Politecnico di Milano
School of Industrial and Information Engineering

Data Mining and Text Mining

February 5, 2020

NAME

CODICE PERSONA/ID

GENERAL INSTRUCTIONS

- **Answers must be clearly written inside the answer box designated for each problem.**
- **All the answers must be adequately motivated.**
- **Pencils are not allowed. The exam consists of 7 sheets of paper. It must be returned with all the 7 sheets. No any other sheet can be added. No sheet can be removed.**
- **This is a closed-book/closed-notes exam.**
- **Only non-programmable calculators are allowed.**
- **Notes/books/mobile phones are not allowed.**
- **If you are caught using forbidden material, the exam will immediately end and an RP grade will be recorded; then, your Data Mining exam will consist of an oral examination from then on.**

COURSE PROJECT SCORE

FINAL TIME

GRADES

1	2	3
4	5	6

SCORING

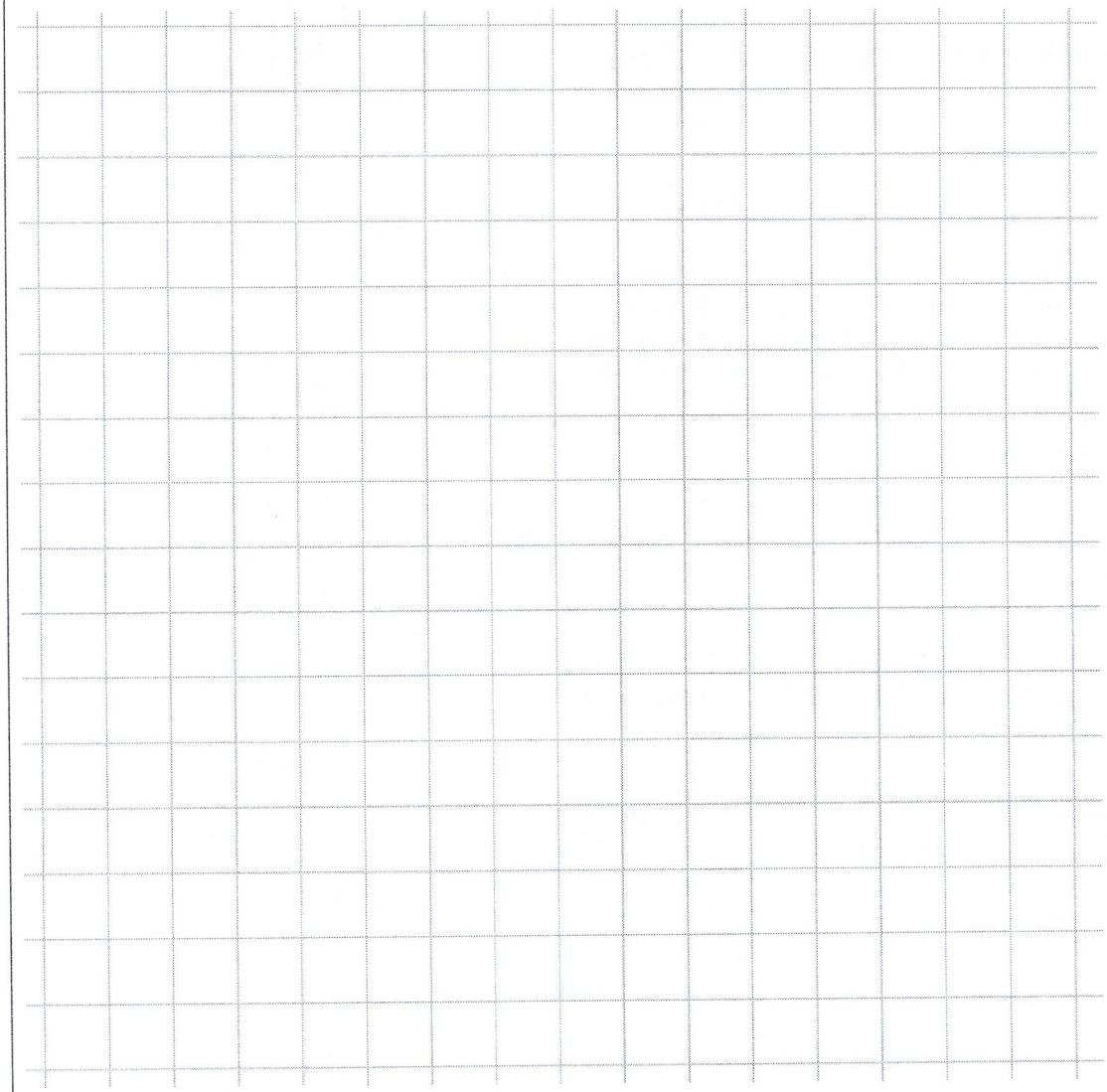
- **A problem left unsolved will amount to zero points.**
- **A completely wrong solution will amount to -3 points**

STUDENTS WHO DID THE COURSE PROJECT HAVE 1:40h TO SOLVE PROBLEMS 1, 2, 3, AND 4

ALL THE OTHER STUDENTS HAVE 2:20h TO SOLVE ALL THE SIX PROBLEMS

Problem 1 (6 points). Consider the following data set in which columns **x** and **y** represent the input variables while column "**label**" contains the cluster label that has been assigned to each example by applying k-means with k=2 and Euclidean distance. Compute for each example the silhouette coefficient using Euclidean distance.

	x	y	label
0	0.0	4.0	0
1	5.0	4.0	1
2	3.0	2.0	1
3	0.0	0.0	0
4	5.0	0.0	1



Problem 1 (continued).

x	y	label	a	b	Silhouette Coefficient
0.0	4.0	0			
5.0	4.0	1			
3.0	2.0	1			
0.0	0.0	0			
5.0	0.0	1			

The silhouette coefficient for point p is computed as,

SC(p) = b(p) – a(p)/max(a(p),b(p))

Let's call p0 the data point in row 0, and p4 the data point in row 4, we have,

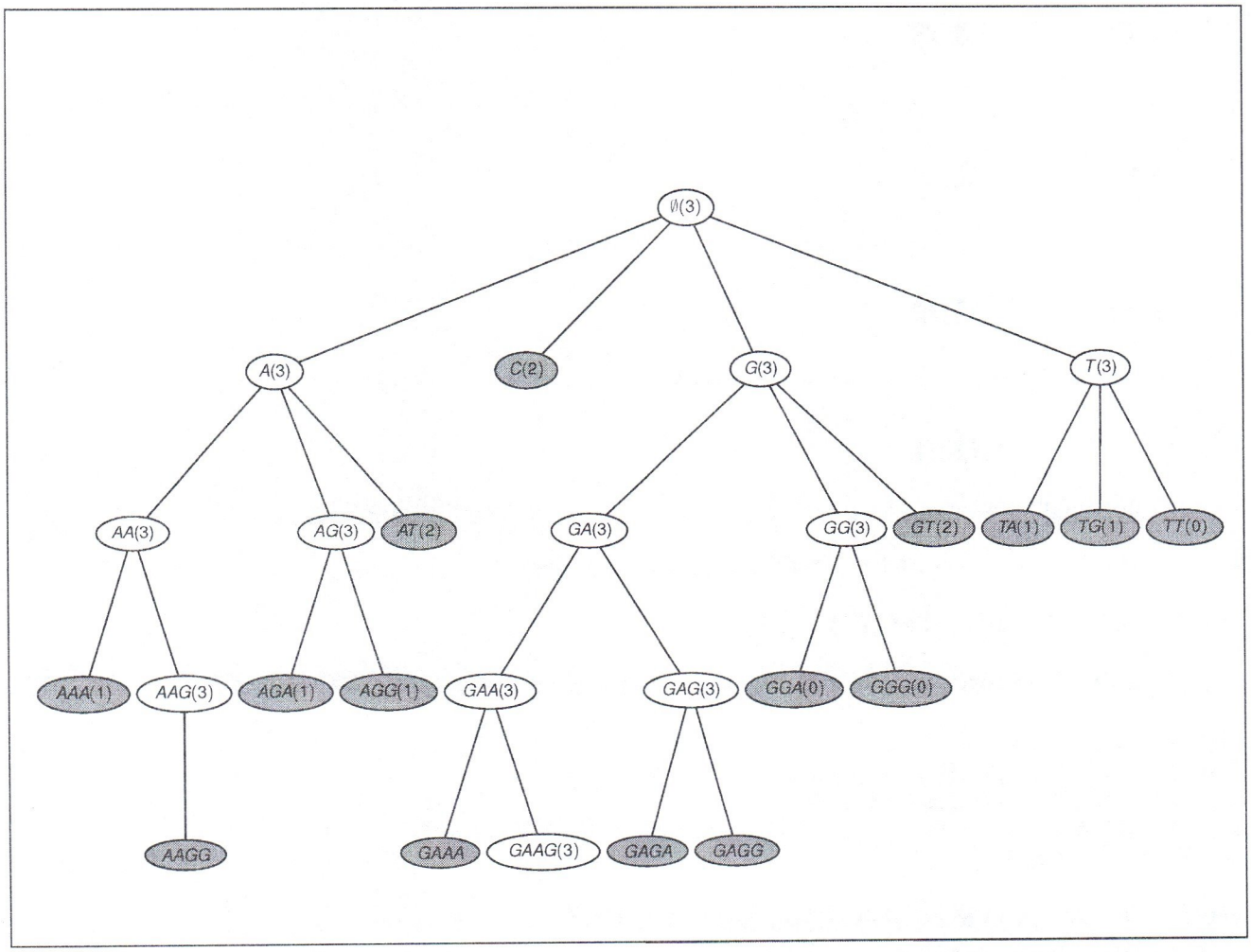
a(p0) = d(p0,p3) = 4
a(p1) = (d(p1,p2)+d(p1,p4))/2 = 3.41
a(p2) = (d(p2,p1)+d(p2,p4))/2 = 2.83
a(p3) = a(p0) = 4
a(p4) = a(p1) = 3.41

b(p0) = (d(p0,p1)+d(p0,p2)+d(p0,p4))/3 = 5.00
b(p1) = (d(p1,p0)+d(p1,p3))/2 = 5.70
b(p2) = (d(p2,p0)+d(p2,p3))/2 = 3.61
b(p3) = b(p0) = 5.00
b(p4) = b(p1) = 5.70

sc(p0) = 0.20
sc(p1) = 0.40
sc(p2) = 0.22
sc(p3) = sc(p0) = 0.20
sc(p4) = sc(p1) = 0.40

Problem 2 (6 points). Given the following sequence database and a minimum support of 3, apply GSP to extract the frequent sequences.

Id	Sequence
s_1	*CAGAAGT*
s_2	*TGACAG*
s_3	*GAAGT*



Problem 2 (continued).

Example

- Given the following sequence database

Id	Sequence
s_1	*CAGAAGT*
s_2	*TGACAG*
s_3	*GAAGT*

- With a minsup of 3, the set of frequent subsequences is
  - A(3), G(3), T(3)
  - AA(3), AG(3), GA(3), GG(3)
  - AAG(3), GAA(3), GAG(3)
  - GAAG(3)

Problem 3 (6 points). Consider the following code from a python notebook that applies three visualization techniques to a data set containing some biking sharing data.

```
import pandas as pd
import numpy as np
import seaborn as sns
sns.set(style="white", color_codes=True)
data = pd.read_csv('london.csv')
input_variables = data.columns[data.columns!='cnt']
target_variable = 'cnt'
```

```
data.head()
```

	cnt	t1	t2	hum	wind_speed	weather_code	is_holiday	is_weekend	season
0	182	3.0	2.0	93.0	6.0	3.0	0.0	1.0	3.0
1	138	3.0	2.5	93.0	5.0	1.0	0.0	1.0	3.0
2	134	2.5	2.5	96.5	0.0	1.0	0.0	1.0	3.0
3	72	2.0	2.0	100.0	0.0	1.0	0.0	1.0	3.0
4	47	2.0	0.0	93.0	6.5	1.0	0.0	1.0	3.0

```
data.describe()
```

	cnt	t1	t2	hum	wind_speed	weather_code	is_holiday	is_weekend	season
count	17414.000000	17414.000000	17414.000000	17414.000000	17414.000000	17414.000000	17414.000000	17414.000000	17414.000000
mean	1143.101642	12.468091	11.520836	72.324954	15.913063	2.722752	0.022051	0.285403	1.492075
std	1085.108068	5.571818	6.615145	14.313186	7.894570	2.341163	0.146854	0.451619	1.118911
min	0.000000	-1.500000	-6.000000	20.500000	0.000000	1.000000	0.000000	0.000000	0.000000
25%	257.000000	8.000000	6.000000	63.000000	10.000000	1.000000	0.000000	0.000000	0.000000
50%	844.000000	12.500000	12.500000	74.500000	15.000000	2.000000	0.000000	0.000000	1.000000
75%	1671.750000	16.000000	16.000000	83.000000	20.500000	3.000000	0.000000	1.000000	2.000000
max	7860.000000	34.000000	34.000000	100.000000	56.500000	26.000000	1.000000	1.000000	3.000000

```
cov=data[input_variables].corr()
```

Visualization #1

```
sns.heatmap(cov,square=True,annot=True,cmap="Blues");
```

Visualization #2

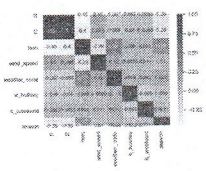
```
sns.clustermap(cov, square=True, annot=True, cmap="Blues");
```

Visualization #3

```
sns.clustermap(data[input_variables])
```

Question #1: What will the first visualization (#1) plot?

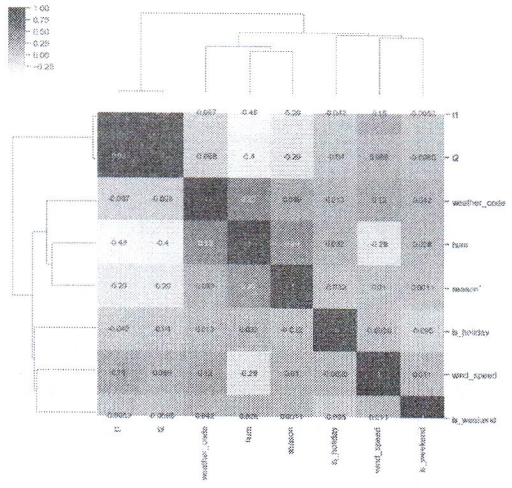
This plots the correlation matrix as a heatmap with darker blue colors representing higher correlation values.



Problem 3 (continued).

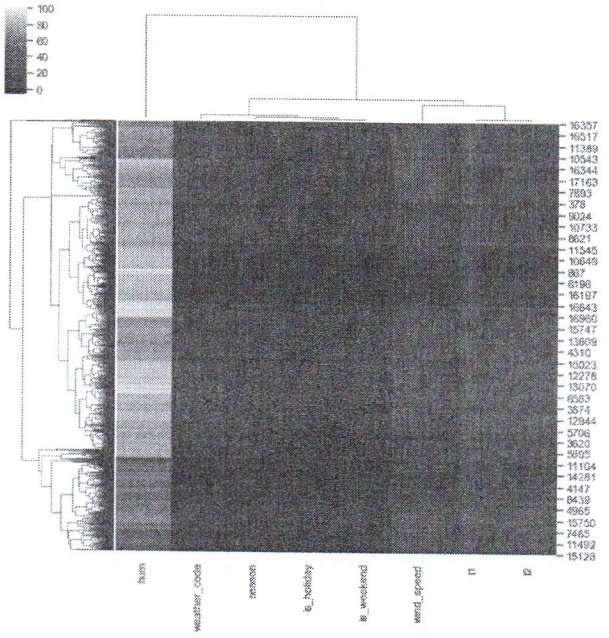
Question #2: What will the second visualization (#2) plot? How does it differ from #1?

This also performs a hierarchical clustering on the correlation matrix and also plots the dendrogram. It also sort the variables according to the dendrogram highlight clusters of variables with similar correlation values over the entire matrix row/column.



Question #3: What will the third visualization (#3) plot? What type of information does it give us?

By applying the clustermap over the entire dataset it plots the heatmap of all the data and also applies hierarchical clustering over the rows (as usual) and over the columns to search group of variables with similar behavior.



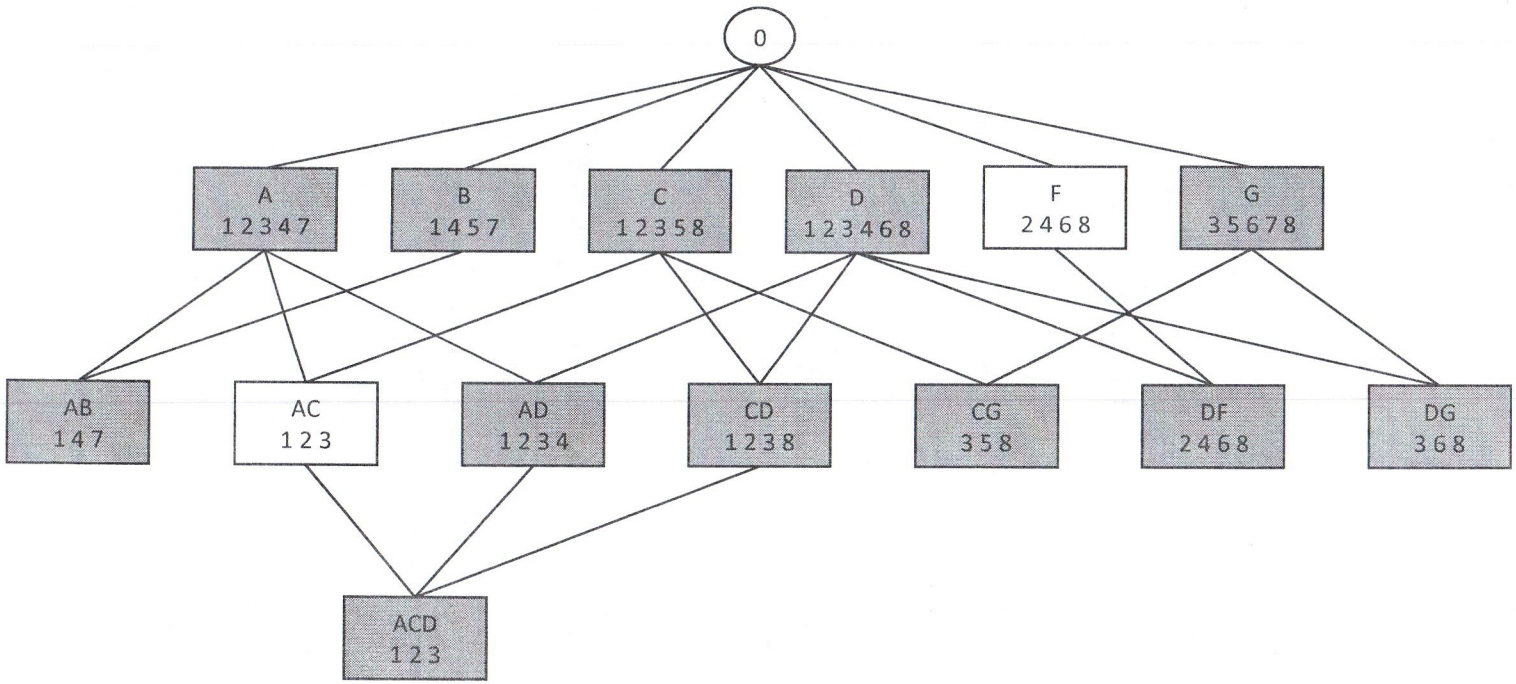
Problem 4 (6 points). Given the data set below and a min support of 3/8 (1) extract all the frequent itemsets using the Eclat algorithm and (2) list the closed and maximal frequent itemsets.

tid	itemset
t_1	*ABCD*
t_2	*ACDF*
t_3	*ACDEG*
t_4	*ABDF*
t_5	*BCG*
t_6	*DFG*
t_7	*ABG*
t_8	*CDFG*

This is problem Q2 from the text book. First we need to compute the vertical representation that is,

A	B	C	D	E	F	G
1	1	1	1	3	2	3
2	4	2	2		4	5
3	5	3	3		6	6
4	7	5	4		8	7
7		8	6			8
			8			

We compute the frequent itemsets and highlight the maximal frequent itemset in grey. The closed frequent itemsets are B, G, AD.


0
A
1 2 3 4 7
B
1 4 5 7
C
1 2 3 5 8
D
1 2 3 4 6 8
F
2 4 6 8
G
3 5 6 7 8
AB
1 4 7
AC
1 2 3
AD
1 2 3 4
CD
1 2 3 8
CG
3 5 8
DF
2 4 6 8
DG
3 6 8
ACD
1 2 3

Problem 5 (6 points). Suppose you are applying bagging trees using 40 base classifiers to a binary classification problem. Each classifier has an error rate e=0.60; assume that classifiers are independent. Compute the probability that the ensemble classifier makes a wrong prediction and comment the result.

Why does it work?

- Suppose there are 25 base classifiers
- Each classifier has error rate, $\varepsilon = 0.35$
- Assume classifiers are independent
- The probability that the ensemble makes a wrong prediction is

$$\sum_{i=13}^{25} \binom{25}{i} \epsilon^i (1-\epsilon)^{25-i} = 0.06$$

If we repeat the computation with 40 classifiers and an error of 0.6 we obtain an ensemble error of 0.87 that is much higher than the error of the single classifiers. This because the error on the binary classification problem is larger than 0.5. Thus it is not convenient to use an ensemble with these parameters unless we modify our interpretation of the classifiers' output.

Problem 6 (3 points). You are given a paper that contains 15 times the word "set", 8 times the word "computing" and 6 times the word "mining". The paper is part of a collection of 39 papers. In this collection, the word "set" appears in 20 papers, the word "computing" appears in 10 papers, while the word "mining" appears in 5 papers. Compute the Inverse Document Frequency (IDF) for the terms "set", "computing" and "mining" and say, according to IDF, which is the most important keyword among them. Comment the result.

Given the number of documents M and the number k of documents in which the word w appears, the IDF is computed as,

$$\mathbf{IDF(w) = \log_2((M+1)/k)}$$

IDF(set) = $\log_2$(40/20) = 1

IDF(computing) = $\log_2$(40/10) = 2

IDF(mining) = $\log_2$(40/5) = 3

"mining" is the most important word since it appears in fewer documents.