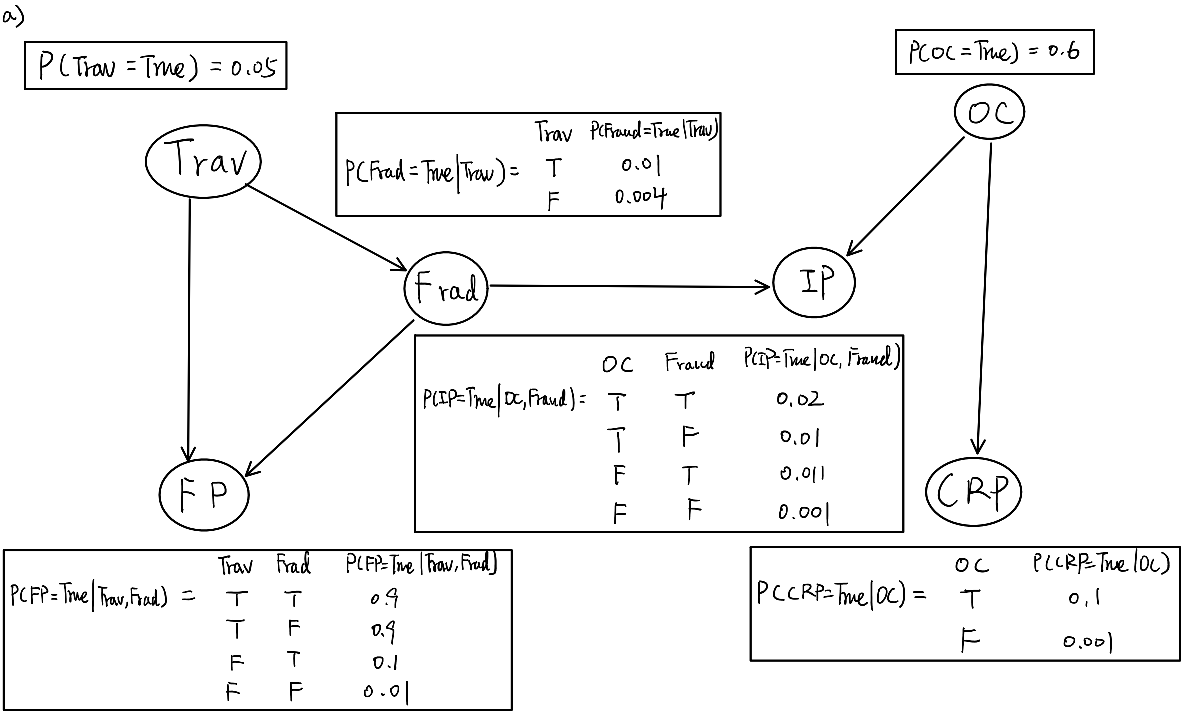
A3

2022年7月2日 星期六　　上午11:08

a)



b)

prior probality: $P(\text{Fraud} = \text{True})$

$f_0(\text{Trav}) =$

Trav	f
T	0.05
F	0.95

$f_1(\text{Trav}, \text{Fraud}) =$

Trav	Fraud	f
T	T	0.01
T	F	0.99
F	T	0.004
F	F	0.996

$f_2(\text{Trav}, \text{Fraud}, \text{FP}) =$

Trav	Fraud	FP	f
T	T	T	0.9
T	T	F	0.1
T	F	T	0.9
T	F	F	0.1
F	T	T	0.1
F	T	F	0.9
F	F	T	0.01
F	F	F	0.99

$f_3(\text{OC}, \text{Fraud}, \text{IP})$

OC	Fraud	IP	
T	T	T	0.02
T	T	F	0.98
T	F	T	0.01
T	F	F	0.99
F	T	T	0.011
F	T	F	0.989
F	F	T	0.001
F	F	F	0.999

$f_4(\text{OC}) =$

OC	f
T	0.6
F	0.4

$f_5(\text{OC}, \text{CRP}) =$

OC	CRP	f
T	T	0.1
T	F	0.9
F	T	0.001
F	F	0.999

$$P(\text{Fraud}) \propto \sum_{\text{Trav}, \text{FP}, \text{IP}, \text{OC}, \text{CRP}} f_0(\text{Trav}) f_1(\text{Trav}, \text{Fraud})\, f_2(\text{Trav}, \text{Fraud}, \text{FP})\, f_3(\text{OC}, \text{Fraud}, \text{IP})\, f_4(\text{OC})\, f_5(\text{OC}, \text{CRP})$$

$$= \sum_{\text{CRP}} \sum_{\text{OC}} f_5(\text{OC}, \text{CRP}) f_4(\text{OC}) \sum_{\text{IP}} f_3(\text{OC}, \text{Fraud}, \text{IP}) \sum_{\text{FP}} \sum_{\text{Trav}} f_0(\text{Trav}) f_1(\text{Trav}, \text{Fraud})\, f_2(\text{Trav}, \text{Fraud}, \text{FP})$$

$$= \sum_{\text{CRP}} \sum_{\text{OC}} f_5(\text{OC}, \text{CRP}) f_4(\text{OC}) \sum_{\text{IP}} f_3(\text{OC}, \text{Fraud}, \text{IP}) \sum_{\text{FP}} f_6(\text{Fraud}, \text{FP})$$

Where $f_6(\text{Fraud}, \text{FP}) = \sum_{\text{Trav}} f_0(\text{Trav}) f_1(\text{Trav}, \text{Fraud})\, f_2(\text{Trav}, \text{Fraud}, \text{FP})$

Where $f_6(Fraud, FP) = \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP)$

=

Fraud	FP	f
T	T	0.00083
T	F	0.00347
F	T	0.054012000000000004
F	F	0.941688

$= \sum_{CRP} \sum_{OC} f_5(OC, CRP) f_4(OC) \sum_{IP} f_3(OC, Fraud, IP)\ f_7(Fraud)$

where $f_7(Fraud) = \sum_{FP} f_6(Fraud, FP) =$

Fraud	f
T	0.0043
F	0.9957

$= f_7(Fraud) \sum_{CRP} \sum_{OC} f_5(OC, CRP) f_4(OC)\ f_8(OC, Fraud)$

where $f_8(OC, Fraud) = \sum_{IP} f_3(OC, Fraud, IP)$

OC	Fraud	f
T	T	1.0
T	F	1.0
F	T	1.0
F	F	1.0

$= f_7(Fraud) \sum_{CRP} f_9(CRP, Fraud)$

where $f_9(CRP, Fraud) = \sum_{OC} f_5(OC, CRP) f_4(OC)\ f_8(OC, Fraud)$

CRP	Fraud	f
T	T	0.06039999999999995
T	F	0.06039999999999995
F	T	0.9396
F	F	0.9396

$= f_7(Fraud)\ f_{10}(Fraud)$

where $f_{10}(Fraud) = \sum_{CRP} f_9(CRP, Fraud)$

Fraud	f
T	1.0
F	1.0

$= f_{11}(Fraud)$

where $f_{11}(Fraud) = f_7(Fraud)\ f_{10}(Fraud)$

Fraud	f
T	0.0043
F	0.9957

So $P(Fraud) =$

Fraud	P(Fraud)
T	$0.0043/(0.0043+0.9957) = 0.0043$
F	$0.9957/(0.0043+0.9957) = 0.9957$

So $P(Fraud = True) = 0.0043 = 0.43\%$

$P(Fraud = True \mid FP = True, IP = False, CRP = True)$

$f_0(Trav) =$

Trav	f
T	0.05
F	0.95

$f_1(Trav, Fraud) =$

Trav	Fraud	f
T	T	0.01
T	F	0.99
F	T	0.004
F	F	0.996

$f_{12}(Trav, Fraud) = f_2(Trav, Fraud, FP{=}True)$

=

Trav	Fraud	f
T	T	0.9
T	F	0.9
F	T	0.1
F	F	0.01

$f_{13}(OC, Fraud) = f_3(OC, Fraud, IP{=}True)$

=

OC	Fraud	f
T	T	0.98
T	F	0.99
F	T	0.989
F	F	0.999

$f_4(OC) =$

OC	f
T	0.6
F	0.4

$f_{14}(OC) = f_5(OC, CRP=True) =$

OC	f
T	0.1
F	0.001

$P(Fraud \mid FP=True, IP=False, CRP=True)$

$$\propto \sum_{Trav, OC} f_0(Trav) f_1(Trav, Fraud) f_{12}(Trav, Fraud) f_{13}(OC, Fraud) f_4(OC) f_{14}(OC)$$

$$= \sum_{OC} f_{13}(OC, Fraud) f_4(OC) f_{14}(OC) \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_{12}(Trav, Fraud)$$

$$= \sum_{OC} f_{13}(OC, Fraud) f_4(OC) f_{14}(OC) f_{15}(Fraud)$$

where $f_{15}(Fraud) = \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_{12}(Trav, Fraud)$

=

Fraud	f
T	0.00083
F	0.054012000000000004

$$= f_{15}(Fraud) \cdot f_{16}(Fraud)$$

where $f_{16}(Fraud) = \sum_{OC} f_{13}(OC, Fraud) f_4(OC) f_{14}(OC)$

=

Fraud	f
T	0.0591956
F	0.0597996

$$= f_{17}(Fraud)$$

where $f_{17}(Fraud) = f_{15}(Fraud) f_{16}(Fraud) =$

Fraud	f
T	4.9132348e-05
F	0.0032298959952000005

Normalize the above result, we get

$P(Fraud \mid FP=True, IP=False, CRP=True) =$

Fraud	f
T	0.014983813147541077
F	0.985016186852459

So $P(Fraud=True \mid FP=True, IP=False, CRP=True) \approx 1.50\%$

c) New evidence: $Trav = True$.

$P(Fraud=True \mid FP=True, IP=False, CRP=True, Trav=True)$

$f_{18}() = 0.05$

$f_{19}(Fraud) = f_1(Trav=True, Fraud) =$

Fraud	f
T	0.01
F	0.99

$f_{20}(Fraud) = f_2(Trav=True, Fraud, FP=True)$

=

Fraud	f
T	0.9
F	0.9

$f_{13}(OC, Fraud) = f_3(OC, Fraud, IP=False)$

=

OC	Fraud	f
T	T	0.98
T	F	0.99
F	T	0.989
F	F	0.999

$f_4(OC) =$

OC	f
T	0.6
F	0.4

$f_{14}(OC) = f_5(OC, CRP=True) =$

OC	f
T	0.1
F	0.001

$P(Fraud \mid FP=True, IP=False, CRP=True, Trav=True)$

$$\propto \sum_{OC} f_{18}() f_{19}(Fraud) f_{20}(Fraud) F_{13}(OC, Fraud) f_4(OC) f_{14}(OC)$$

$P(Fraud \mid FP=True, IP=False, CRP=True, Trav=True)$

$\propto \sum_{OC} f_{18}()\, f_{19}(Fraud)\, f_{20}(Fraud)\, f_{13}(OC, Fraud)\, f_4(OC)\, f_{14}(OC)$

$= f_{18}()\, f_{19}(Fraud)\, f_{20}(Fraud)\, f_{21}(Fraud)$

where $f_{21}(Fraud) = \sum_{OC} f_{13}(OC, Fraud)\, f_4(OC)\, f_{14}(OC)$

Fraud	f
T	0.0591956
F	0.0597996

$= f_{22}(Fraud)$

Where $f_{22}(Fraud) = f_{18}()\, f_{19}(Fraud)\, f_{20}(Fraud)\, f_{21}(Fraud)$

Fraud	f
T	0.0005327604000000001
F	0.0532814436

Normalize the factor above, we get:

$P(Fraud \mid FP=True, IP=False, CRP=True, Trav=True) =$

Fraud	f
T	0.009899995919292984
F	0.9901000040807071

So $P(Fraud=True \mid FP=True, IP=False, CRP=True, Trav=True) = 0.99\%$

d) Before we start this question, I need to make some assumptions. There are six variables in total, and five of them can be used as evidence since we want to query Fraud. However, most of those variables are not in our control given that we are just an employee in the credit card company.

OC: There is nothing I can do to change this evidence since it is a fact of the card holder

Fraud: It is the query

Trav: I cannot affect whether the card holder is travelling or not

FP: Assume we do not know the card holder, then we also do not know if the purchase is a FP

IP: Given that we want to purchase something online, we need this to be True

CRP: this is a variable that we can decide.

Now, if we do not do anything before the purchase, then the query becomes:

$P(Fraud = True \mid IP = True)$

$f_0(Trav) =$

Trav	f
T	0.05
F	0.95

$f_1(Trav, Fraud) =$

Trav	Fraud	f
T	T	0.01
T	F	0.99
F	T	0.004
F	F	0.996

$f_2(Trav, Fraud, FP) =$

Trav	Fraud	FP	f
T	T	T	0.9
T	T	F	0.1
T	F	T	0.9
T	F	F	0.1
F	T	T	0.1
F	T	F	0.9
F	F	T	0.01
F	F	F	0.99

$f_3(OC, Fraud, IP=True) = f_6(OC, Fraud) =$

OC	Fraud	f
T	T	0.02
T	F	0.01
F	T	0.011
F	F	0.001

$f_5(OC, CRP) =$

OC	CRP	f
T	T	0.1
T	F	0.9
F	T	0.001
F	F	0.999

$f_4(OC) =$

OC	f
T	0.6
F	0.4

$$P(Fraud \mid IP = True) \propto \sum_{Trav,FP,OC,CRP} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP) f_6(OC, Fraud) f_4(OC) f_5(OC, CRP)$$

$$= \sum_{CRP} \sum_{OC} f_6(OC, Fraud) f_4(OC) f_5(OC, CRP) \sum_{FP} \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP)$$

$$= \sum_{CRP} \sum_{OC} f_6(OC, Fraud) f_4(OC) f_5(OC, CRP) \sum_{FP} f_7(Fraud, FP)$$

Where

$$f_7(Fraud, FP) = \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP)$$

Fraud	FP	f
T	T	0.00083
T	F	0.00347
F	T	0.054012000000000004
F	F	0.941688

$$= f_8(Fraud) \sum_{CRP} \sum_{OC} f_6(OC, Fraud) f_4(OC) f_5(OC, CRP)$$

Where

$$f_8(Fraud) = \sum_{FP} f_7(Fraud, FP)$$

Fraud	f
T	0.0043
F	0.9957

$$= f_8(Fraud) \sum_{CRP} f_9(Fraud, CRP)$$

Where

$$f_9(Fraud, CRP) = \sum_{OC} f_6(OC, Fraud) f_4(OC) f_5(OC, CRP)$$

Fraud	CRP	f
T	T	0.0012044000000000002
T	F	0.0151956
F	T	0.0006004000000000001
F	F	0.0057996

$$= f_8(Fraud) f_{10}(Fraud)$$

Where

$$f_{10}(Fraud) = \sum_{CRP} f_9(Fraud, CRP)$$

Fraud	f
T	0.0164
F	0.0064

$$= f_{11}(Fraud)$$

Where

$$f_{11}(Fraud) = f_8(Fraud) f_{10}(Fraud)$$

Fraud	f
T	7.052000000000001e-05
F	0.006372480000000001

We normalize the above factor to get:

$P(Fraud \mid IP = True) =$

Fraud	f
T	0.010945211857830203
F	0.9890547881421697

This means that there is a possibility of 0.0109 that this transaction will be treated as a fraudulent transaction.

However, what I can do is to make a computer related purchase prior to this purchase, which will add an evidence that CRP = True, so the query becomes:

P (Fraud = True | IP = True , CRP = True)

f_0 (Trav) =

Trav	f
T	0.05
F	0.95

f_1 (Trav , Fraud) =

Trav	Fraud	f
T	T	0.01
T	F	0.99
F	T	0.004
F	F	0.996

f_2 (Trav, Fraud, FP) =

Trav	Fraud	FP	f
T	T	T	0.9
T	T	F	0.1
T	F	T	0.9
T	F	F	0.1
F	T	T	0.1
F	T	F	0.9
F	F	T	0.01
F	F	F	0.99

f_3 (OC, Fraud, IP = True) = f_6 (OC, Fraud)

OC	Fraud	f
T	T	0.02
T	F	0.01
F	T	0.011
F	F	0.001

f_4 (OC) =

OC	f
T	0.6
F	0.4

f_{12}(OC) = f_7(OC, CRP= True) =

OC	f
T	0.1
F	0.001

$$P(Fraud \mid IP = True, CRP = True)$$

$$\propto \sum_{Trav,FP,OC} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP) f_6(OC, Fraud) f_4(OC) f_{12}(OC)$$

$$= \sum_{OC} f_6(OC, Fraud) f_4(OC) f_{12}(OC) \sum_{FP} \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP)$$

$$= \sum_{OC} f_6(OC, Fraud) f_4(OC) f_{12}(OC) \sum_{FP} f_7(Fraud, FP)$$

Where

$$f_7(Fraud, FP) = \sum_{Trav} f_0(Trav) f_1(Trav, Fraud) f_2(Trav, Fraud, FP)$$

Fraud	FP	f
T	T	0.00083
T	F	0.00347
F	T	0.054012000000000004
F	F	0.941688

$$= f_8(Fraud) \sum_{OC} f_6(OC, Fraud) f_4(OC) f_{12}(OC)$$

Where

$$f_8(Fraud) = \sum_{FP} f_7(Fraud, FP)$$

Fraud	f
T	0.0043
F	0.9957

$$= f_8(Fraud) f_{13}(Fraud)$$

Where

$$f_{13}(Fraud) = \sum_{OC} f_6(OC, Fraud) f_4(OC) f_{12}(OC)$$

Fraud	f
T	0.0012044000000000002
F	0.0006004000000000001

$$= f_{14}(Fraud)$$

Where

$$f_{14}(Fraud) = f_8(Fraud) f_{13}(Fraud)$$

Fraud	f
T	5.178920000000001e-06
F	0.00059781828

We normalize the above factor to get:

$P(Fraud \mid IP = True, CRP = True) =$

Fraud	f
T	0.00858863026229641
F	0.9914113697377036

This means that there is a possibility of 0.00859 that this transaction will be treated as a fraudulent transaction.

We can see that after I do the CRP (a computer related purchase), the probability that my target purchase on the internet has dropped from 1.09% to 0.859%. Therefore, in order to reduce the possibility that the transaction will be rejected as a possible fraud, I will make a computer related purchase prior to the target purchase.

PartB

a. Code is

```
from collections import defaultdict
import sys
import numpy as np
ATHEISM = 0
BOOKS = 1
if __name__ == "__main__":
    # Parse the input parameter to get the data folder, using current working directory as default
    args = sys.argv[1:]
    data_folder = args[0] if len(args) > 0 else '.'
    if data_folder.endswith("/") or data_folder.endswith("\\"):
        data_folder = data_folder[:-1]
    # Load data
    train_data = np.loadtxt(f"{data_folder}/trainData.txt", delimiter=" ")
    train_label = np.loadtxt(f"{data_folder}/trainLabel.txt", delimiter=" ")
    train_label = np.array([int(i - 1) for i in train_label])
    test_data = np.loadtxt(f"{data_folder}/testData.txt", delimiter=" ")
    test_label = np.loadtxt(f"{data_folder}/testLabel.txt", delimiter=" ")
    test_label = np.array([int(i - 1) for i in test_label])
    words = []              # The index of words
    words_mapping = []      # The mapping of words for printing
    with open(f"{data_folder}/words.txt") as f:
        words_mapping = [line.strip() for line in f]
    with open(f"{data_folder}/words.txt") as f:
        words = range(1, 1 + len(f.readlines()))

    # Parse the train data into a dict, key: doc_id, value: set of word_id for better performance
    train_dict = defaultdict(set)
    for (doc_id, word_id) in train_data:
        train_dict[int(doc_id)].add(int(word_id))
    test_dict = defaultdict(set)
    for (doc_id, word_id) in test_data:
        test_dict[int(doc_id)].add((int(word_id)))
    docs = list(train_dict.keys())
    docs.sort()
    docs = np.array(docs)
    train_label = train_label[docs - 1]
    parsed_train_data = np.array([[ 1 if word_id in train_dict[doc_id] else 0 for word_id in words] for doc_id in
docs])
    atheism_data = parsed_train_data[train_label == ATHEISM]
    books_data = parsed_train_data[train_label == BOOKS]

    theta_c = sum(train_label) / len(train_label)
    theta_1s = (np.sum(atheism_data, axis=0) + 1) / ( len(atheism_data) + 2)
    theta_2s = (np.sum(books_data, axis=0) + 1) / ( len(books_data) + 2)

    discrimination = np.abs(np.log(theta_1s) -  np.log(theta_2s))
    top_discri_ids = (-discrimination).argsort()[:10]
    top_discri_words = [words_mapping[i] for i in top_discri_ids]
    print("The 10 most discriminative word features are:")
    print(top_discri_words)
    def predict(docs_data):
        y_pred = [0] * len(docs_data)
        for i, x in enumerate(docs_data):
            atheism_prob = np.sum(x * np.log(theta_1s) + (1-x) * np.log(1-theta_1s)) + np.log(theta_c)
            books_prob = np.sum(x * np.log(theta_2s) + (1-x) * np.log(1-theta_2s)) + np.log(1-theta_c)
            y_pred[i] = ATHEISM if atheism_prob > books_prob else BOOKS
        return np.array(y_pred)
    train_acc = np.sum(train_label == predict(parsed_train_data)) / len(train_label)
```

```python
    print(f"Train accuracy: {train_acc}")
    test_docs = list(test_dict.keys())
    test_docs.sort()
    test_docs = np.array(test_docs)
    test_data = np.array([[ 1 if word_id in test_dict[doc_id] else 0 for word_id in words] for doc_id in
test_docs])
    test_label = test_label[test_docs - 1]
    test_acc = np.sum(test_label == predict(test_data)) / len(test_label)
    print(f"Test accuracy: {test_acc}")
```

b. The print out for the 10 most discriminative word features are:

    The 10 most discriminative word features are:
    ['christian', 'religion', 'atheism', 'books', 'christians', 'library', 'religious', 'novel', 'libraries', 'beliefs']

   So the words are: christian, religion, atheism, books, christians, library, religious, novel, libraries, beliefs
   I think they are good features. From the ML, we know that the probability used here represents the number of times each word appears in the article of certain label, so if the difference is larger, it means that the word is much more likely to appear in one type of articles than the other one.
   Besides, naturally, the meaning of these words seem to strongly represent the type/label of the articles. For example, the words christian, religion, atheism, christians, religious, beliefs are highly related to the Atheism topic. When one see those words in a post, he will probably identify that post to be something related to religion. Although there could be many categories that are related to religions, in our case, it will be Atheism rather than Books.

c. The training accuracy is 0.915068493150685, which is around 91.51%
   The test accuracy is 0.7489655172413793, which is around 74.90%

d. This is probably not a reasonable assumption because we need to consider how natural language works. People tends to use words with similar meanings. And, as I suggested earlier, words can suggest a certain topic, which, in turn, means that if we see some words, then it is possible that the article is talking about a topic, so other words that suggests this topic are more likely to appear. For example, if I see the word "library" in an article, then I would probably expect that I will also see the word "book". In other words, the semantics of each word make dependent. Another example can be that, once I see the word "station", I know there is probably some word before it like "police station" or "train station", thus the probability of word "train"/"police" now depends on the probability of word "station"

e. We can also try learn the architecture of the model. In the previous discussion, we are using a simple Bayesian network that the children nodes are independent from each other. However, we can somehow assume some dependency by adding edges between those child nodes. Then, we can use the Bayesian network parameter learning by having parameters for each combination of variable and its parent, then we can do ML again.

f. In this questions, the hypothesis is the θ parameters. For ML, we only need to find θ which maximize the $P(d|\theta)$ and we can do this by deriving a closed form fomula for the probablity and differentiate it. However, for MAP, we need to find θ which maximize the $P(\theta|d) \propto P(d|\theta)P(\theta)$, and this means that we need to have a (prior) distribution over θ. And this can be hard to find for some complicated model. However, in ourcase, we might potentially use the binary model and get the distribution over hypothesis θ