MMD PnP1

Tobias Winter

March 2025

Contents

1 Task 1: The search for ESP

1.1 (a)

Question:
[1 point(s)] What is the probability that at least 1 participant out of 1000 gets

all 6 cards right?

Answer:

The probability to guess right is $p = \frac{1}{3}$
$X \sim Bin(n, p)$ where n is the number of persons participating.
Then the probability to guess all cards correct is:

$$P(X = 6) = \left(\frac{1}{3}\right)^6 = \frac{1^6}{3^6} = \frac{1}{729} \tag{1}$$

Probability at least one person guesses all cards correct

$$P(X \geq 1) = 1 - P(X = 0) = 1 - (1 - p)^n \tag{2}$$

and for n = 1000

$$P(X \geq 1) = 1 - P(X = 0) = 1 - (1 - p)^{1000} = 1 - \left(1 - \frac{1}{729}\right)^{1000} \approx 0.747 \tag{3}$$

That means it should not be that surprising that at least one person out of 1000 will guess all cards right

1.2 (b)

Question:
[1 point(s)] What is the probability that a participant succeeds in getting 12 cards right? (6 cards in a first trial, and 6 cards in a second trial).

Answer: Assuming that the second trail is independent of the first trail

$$P(X = 12) = P(X = 6) \cdot P(X = 6) = \left(\frac{1}{3}\right)^6 \cdot \left(\frac{1}{3}\right)^6 = \left(\frac{1}{3}\right)^{12} \tag{4}$$

1.3 (c)

Question:
[2 point(s)] Assume that there actually exists a person with some form of ESP in the sense that they get 6 cards right with probability 0.01. How can we distinguish between a person with ESP and one without? Provide a description of an experiment to conduct to this end. Which parameters of the experiment characterize how certain we are about a person having ESP? Can we ever be certain that a person has ESP?

Answer:
We assume that a person with ESP has a change to guess 6 cards right in n trails is 0.01. We also assumed that $X \sim Bin(n, p)$ and because of the central

limit theorem if $n \to \infty$ we get a normal distribution and can do a hypotheses test.

$$H_0 : p = 0.01 \tag{5}$$

$$H_1 : p = \frac{1}{729} \tag{6}$$

If we choose n large enough and have a $\alpha = 0.05$ (for example) and our p-value < 0.05 (for example) than we can accept H_0 and we can dismiss H_1.

The parameters that t characterize how certain we are then our α and n. We can never me for sure that a person has ESP, we can only accept or dismiss H_0.

2 Task 2: MapReduce – Counting Palindromes

2.1 (a)

Question:
[2 point(s)] Specify the map and reduce functions for computing the counts in the form of pseudo-code. Specify the inputs and outputs for the functions.
Answer:

Algorithm 1 map(key, value)

Require: key: document name, value: text of document

```
for each word w in value do
    if is_palindromes(word) then
        yield(word, 1)
    end if
end for
```

Algorithm 2 reduce (words, count)

Require: words: found palindromes, count: count of that palindrome

```
result = 0
for each word w in value do
    result += 1
end for
yield (palindrome, n)
```

3 Task 3: MapReduce – Mean Absolute Error of Linear Regression

$$R(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{w}^T x_i| \tag{7}$$

3.1 (a)

Question:
[2 point(s)] Specify the map and reduce functions for computing R(w) in the form of pseudo-code. Specify the inputs and outputs for the functions.

Algorithm 3 map(dataset, (feature,label))

Require: dataset: dataset name, (feature,label): feature and label

1: **for** each x,y in (feature,label) **do**
2: $absolute\_error = |y_i - \boldsymbol{w}^T xi|$
3: $yield(dataset, absolute\_error)$
4: **end for**

Algorithm 4 reduce (dataset, abs_errors)

Require: dataset: tuple of x_i and y_i, abs_errors: absolute errors computed by map

1: $absolute\_error = 0$
2: $iterations = 0$
3: **for** each abs_error in abs_errors **do**
4: absolute_error += abs_error
5: iterations += 1
6: **end for**
7: $yield(dataset, \frac{absolute\_error}{iterations})$

3.2 (b)

The property that enables us to easily parallelize a least squares problem is that we only have to compute the following:

$$\left(y_i - w^T x_i\right)^2 \tag{8}$$

we do not care for $i-1$ or $i+1$. We only care about the current computation. Thats why we can give each Thread/Machine a different index to compute.

If each mapper needs to know about the results of the other mappers, that is when it gets harder to parallelize.

For example:

$$\sum_{i,j} \left(y_i - y_j - w^T(x_i - x_j)\right)^2 \tag{9}$$

where we try to compute the pairwise differences between the labels $y_i - y_j$ and the predictions $w^T(x_i - x_j)$

4 Task 4: Min-hashing for Approximate Retrieval

Assume that we want to use min-hashing for approximate retrieval of text-documents. To this end, we assume that each text-document is represented by an indicator vector (bag-of-words) indicating which words from a global dictionary (consisting of all the words in the text-documents) are present in a specific document.

Assume the following text-documents t1, . . . , t4:

1. t1 this is mining massive data
2. t2 mining massive data is cool
3. t3 machine learning is cool
4. t4 I just had a good lunch

4.1 (a)

Question:
Determine the global dictionary and represent each text-document as a bag-of-words.

Answer:
The global dictionary for t1,...,t4 is:

$0 \rightarrow$ this,
$1 \rightarrow$ is,
$2 \rightarrow$ mining,
$3 \rightarrow$ massive,
$4 \rightarrow$ data,
$5 \rightarrow$ cool,
$6 \rightarrow$ machine,
$7 \rightarrow$ learning,
$8 \rightarrow$ I,
$9 \rightarrow$ just,
$10 \rightarrow$ had,
$11 \rightarrow$ a,
$12 \rightarrow$ good,
$13 \rightarrow$ lunch.

	$t1$	$t2$	$t3$	$t4$
this	1	0	0	0
is	1	1	1	0
mining	1	1	0	0
massive	1	1	0	0
data	1	1	0	0
cool	0	1	1	0
machine	0	0	1	0
learning	0	0	1	0
I	0	0	0	1
just	0	0	0	1
had	0	0	0	1
a	0	0	0	1
good	0	0	0	1
lunch	0	0	0	1

0

4.2 (b)

Question:
Consider the hash functions

$$h_1(x) = 3x + 1 \ mod \ 8 \tag{10}$$

$$h_2(x) = 5x + 3 \ mod \ 8 \tag{11}$$

Compute the signature matrix of the documents.

Answer:

First we create the incident matrix for t1,...,t4 + the hash values for each word.

Word idx i	t_1	t_2	t_3	t_4	$h_1(i)$	$h_2(i)$
0	1	0	0	0	$(3 \cdot 0 + 1) \bmod 8 = 1$	$(5 \cdot 0 + 3) \bmod 8 = 3$
1	1	1	1	0	$(3 \cdot 1 + 1) \bmod 8 = 4$	$(5 \cdot 1 + 3) \bmod 8 = 0$
2	1	1	0	0	$(3 \cdot 2 + 1) \bmod 8 = 7$	$(5 \cdot 2 + 3) \bmod 8 = 5$
3	1	1	0	0	$(3 \cdot 3 + 1) \bmod 8 = 2$	$(5 \cdot 3 + 3) \bmod 8 = 2$
4	1	1	0	0	$(3 \cdot 4 + 1) \bmod 8 = 5$	$(5 \cdot 4 + 3) \bmod 8 = 7$
5	0	1	1	0	$(3 \cdot 5 + 1) \bmod 8 = 0$	$(5 \cdot 5 + 3) \bmod 8 = 4$
6	0	0	1	0	$(3 \cdot 6 + 1) \bmod 8 = 3$	$(5 \cdot 6 + 3) \bmod 8 = 1$
7	0	0	1	0	$(3 \cdot 7 + 1) \bmod 8 = 6$	$(5 \cdot 7 + 3) \bmod 8 = 6$
8	0	0	0	1	$(3 \cdot 8 + 1) \bmod 8 = 1$	$(5 \cdot 8 + 3) \bmod 8 = 3$
9	0	0	0	1	$(3 \cdot 9 + 1) \bmod 8 = 4$	$(5 \cdot 9 + 3) \bmod 8 = 0$
10	0	0	0	1	$(3 \cdot 10 + 1) \bmod 8 = 7$	$(5 \cdot 10 + 3) \bmod 8 = 5$
11	0	0	0	1	$(3 \cdot 11 + 1) \bmod 8 = 2$	$(5 \cdot 11 + 3) \bmod 8 = 2$
12	0	0	0	1	$(3 \cdot 12 + 1) \bmod 8 = 5$	$(5 \cdot 12 + 3) \bmod 8 = 7$
13	0	0	0	1	$(3 \cdot 13 + 1) \bmod 8 = 0$	$(5 \cdot 13 + 3) \bmod 8 = 4$

For a document t_j, we look at all row indices i where t_j has a 1. Then:

$$signiture(t_j, h_k) = \min_{i \in t_j} \{\, h_k(i) \}. \tag{12}$$

Document $t_1 = \{0, 1, 2, 3, 4\}$**.**

$$\begin{aligned} h_1(i) \text{ for } i \in t_1 &= \{\, 1, 4, 7, 2, 5\}, \quad \min = 1, \\ h_2(i) \text{ for } i \in t_1 &= \{\, 3, 0, 5, 2, 7\}, \quad \min = 0. \end{aligned}$$

Hence signature(t_1) = (1, 0).

Document $t_2 = \{1, 2, 3, 4, 5\}$**.**

$$\begin{aligned} h_1(i) \text{ for } i \in t_2 &= \{\, 4, 7, 2, 5, 0\}, \quad \min = 0, \\ h_2(i) \text{ for } i \in t_2 &= \{\, 0, 5, 2, 7, 4\}, \quad \min = 0. \end{aligned}$$

Hence signature(t_2) = (0, 0).

Document $t_3 = \{1, 5, 6, 7\}$**.**

$$\begin{aligned} h_1(i) \text{ for } i \in t_3 &= \{\, 4, 0, 3, 6\}, \quad \min = 0, \\ h_2(i) \text{ for } i \in t_3 &= \{\, 0, 4, 1, 6\}, \quad \min = 0. \end{aligned}$$

Hence signature(t_3) = (0, 0).

Document $t_4 = \{8, 9, 10, 11, 12, 13\}$**.**

$$h_1(i) \text{ for } i \in t_4 = \{\,1, 4, 7, 2, 5, 0\}, \quad \min = 0,$$
$$h_2(i) \text{ for } i \in t_4 = \{\,3, 0, 5, 2, 7, 4\}, \quad \min = 0.$$

Hence signature$(t_4) = (0,\, 0)$.
The signature matrix is:

	t_1	t_2	t_3	t_4
h_1	1	0	0	0
h_2	0	0	0	0

4.3 (c)

Question:
Compute the similarity of the documents. What is the most similar document to t1? What about t3? What is the similarity of t4 to the other documents?

Answer:

I computet the similaritys between each document and put them into a matrix using the Jaccard similarity:

$$Sim(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{13}$$

	t_1	t_2	t_3	t_4
t_1	1	$\frac{2}{3}$	$\frac{1}{8}$	0
t_2	$\frac{2}{3}$	1	$\frac{2}{7}$	0
t_3	$\frac{1}{8}$	$\frac{2}{7}$	1	0
t_4	0	0	0	1

- **For t_1:** most similar is t_2 (similarity = 2/3).
- **For t_2:** most similar is t_1 (similarity = 2/3).
- **For t_3:** most similar is t_2 (similarity = 2/7).
- **For t_4:** similarity = 0 to all other documents, i.e. no overlap.

5 Task 5: Curse of Dimensionality

When the number of features p is large, there tends to be a deterioration in the performance of k nearest neighbor and other local approaches that perform prediction using only observations that are near the test observation for which a prediction must be made. This phenomenon is known as the curse of dimensionality, and it ties into the fact that so-called non-parametric approaches often perform poorly when p is large. We will now investigate this curse.

5.1 (a)

Question:
Suppose that we have a set of observations, each with measurements on p = 1 feature, X. We assume that X is uniformly (evenly) distributed on [0, 1]. Associated with each observation is a response value. Suppose that we wish to predict a test observation's response using only observations that are within 5% of the range of X closest to that test observation. For instance, in order to predict the response for a test observation with X = 0.6, we will use observations in the range [0.575, 0.625]. On average, what fraction of the available observations will we use to make the prediction?

Answer:
The observations are $X \sim Uni(a, b)$ between [0,1]. And we want to calculate the average fraction of observations around $X = 0.6$ using only 5% of the total observations. Meaning we use the interval [x - 0.025, x + 0.025] $\rightarrow$ [0.575, 0.625]. Meaning the fraction of available observations is:

$$\frac{0.05}{1} = 0.05 \tag{14}$$

5.2 (b)

Question:
Now suppose that we have a set of observations, each with measurements on p = 2 features, X1 and X2. We assume that (X1, X2) are uniformly distributed on [0, 1] $\times$ [0, 1]. We wish to predict a test observation's response using only observations that are within 5% of the range of X1 and within 5% of the range of X2 closest to that test observation. For instance, in order to predict the response for a test observation with X1 = 0.6 and X2 = 0.35, we will use observations in the range [0.575, 0.625] for X1 and in the range [0.325, 0.75] for X2. On average, what fraction of the available observations will we use to make the prediction

Answer:

This is the same principle as before but we now have two dimensions.

$$0.05 \cdot 0.05 = 0.05^2 = 0.0025 \tag{15}$$

In general the expected fraction of available observations is:

$$(1 - \epsilon)^p \tag{16}$$

5.3 (c)

Question:
Now suppose that we wish to make a prediction for a test observation by creating a p-dimensional hypercube centered around the test observation that contains, on average, 5% of the training observations. For p $\in \{1, 2, 5, 10, 100\}$, what is the length of each side of the hypercube? Comment on your answer. Note: A

hypercube is a generalization of a cube to an arbitrary number of dimensions. When p = 1, a hypercube is simply a line segment, when p = 2 it is a square, and when p = 100 it is a 100-dimensional cube.

Answer:
In the lecture we saw the following:

$$E[B] = N(1-\epsilon)^p \tag{17}$$

Where B are the neighbors, p is the dimension and N is the total number of observations. We can see that $(1-\epsilon)^p$ is the side length of a hypercube. We kind of saw this in b) where we had [0,1] x [0,1]. Here we could have drawn a square to represent the total area of our observations.

We

To now find the side length of the hypercube in p dimension we just rearrange and get:

$$\frac{E[B]}{N} = (1-\epsilon)^p \tag{18}$$

$$\left(\frac{E[B]}{N}\right)^{\frac{1}{p}} = (1-\epsilon) \tag{19}$$

If we now plug in the values for $\frac{E[B]}{N} = 0.05$ (since we only care about 5 %) and $p_i \in \{1, 2, 5, 10, 100\}$ we get the following side lengths:

p	$1-\epsilon$
1	0.05
2	0.2236
5	0.5493
10	0.7411
100	0.9705

6 Task 6: MapReduce – Matrix-vector Multiplication

6.1 (a)

Question:
Explain how map-reduce can be applied when the matrix M is distributed among multiple-machines but the vector v can be fit into the main memory of a compute node. Specify the map and reduce functions for computing the matrixvector product in the form of pseudo-code. Specify the inputs and outputs for the functions.

Answer:
As of the book MMDS section 2.3.1 we want to compute $\boldsymbol{x}$ where each x_i is given by:

$$x_i = \sum_{j=1}^{n} m_{ij} v_j \tag{20}$$

In the map phase, for each record (i, j, $m_{i,j}$) in a chunk of M, we multiply m_{ij} by v_j.

In the reduce phase we just collect all the products and sum them up.

Algorithm 5 map (key, value)

Require: key: name of the matrix, value: triple of (i, j, $m_{i,j}$)
{We assume that v is global or also passed in the value. But the whole v is in each mappers memory}
1: i, j, $m_{i,j}$ = value
2: $patrial\_product = m_{i,j} \cdot v[j]$
3: $yield(i, patrial\_product)$

Algorithm 6 reduce (i, *partial_products*)

Require: i: current row, *partial_products*: the products computed by the mapper
1: sum = 0
2: **for** value in *partial_products* **do**
3: sum += value
4: **end for**
5: yield(i, sum) {this is the x_i from above}

6.2 (b)

Question:
Consider the case in which v is too large to be read into the main memory of a compute node and must be partitioned into stripes as explained in section 2.3.2 in the MMDS book. Specify the map and reduce functions for computing the matrix-vector product in the form of pseudo-code. Specify the inputs and outputs for the functions.

Answer:
According to the MMDS book we can nearly take the same approach as if v fits fully into memory, but we first need to partition the matrix M and the vector v into N vertical stripes each with equal length. Therefore each pice of v has length $\frac{n}{N}$ and we can express each pice as $v^{(1)}, v^{(2)}, ..., v^{(N)}$. M is also partition into N vertical stripes from $M^{(1)}, ..., M^{(N)}$. Now we can do the same calculation but we only calculate the partial product of each stripe.

Algorithm 7 map (*stripe_index*, value)

Require: *stripe_index*: is the m-th stripe index value: triple of (i, j, $m_{i,j}$) of the stripe {We again assume that $v^{(stripe\_index)}$ is global or also passed in the value. So it is in memory}

1: i, j, $m_{i,j}$ = value
2: $patrial\_product = m_{i,j} \cdot v[j]$
3: $yield(i, patrial\_product)$ {yielding the patrial contribution to x_i with the key i}

Algorithm 8 reduce (i, *partial_products*)

Require: i: current row, *partial_products*: all partial sums contributed from the N stripes of M for this row

1: sum = 0
2: **for** value in *partial_products* **do**
3: sum += value
4: **end for**
5: yield(i, sum) {sum is now the same x_i from above}