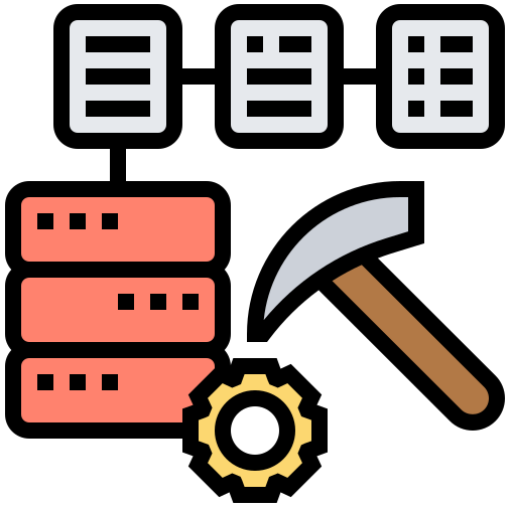

Data Mining

Homework 1

Ashkan Ansarifard

1970082

A.Y. 2023/24

Problem 1

1. I choose the probability space such that each element in Ω corresponds to a unique ordering of the cards in the deck.

   Mathematically, $\Omega = \{\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{52!}\}$, where each σ_i is a distinct permutation.

   Since each permutation in Ω is equally likely when we shuffle a standard deck of cards, we have a uniform probability distribution.

   The probability measure P assigns equal probabilities to all permutations in Ω.

   Therefore, for any event $A \subseteq \Omega$, we have:

$$P(A) = \frac{|A|}{52!}$$

   Specifically, for each individual permutation σ_i in Ω:

$$P(\sigma_i) = \frac{1}{52!}$$

2. Probability of the each event is:

   (a) **Finding the probability that the first two cards include at least one ace**

   We can use the complement rule:

$$\text{Probability (at least one ace in the first two cards)} =$$

$$1 - \text{ Probability (no aces in the first two cards)}$$

   Now, let's calculate the probability of not getting any aces in the first two cards. There are 52 cards in the deck initially, and 48 of them are not aces. When we pick the first card, there are 48 non-ace cards out of 52 possibilities. After picking the first card, there are now 51 cards left in the deck, with 47 of them being non-ace cards. Therefore, the probability of not getting an ace on the second card, given that the first card was not an ace, is:

$$\frac{48}{52} \cdot \frac{47}{51}$$

So, the probability that the first two cards contain at least one ace is:

$$1 - \left(\frac{48}{52} \cdot \frac{47}{51}\right) \approx 0.1494$$

(b) **Finding the probability that the first five cards include at least one ace**

Same as item (a) we can use the complement rule:

$$\text{Probability (at least one ace in the first five cards)} =$$

$$1 - \text{ Probability (no aces in the first five cards)}$$

Same as previous item, the probability of not getting an ace on the first 5 cards, is:

$$\frac{48}{52} \cdot \frac{47}{51} \cdot \frac{46}{50} \cdot \frac{45}{49} \cdot \frac{44}{48}$$

So, the probability that the first five cards include at least one ace:

$$1 - \left(\frac{48}{52} \cdot \frac{47}{51} \cdot \frac{46}{50} \cdot \frac{45}{49} \cdot \frac{44}{48}\right) \approx 0.3411$$

(c) **The first two cards are a pair of the same rank**

To find the probability that the first two cards drawn from a standard deck are a pair of the same rank, you can approach it as follows:

- There are 52 cards in a standard deck.
- For the first card, there are no restrictions, so there are 52 possibilities.
- For the second card, you want it to be of the same rank as the first card, which means there are 3 cards of the same rank left in the deck (since there are 4 cards of each rank in a standard deck, and you've already drawn one).

So, the probability of drawing a pair of the same rank for the first two cards is:

$$\text{Probability} = \frac{52/52 \cdot 3/51}{1} = \frac{1}{17}$$

So, the probability of drawing the first two cards as a pair of the same rank is $1/17 \approx 0.0588$.

(d) **The first five cards are all diamonds.**

There are 13 diamonds in a standard deck.

The probability of drawing a diamond on the first draw is $\frac{13}{52}$ because there are 13 diamonds out of 52 cards.

Similarly, for the second, third, fourth, and fifth draws, the probabilities are as follows:

Second draw: $\frac{12}{51}$

Third draw: $\frac{11}{50}$

Fourth draw: $\frac{10}{49}$

Fifth draw: $\frac{9}{48}$

Because these are independent events, we can multiply these probabilities together:

$$\frac{13}{52} \cdot \frac{12}{51} \cdot \frac{11}{50} \cdot \frac{10}{49} \approx 0.00018184$$

So, the probability of drawing the first five cards as diamonds is approximately 0.00018184, or about 0.0182%.

(e) **The first five cards form a full house**

There are $\binom{4}{3} = 4$ different ways to choose 3 cards of the same type and $\binom{4}{2} = 6$ different ways to choose 2 cards of the same type. To get a full house, you need to first pick the type of the 3 of a kind, which is $\binom{13}{1} = 13$ different choices, and choose the type of the pair, which is $\binom{12}{1} = 12$ different choices. The order does not count, so there are $\binom{13}{1} \cdot \binom{4}{3} \cdot \binom{12}{1} \cdot \binom{4}{2}$ ways to have a full house.

$$\binom{13}{1} \cdot \binom{4}{3} \cdot \binom{12}{1} \cdot \binom{4}{2} = 3744$$

Note that there are $\binom{52}{5} = 2,598,960$ different combinations for the first 5 cards, so the probability of being dealt a full house is:

$$\frac{3744}{2,598,960} \approx 0.14\%$$

3. Simulation results[1]:

Event	Analytical Solution	Simulation (10,000 iterations)	Simulation (100,000 iterations)
a	0.1494	0.1529	0.1492
b	0.3411	0.3411	0.3397
c	0.0588	0.0591	0.0585
d	0.0001	0.0006	0.0005
r	0.0014	0.0020	0.0014

[1]Results are derived from CardSimulator in repo/problem1/CardSimulator.py

Problem 2

1. For calculating the probability that the baby born at midnight was a boy I design the following probability space: let's define E_1 and E_2 as following:

   b: Number of boys before midnight (In our case 4)

   g: Number of girls before midnight

   E_1: boy is born at midnight

   E_2: girl is born at midnight

   Let say that initially there were n girls.

   B: boy picked up by nurse

   G: girl picked up by nurse

2. We have to calculate $P(E_1|B)$:

$$P(E_1|B) = \frac{P(E_1 \cap B)}{P(B)}$$

   Assumption: I assumed, when a child is born, the probability of that child being either a boy or a girl is 50%

   The chance of picking a boy, which we'll call P(B), is calculated by adding together the probability of picking a boy such that the boy was born at midnight and the probability of picking a boy such that the girl was born at midnight.

$$P(B) = \frac{1}{2} \cdot P(B|E_1) + \frac{1}{2} \cdot P(B|E_2)$$

$$P(B) = \frac{1}{2} \cdot \frac{b+1}{b+1+g} + \frac{1}{2} \cdot \frac{b}{b+1+g} = \frac{1}{2} \cdot \frac{2b+1}{b+1+g}$$

   As $P(E_1 \cap B)$ = $P(B \cap E_1)$, and $P(B \cap E_1)$ = $P(B|E_1) \cdot P(E_1) \ldots$

$$P(E_1|B) = \frac{P(B|E_1) \cdot P(E_1)}{P(B)}$$

   So,

$$P(E_1|B) = \frac{\frac{1}{2} \cdot \frac{b+1}{b+1+g}}{\frac{1}{2} \cdot \frac{2b+1}{b+1+g}} = \frac{b+1}{2b+1} = \frac{5}{9} \approx 0.56$$

Problem 3

Let's define the following probability space for this problem:

n: Number of times your roll three dices

E_{11} : Sum is 11

E_{16} : Sum is 16

E_{11}^C : Sum is not 11

E_{16}^C : Sum is not 16

Let's simulate some numbers of rolling times and calculate the possible outcomes to gain an insight:

$$n = 1 \rightarrow \begin{cases} E_{11}^1 & P(E_{11}) = \frac{18}{216} \\ E_{16}^1 & P(E_{16}) = \frac{6}{216} \\ (E_{11}^1 \textit{ and } E_{16}^1)^C & P(E_{11}^C \cup E_{16}^C) = \frac{195}{216} \end{cases}$$

$$n = 2 \xrightarrow[\text{in n = 1}]{\text{Sum was not 11 not 16}} \begin{cases} E_{11}^2 & P(E_{11}) = \frac{195}{216} \cdot \frac{18}{216} \\ E_{16}^2 & P(E_{16}) = \frac{195}{216} \cdot \frac{6}{216} \\ (E_{11}^2 \textit{ and } E_{16}^2)^C & P(E_{11}^C \cup E_{16}^C) = \frac{195}{216} \cdot \frac{195}{216} \end{cases}$$

$$n = 3 \xrightarrow[\text{not n = 1 nor in n = 2}]{\text{Sum was not 11 not 16}} \begin{cases} E_{11}^3 & P(E_{11}) = \frac{195}{216} \cdot \frac{195}{216} \cdot \frac{18}{216} \\ E_{16}^3 & P(E_{16}) = \frac{195}{216} \cdot \frac{195}{216} \cdot \frac{6}{216} \\ (E_{11}^3 \textit{ and } E_{16}^3)^C & P(E_{11}^C \cup E_{16}^C) = \frac{195}{216} \cdot \frac{195}{216} \cdot \frac{195}{216} \end{cases}$$

So, we found a trend!

The probability that you stop because you see a sum of 16 in the n^{th} time is:

$$P(E_{16})^n = P(E_{11}^C \cup E_{16}^C)^{n-1} \cdot P(E_{16})$$

Which will be:

$$P(E_{16})^n = (\frac{195}{216})^{n-1} \cdot \frac{6}{216}$$

Problem 4

To find the expectation of X, we need to calculate the average number of times the word "mining" appears in a randomly typed sequence of $100,000,000,000$ letters.

The word "mining" consists of 6 letters, and each letter has an equal probability of being typed by the monkey. So, the probability of the monkey typing the word "mining" in a specific order is $\left(\frac{1}{26}\right)^6$.

Now, let's calculate the number of possible positions where the word "mining" can start in a sequence of $100,000,000,000$ letters. Since there are $100,000,000,000 - 6 + 1 = 99,999,999,995$ possible starting positions, the probability of the monkey typing the word "mining" in any of these positions is $99,999,999,995 \left(\frac{1}{26}\right)^6$.

To find the expectation of X, we multiply the probability of the monkey typing the word "mining" in any given position by the total number of possible starting positions:

$$\text{Expectation of } X = 99,999,999,995 \left(\frac{1}{26}\right)^6$$

Let's calculate this value:

$$\text{Expectation of } X = 99,999,999,995 \left(\frac{1}{26}\right)^6$$

$$\text{Expectation of } X \approx 99,999,999,995 \times 3.2371e^{-9}$$

$$\text{Expectation of } X \approx 323.71$$

Therefore, the expectation of X, the number of times the word "mining" appears, is approximately 323.71.

Problem 5

To find the probability of seeing a bicycle in a given time interval, as said in the problem, we can assume that the probability of seeing a bicycle remains constant over time.

We already know the probability of seeing a bicycle in 45 minutes as $P(45)$, which is 97%. We want to find the probability of seeing a bicycle in a 15-minute interval, written as $P(15)$.

Since the probability remains constant, we can assume that the ratio of probabilities is the same as the ratio of time intervals:

$$\frac{P(15)}{P(45)} = \frac{15}{45}$$

To find $P(15)$, we can rearrange the equation:

$$P(15) = \frac{15}{45} \times P(45)$$

Substituting the given value of $P(45)$ as 97% or 0.97:

$$P(15) = \frac{15}{45} \times 0.97$$

Calculating this expression:

$$P(15) = 0.32$$

Therefore, the probability of seeing a bicycle in a 15-minute interval is 32%.

Problem 6

1. Erdős–Rényi random graph is denoted by $\mathcal{G}(n,p) = (\Omega, \mathcal{F}, P)$ where Ω is the sample space of all possible graphs on n vertices with the size: $|\Omega| = 2^N = 2^{\binom{n}{2}}$, $\mathcal{F}$ are the events and P would be the probability measure that for each graph
2. If we fix $p \in [0,1]$ and choose edges according to Bernoulli's trials, then P would be: $G \in \Omega$ assigns probability: $P(G) = p^m \cdot (1-p)^{N-m}$ where m is the number of edges in G
3. To calculate the probability of exactly one triangle, we need to consider:
   - Choosing the Three Nodes for the Triangle: Choosing three nodes out of n: $\binom{n}{3}$
   - Ensuring the Presence of Edges within the Triangle: the probability of these edges existing within the triangle is p^3 because these events are independent.
   - Ensuring No Other Edges: To ensure no other edges are present in the graph, all the remaining pairs of nodes should not have edges between them. The probability of this is $(1-p)^{(\binom{n}{3}-3)}$

   So, overall: $P(\text{Exactly one triangle}) = \binom{n}{3} \cdot (p^3) \cdot (1-p)^{\frac{n(n-1)}{2}-3}$
4. Similar to the previous event, here we need to consider the following:
   - There are $(n-1)$ edges in the path, and all of them must exist with probability p.
   - Since these edges are independent, we can raise the probability p to the power of $(n-1)$ to account for the presence of all these edges.
   - The remaining edges, that could form triangles or other shape in the graph should not exist. So, we also need to account for the absence of these additional edges. The probability of edges not in the path not existing is $(1-p)$, and there are $\binom{n}{2} - (n-1)$ such edges.

   So, $P(\text{All nodes connected in a line}) = (p^{n-1}) \cdot ((1-p)^{\binom{n}{2}-(n-1)})$
5. The expected number of edges in a random graph created according to the $\mathcal{G}(n,p)$ model can be found using the concept of linearity of expectation. for each pair of nodes (i,j), we can calculate the expected number of edges contributed by that pair:
   - If the edge (i,j) exists, it contributes 1 edge with probability p.
   - If the edge (i,j) does not exist, it contributes 0 edges with probability $(1-p)$.

Expected number of edges = Sum of expectations for all pairs of nodes

$$\begin{aligned}\text{Expected number of edges} = p \cdot (\text{number of ways to choose 2 nodes}) \cdot 1 \\ + (1-p) \cdot (\text{number of ways to choose 2 nodes}) * 0\end{aligned}$$

So, the expected number of edges $= p \cdot \binom{n}{2} = p \cdot \frac{n(n-1)}{2}$

6. To find the expected number of 3-stars, we consider all possible combinations of central nodes and peripheral nodes in the graph:

   - There are n ways to choose the central node (any of the n nodes).
   - There are $\binom{n-1}{3}$ ways to choose three peripheral nodes from the remaining n-1 nodes.
   - For each 3-star subgraph, there are p^3 chances that edges exist between the central node and the three peripheral nodes.
   - There is $(1-p)^{3(n-4)}$ probability that no edges exist between the peripheral nodes.

   So: To find the expected number of 3-stars, we sum the probabilities of each possible 3-star configuration:

$$\text{Expected number of 3-stars} = n \cdot \binom{n-1}{3} \cdot p^3 \cdot (1-p)^{3(n-4)}$$

7. Like item 6, in a graph made using the G(n,p) model, a k-star is a subgraph where one central node is connected to k peripheral nodes, and no edges exist between the peripheral nodes.

   To find the expected number of k-stars, we consider all possible ways to choose the central node and the k peripheral nodes:

$$\text{Expected number of k-stars} = n * \binom{n-1}{k} * p^k * (1-p)^{k(n-k)}$$

Problem 7

For this problem a console program is implemented based on retrieving data using two different libraries:

1. *requests* Python library → DirectAPIClient.py
2. *meteomatics.api* library → HighLevelAPIClient.py

Initially, the user chooses based on which library he wants to retrieve the data.

The **HighLevelAPIClient** class, uses meteomatics.api library which is a connector created by meteomatics to simplify the integration of the data into existing workflows. It is more easy to use and is supported in several programming languages.

The **DirectAPIClient** class uses requests Python library in which we construct the API URL ourself.

Both of these classes can be used in this program but because using requests Python library was requested in the problem 7, I describe more in detail what I'm doing in DirectAPIClient class to retrieve the data.

Retrieving Data Using DirectAPIClient Class

The `DirectAPIClient` class retrieves weather data from the Meteomatics API through the following steps:

1. It formats the time range for the query using the provided start date, end date, and interval.
2. The API URL is constructed by combining the base URL, time range, selected parameters, coordinates (latitude and longitude), and specifying the output format as JSON.
3. An HTTP GET request is sent to the constructed API URL, including your Meteomatics API username and password for authentication.
4. The response from the API is captured and checked for its status code.
5. In the case of a successful response (status code 200), the function parses the JSON response and extracts the weather data.
6. The weather data is processed to obtain time series for each selected parameter.
7. A Plotly figure is created, with traces (lines) added for each parameter to visualize trends over time.

8. The function includes error handling for errors based on the given HTTP status codes and provides informative error messages.
9. Also exception handling is implemented to catch any unexpected exceptions during the process.
10. The Plotly figure is displayed, allowing users to analyze and visualize the weather data.

How to Use The Program

Run Python Main.py

Follow the prompts to select parameters, input dates and time intervals, and choose either the high-level or direct API client.

The program will then query the Meteomatics API, retrieve weather data, and display it as a Plotly graph.

You can visualize and analyze the trends of the selected weather parameters over a any time period that you want.

Example

In this example, Precipitation, Wind speed, Pressure, Weather Symbol and UV trend over 24 hours is plotted.

```
"E:\EDU\Data Mining\dm-projects\hw1\repo\venv\Scripts\python.exe" "E:\EDU\Data Mining\dm-projects\hw1\repo\Main.py"
Enter 'list' to see available parameters or press Enter to continue:
Enter 'meteomatics' to use the high-level API client with the meteomatics library or enter 'requests' to use the requests module: meteomatics
Enter your Meteomatics API username: sapienza_ansarifard_ashkan
Enter your Meteomatics API password: WPDq43z3Mv
Enter latitude: 41.902782
Enter longitude: 12.496366
Enter weather parameters (comma-separated): "t_2m:C", "precip_1h:mm", "wind_speed_10m:ms","msl_pressure:hPa", "weather_symbol_1h:idx", "uv:idx"
Enter start date and time (e.g., 2023-10-05 00:00:00): 2023-10-12
Enter end date and time (e.g., 2023-10-16 00:00:00): 2023-10-13
Enter data interval in hours: 1

Process finished with exit code 0
```

Figure 1: Console Output of the Example

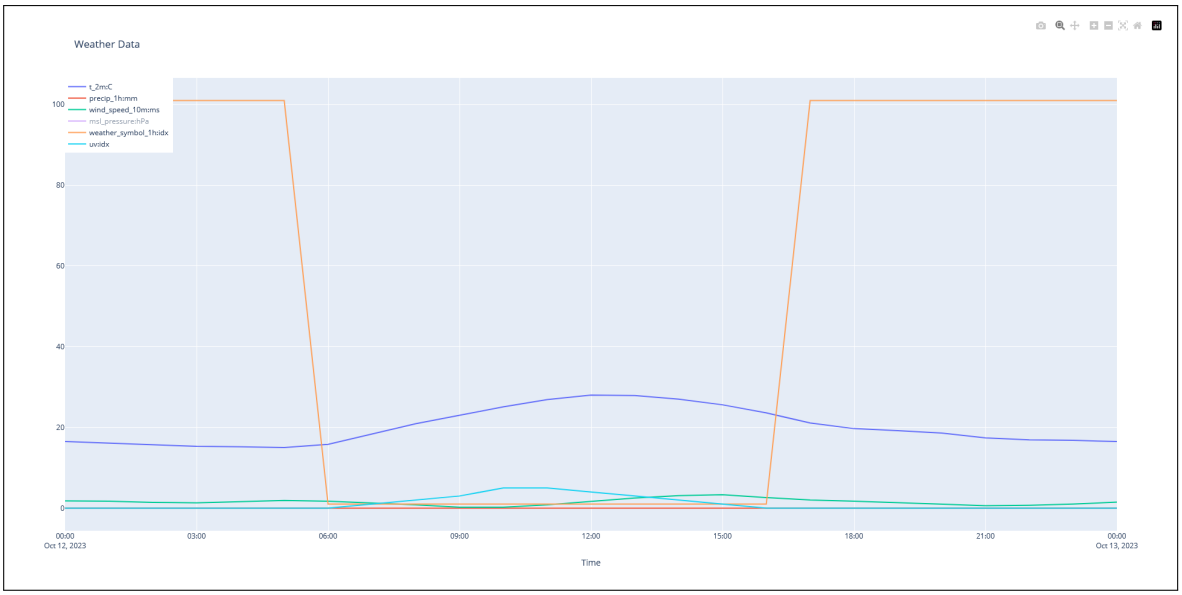

Figure 2: Plot of Precipitation, Wind speed, Pressure, Weather Symbol and UV trend over 24 hours