situations to proof. Consider about the most complicated patterns, (+ + - + - + -), that has six parts. We call this kind of pattern is the most complicated is because it need the biggest amount of partition. If the pattern can be separated, all of other pattern can be separated by merge the same sign to the same region.

One reason to explain is a triangle can separated a circle at most in 6 parts, 2 parts per an angle. The most complicated situation can be generated. So it is fine for our illustration. As a result, $D \geq 7$.

- (3) With the same idea, a convex polygon has n vertices can separate a circle in $2n$ parts. That's means we can generated the most complicated patterns, (+ - + - ...). For example, if we want to generate (+ - + - + - + -) pattern, we only need a quadrilateral to do this (4 vertices separated a circle to 8 parts) if we put all 8 points on a circle. Therefore, no matter the input size N , we can find a polygon with $\left\lceil \frac{N}{2} \right\rceil$ node to separate it. As a result, D is infinite.

3.2 VC Dimension of Perceptron

- (1) There are $(d+1)$ points on which all 2^{d+1} label patterns can be produced from G_p . We show the matrix from of $(d+1)$ points do inner product with the decision function and the patterns on which they are generated by equation below:

$$\begin{bmatrix} x_1 & -1 \\ \vdots & \vdots \\ x_{d+1} & -1 \end{bmatrix}_{(d+1) \times (d+1)} \begin{bmatrix} \omega_1 \\ \vdots \\ \theta \end{bmatrix}_{(d+1) \times 1} = \begin{bmatrix} s_1 \\ \vdots \\ s_{d+1} \end{bmatrix}_{(d+1) \times 1}$$

Rewrite in the simpler from:

$$P_1 W = S_1 \quad (3.3.1)$$

Where P_1 is a $(d+1)$ by $(d+1)$ matrix represent the inputs, W is decision function and S is the sign value of the inner product result of P_1 and W . We consider about $(d+1)$ dimension all are linear independent. When P over a field K , it is equivalent to an inverse matrix of P is exist such that:

$$W = P_1^{-1} S_1 \quad (3.3.2)$$

Now we can produce all of 2^{d+1} patterns and get all of their different decision function by the treat equation (3.3.2) as an inverse function of (3.3.1). Thus, $(d+1)$ points are existed on which all 2^{d+1} labels patterns can be produced from G_p .

- (2) We use the matrix to simulate the situation and the fact that at least one vector is dependent:

$$\begin{bmatrix} x_1 & -1 \\ \vdots & \vdots \\ x_{d+2} & -1 \end{bmatrix}_{(d+2) \times (d+1)} \begin{bmatrix} \omega_1 \\ \vdots \\ \theta \end{bmatrix}_{(d+1) \times 1} = \begin{bmatrix} s_1 \\ \vdots \\ s_{d+2} \end{bmatrix}_{(d+2) \times 1}$$

We use Gauss-Jordan elimination get a new matrix like this:

$$\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & 0 \end{bmatrix}_{(d+2) \times (d+1)} \begin{bmatrix} \omega_1 \\ \vdots \\ \theta \end{bmatrix}_{(d+1) \times 1} = \begin{bmatrix} s'_1 \\ \vdots \\ s'_{d+2} \end{bmatrix}_{(d+2) \times 1}$$

We have a matrix combination with an identity matrix with $(d+1)$ and a row in which all of entries are 0's, because the truth $(d+2)$ vectors of dimension $(d+1)$ have to be linear depend. It means the rank of matrix at most $(d+1)$. Suppose decision function has a $(d+1)$ rank solution, the rank of right hand side should be $(d+1)$. However, s'_{d+2} must be 0 for our assumption and It is impossible to generated 2^{d+2} patterns. Thus, $(d+2)$ points are not existed on which all 2^{d+2} labels patterns can be produced from G_p .

3.3 Experiment with 1-Nearest Neighbor

- (1) We implement 1-nearest neighbor by implement a k-nearest neighbor algorithm and set k as 1. The way we done this by using a kd-tree data structure, which provide by Andrea Tagliasacchi in MATLAB CENTRAL. We will return the nearest neighbor's information (Figure 3.3(1)).

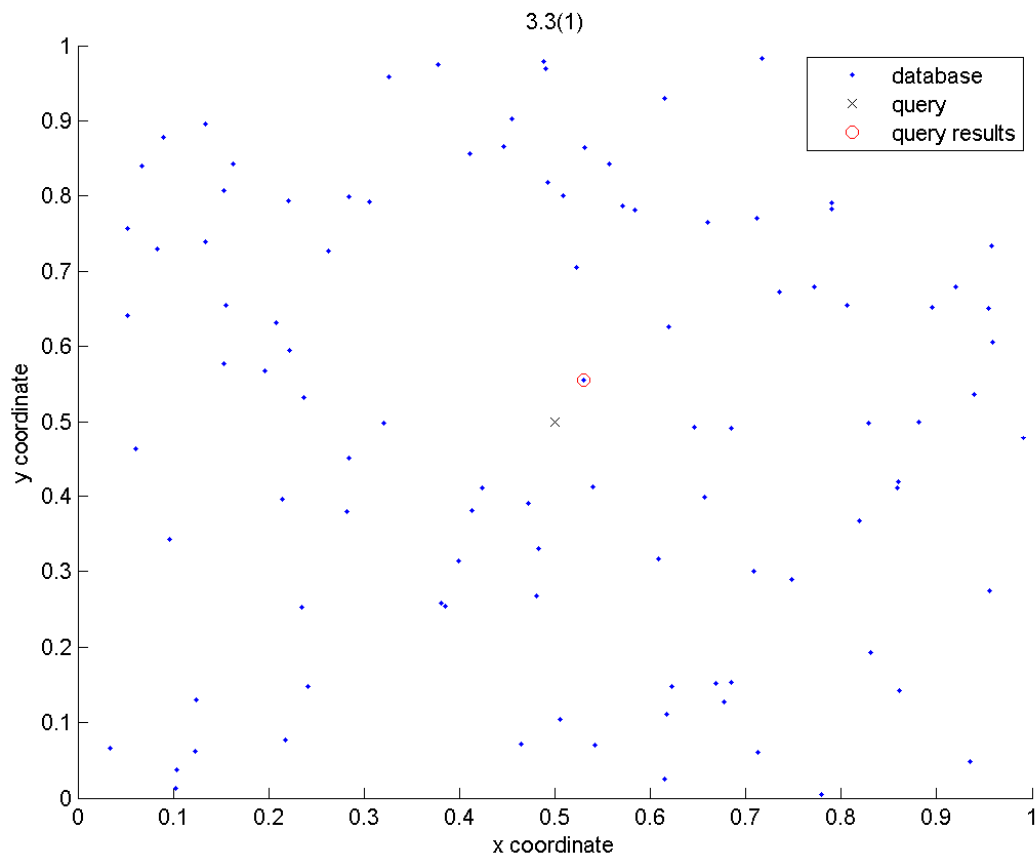


Figure 3.3(1)

We get $v(A(Z)) = 0$ and $\hat{\pi}(A(Z)) = 0.13$.

- (2) We created a random permutation from 1 to 100, and append the permutation to each data row. Sorted the whole data set with the last column, and pick the first 80 data as base and last 20 data as a validation set. Use the 1-nearest neighbor algorithm, we get $v_v(A(Z_b)) = 0.2$ and $\hat{\pi}(A(Z_b)) = 0.114$.

- (3) We modify the program to do κ -fold 1-nearest neighbor, the figure showed as figure 3.3(3)-1 :

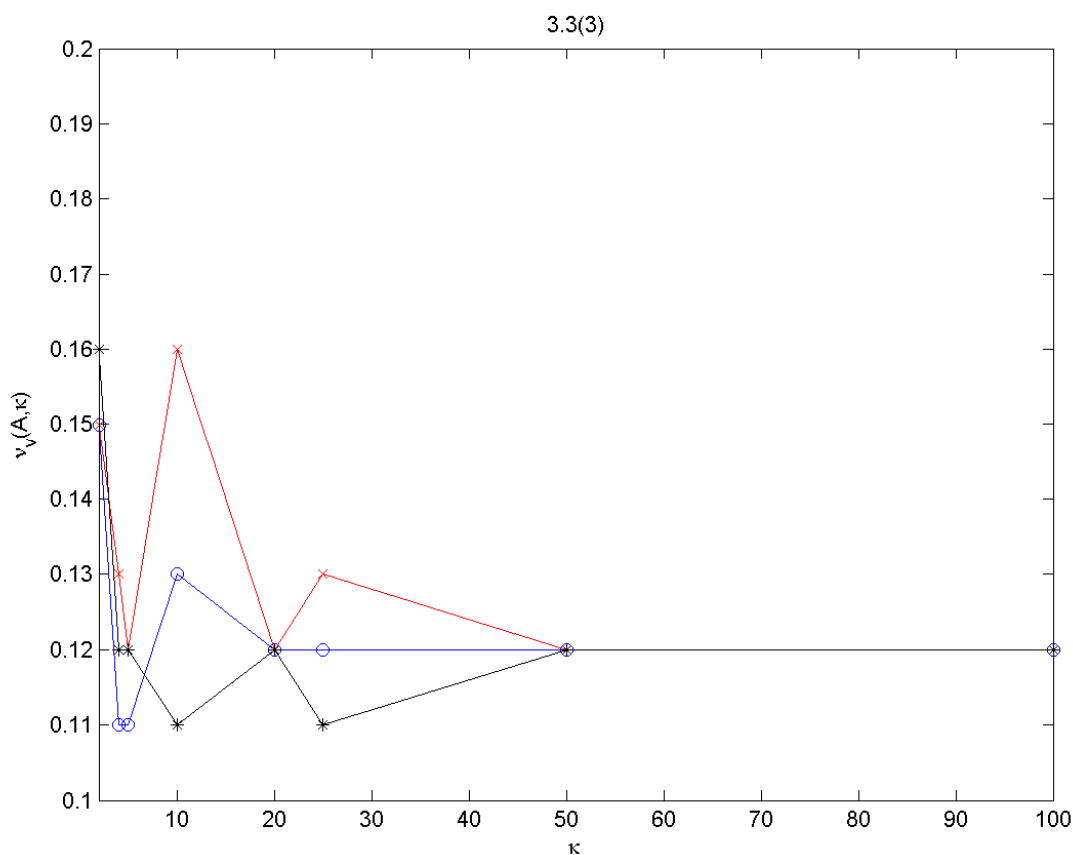


Figure 3.3(3)-1

Because we chose the validation set as random sampling, we could have lots of different situation as we showed above. However the tendency is when kappa growth up, the validation error got smaller and converged on 0.12.

Even though they have several cases, we could observe that at the end the validation error will converge to 0.12. It's because when kappa equal to 100, the process is sequential. All data will be considered as validation set once.

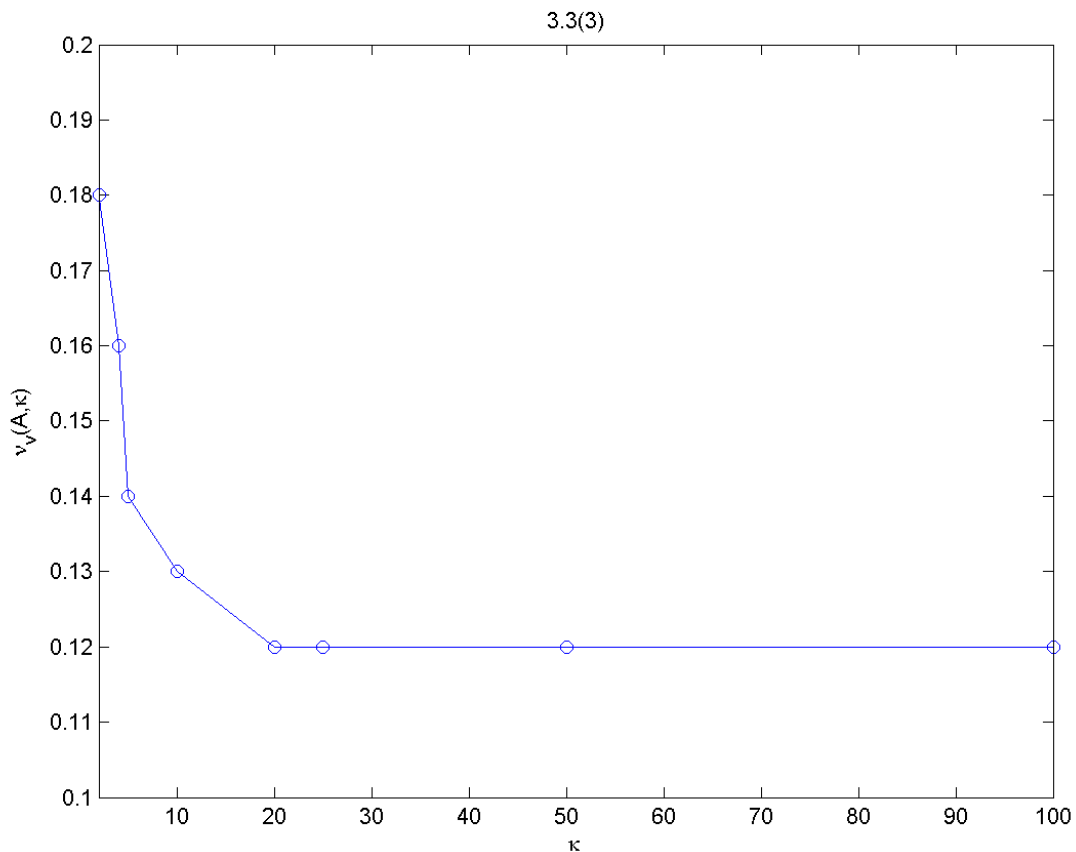


Figure 3.3(3)-2

We chose a result and use it as a contrast example, and it is showed at Figure 3.3(3)-2.

Compare the validation error $v_c(A, \kappa)$ in the third part with $\hat{\pi}(A(Z))$ in first part and $v_v(A(Z_b))$ at the second part. We can show the absolutely value of difference between each entry in Table 3.3(3). We can see the best result was in K equal to 10.

Undoubtedly, the classical validation, which use in the second part, may give us a good result. But it need you are lucky enough to pick the right validation set. Using the cross-validation reduced the risk of picking a wrong validation set, and its effect has showed.

$v_c(A, \kappa)$	$\kappa = 2$	$\kappa = 4$	$\kappa = 5$	$\kappa = 10$	$\kappa = 20$	$\kappa = 25$	$\kappa = 50$	$\kappa = 100$
	0.18	0.16	0.14	0.13	0.12	0.12	0.12	0.12
$v_v(A(Z_b)) = 0.2$	0.02	0.04	0.06	0.07	0.08	0.08	0.08	0.08
$\hat{\pi}(A(Z)) = 0.13$	0.05	0.03	0.01	0.00	0.01	0.01	0.01	0.01

Table 3.3(3)

3.4 Experiment with K-Nearest Neighbor

(1) The figure we have found as figure 3.4(1):

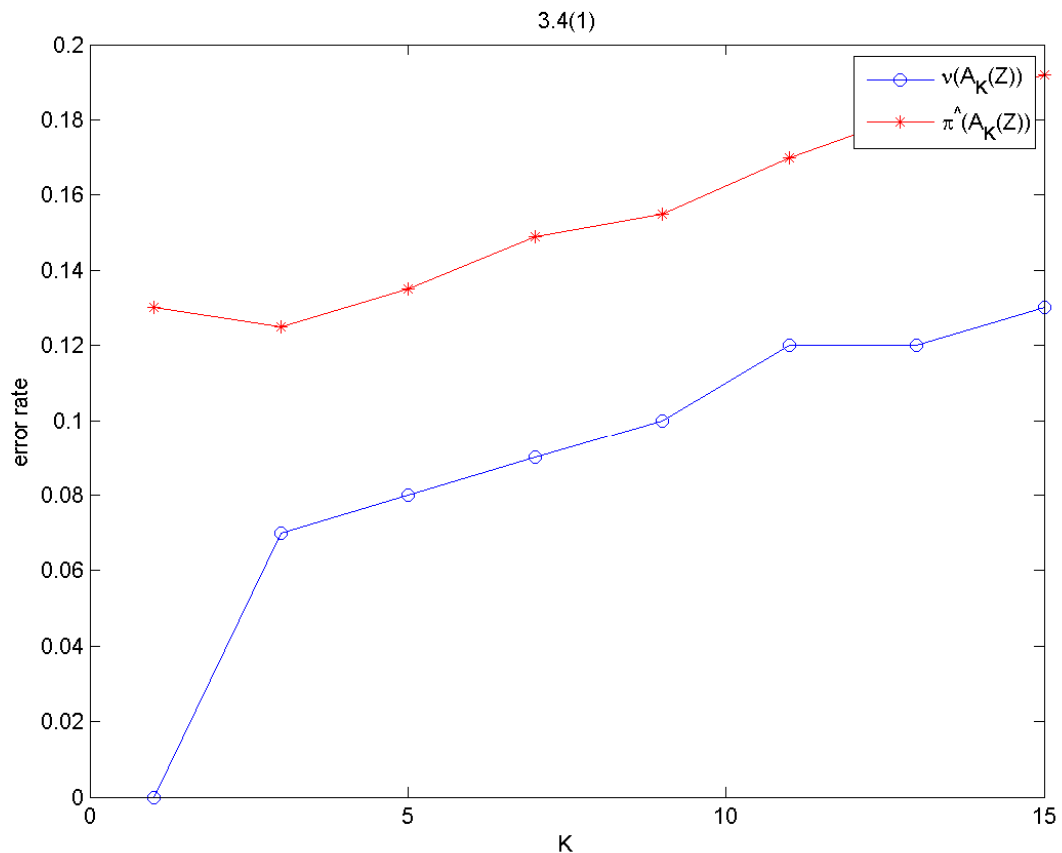


Figure 3.4(1)

The training error $v(A_K(Z)) = \{0.00, 0.07, 0.08, 0.09, 0.10, 0.12, 0.12, 0.13\}$, and testing error $\hat{\pi}(A_K(Z)) = \{0.13, 0.125, 0.135, 0.149, 0.155, 0.170, 0.182, 0.192\}$ with bins = [1, 3, 5, 7, 9, 11, 13, 15]. We can see when $K = 1$, the testing error is the same as Problem 3.3(1).

We could find out a trick things, the training error was minimum when $K = 1$. But the fact is, when $K = 3$ the training function would get the best result because it got a minimum testing error. The training error could not reflect the truth situation.

(2) We showed the result in figure 3.4(2) and table 3.4(2) :

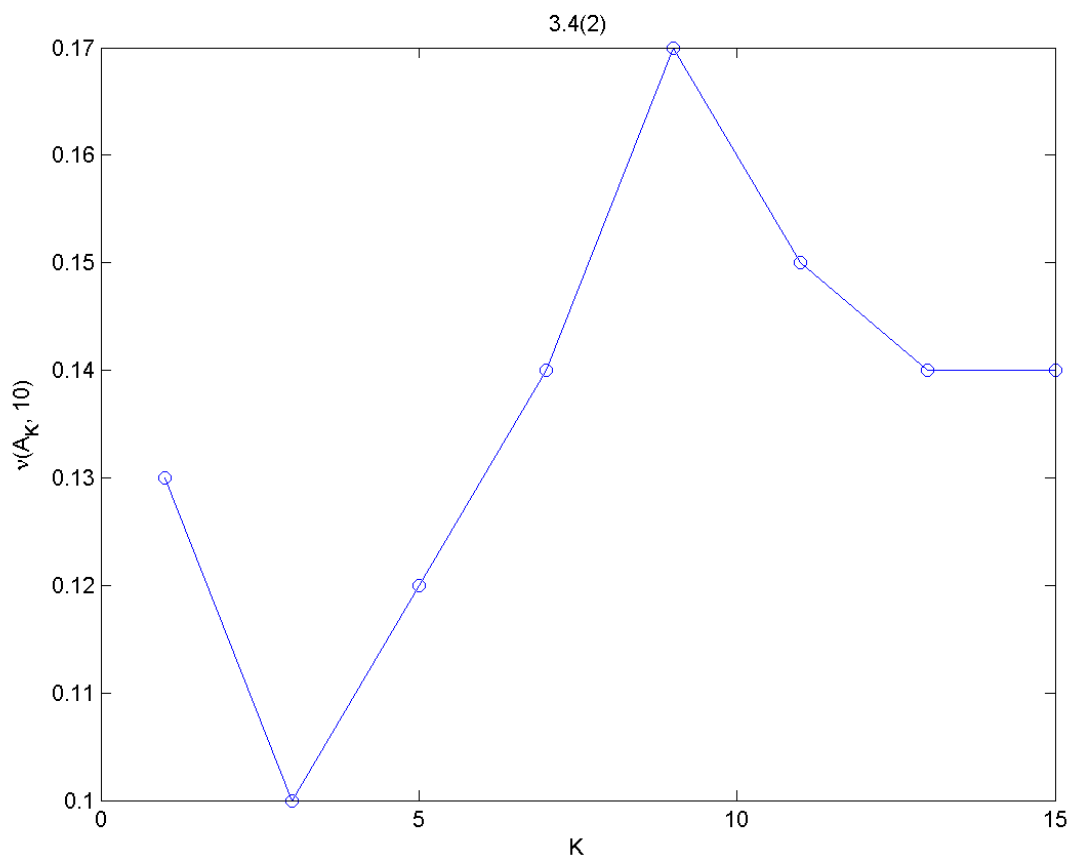


Figure 3.4(2)

	K = 1	K = 3	K = 5	K = 7	K = 9	K = 11	K = 13	K = 15
$v_c(A_K, 10)$	0.13	0.10	0.12	0.14	0.17	0.15	0.14	0.14
$\hat{\pi}(A_K(Z))$	0.13	0.125	0.135	0.149	0.155	0.170	0.182	0.192

Table 3.4(2)

We can figure out when $K = 1$, the testing error $\hat{\pi}(A_K(Z))$ and testing error of validation set $v_c(A_K, 10)$ was close. When K got bigger, the two values got more far away to each others.

However, we could see that it reflect the truth situation, because both we have a minimum testing error and cross-validation error at $K = 3$. So the cross-validation error method indeed gives us a good judgment to choose algorithm.

3.5 Mysterious B-function Leads to Bonus

Proof $B(M, N) \geq \sum_{m=0}^{M-1} \binom{M}{m}$, we show to things.

$$(1) \quad B(N, M) \geq B(N-1, M) + B(N-1, M-1)$$

Proof:

Consider the set S achieve $B(N, M)$. First, we project the vectors in S into the first $N-1$ dimensions to get $V = \{v_i\}$, where v_i 's are unique. We want to Separate V to three disjoint subset:

- 1 A_1 : There is only $(v_i, +)$ in S , but no $(v_i, -)$.
- 2 A_2 : There is only $(v_i, -)$ in S , but no $(v_i, +)$.
- 3 A_3 : Both $(v_i, +)$ and $(v_i, -)$ are in S .

By reorganizing the rows, we get

$$S = \begin{bmatrix} A_1 & + \\ A_2 & - \\ A_3 & + \\ A_3 & - \end{bmatrix}; \quad V = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix}$$

Now let $a = |A_1| + |A_2|$, $b = |A_3|$. We see that

$$|V| = a + b; \quad B(N, M) = |S| = a + 2b \quad (3.5.1)$$

(a) We first look at V :

If V is not M -incomplete, obviously S is also not M -incomplete. $\rightarrow \leftarrow$

Thus, V must be M -incomplete, and we look at V and the last column of it, V'

$$V' = \begin{bmatrix} A_1 & + \\ A_2 & - \\ A_3 & + \end{bmatrix}$$

We can see V' is N dimensions, and size of rows same as V . Thus we have

$$|V'| = a + b; \quad |V'| \geq B(N-1, M) \quad (3.5.2)$$

(b) We now look at the subset:

$$S_3 = \begin{bmatrix} A_3 & + \\ A_3 & - \end{bmatrix}$$

If A_3 is not $(M-1)$ -incomplete, then S_3 (and hence S) is not M -incomplete. $\rightarrow \leftarrow$

Thus A_3 must be $(M-1)$ -incomplete, and by look at A_3 and the last column of it, S'_3

$$S'_3 = [A_3 \quad -]$$

We can see S'_3 is N dimensions, and size of rows same as A_3 . Thus we have

$$|S'_3| = b; \quad S'_3 \geq B(N-1, M-1) \quad (3.5.3)$$

By combining (3.5.1), (3.5.2) and (3.5.3). we get the desire result.

$$(2) \quad B(N, M) \geq \sum_{m=0}^{M-1} \binom{N}{m}$$

Proof by mathematic induction:

(a) Base:

Consider the matrix $B(N, M)$ is first row and column:

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ 2 & & & \\ \vdots & & & \\ 2 & & & \end{bmatrix}_{M \times N}$$

We have $B(N, 1) = 1 \geq \sum_{m=0}^0 \binom{N}{m} = 1$, and $B(1, M) = 2 \geq \sum_{m=0}^{M-1} \binom{1}{m} = 2$ for $M \geq 2$.

(b) Induction: by part (1)

$$B(N, M) \geq B(N-1, M) + B(N-1, M-1)$$

We have

$$B(N, M) \geq \sum_{m=0}^{M-1} \binom{N-1}{m} + \sum_{m=0}^{M-2} \binom{N-1}{m}$$

By Pascal's rule

$$B(N, M) \geq \sum_{m=0}^{M-1} \binom{N}{m}$$

Proof done.