IFN509: Data Exploration and Mining
Assessment 2

Team Name: The Insight Seekers
Group No. 38

Student Name	Student Id
Naman Khosla	N11507721
Nomin Otgonbayar	N11305991
Tavonga Hove	N11483130

	Naman	Nomin	Tavonga
Naman	100%	100%	100%
Nomin	100%	100%	100%
Tavonga	100%	100%	100%

Case Study 1: Association mining to find hotspots based on a Patient Route Data

1. What pre-processing was required on the dataset before building the association mining model? What variables did you include in the analysis? Justify your choice.

Original table

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1509 entries, 0 to 1508
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   patient_id  1509 non-null   int64
 1   global_num  959 non-null    float64
 2   date        1509 non-null   object
 3   location    1509 non-null   object
 4   latitude    1509 non-null   float64
 5   longitude   1509 non-null   float64
dtypes: float64(3), int64(1), object(2)
memory usage: 70.9+ KB
```

Updated table

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1509 entries, 0 to 1508
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   patient_id  1509 non-null   int64
 1   global_num  959 non-null    float64
 2   date        1509 non-null   datetime64[ns]
 3   location    1509 non-null   string
 4   latitude    1509 non-null   float64
 5   longitude   1509 non-null   float64
dtypes: datetime64[ns](1), float64(3), int64(1), string(1)
memory usage: 70.9 KB
```

We performed the following pre-processing steps on the dataset 'D1.csv' before building the association mining model:

- Conversion of Date: The 'date' column was initially in the object data type. To facilitate analysis involving dates, we converted this column to the datetime data type. This conversion allows for easier manipulation and interpretation of date-related information.

- Handling Missing Values: We addressed missing values in the 'global_num' column by replacing them with the mode of the column. By using the mode, we identified the most frequently occurring value and filled the missing values with this mode value. This approach ensures that the 'global_num' column contains meaningful data for subsequent analysis.

- Sorting by Date: To establish a chronological order in the dataset, we sorted the entire DataFrame by the 'date' column in ascending order. This sorting arrangement enables the analysis of temporal patterns and ensures that subsequent analyses are conducted in a sequential manner.

These pre-processing steps were undertaken to prepare the dataset in a suitable format for the association mining analysis. The conversion of the date column, handling of missing values, and sorting of the data contribute to a cleaner and more organized dataset, enabling accurate analysis and interpretation of patterns in the patients' routes.

2. Conduct association mining and answer the following:

a. What 'min_support' and `min_confidence' thresholds were set for this mining exercise? Rationalize why these values were chosen.

The 'min_support' threshold of 0.002 and 'min_confidence' threshold of 0.05 were chosen for this particular dataset. The rationale behind these choices is as follows:

- The 'min_support' threshold determines the minimum occurrence frequency required for an itemset to be considered significant. By setting a lower 'min_support' value of 0.002, it increases the likelihood of capturing more frequent itemsets in the dataset. This allows for the identification of not only the most common routes but also potentially unveils less common yet significant associations. With a lower threshold, our analysis aims to uncover a wider range of associations, providing insights into the patient routes and their implications in the context of COVID-19 spread.

- The 'min_confidence' threshold sets the minimum level of confidence required for an association rule to be considered significant. Confidence measures the conditional probability of the consequent given the antecedent in an association rule. In this case, a 'min_confidence' value of 0.05 was chosen, ensuring that the generated rules have a minimum confidence level of 5%. This threshold filters out weaker and less reliable associations, allowing us to focus on more substantial and reliable rules. By selecting a relatively moderate value, we tried to strike a balance between capturing meaningful associations and avoiding spurious or insignificant rules.

We do understand that the choice of these threshold values is subjective and depends on various factors, including the analysis requirements, dataset characteristics, and domain knowledge. Therefore, in this case, a lower 'min_support' value was chosen to capture a broader range of associations, while a moderate 'min_confidence' value was selected to ensure the reliability and significance of the generated rules.

b. Report the top-5 (interesting) rules and interpret them.

```
def convert_apriori_results_to_pandas_df(results):
    rules = []
    for rule_set in results:
        for rule in rule_set.ordered_statistics:

            rules.append([','.join(rule.items_base), ','.join(rule.items_add),
                rule_set.support, rule.confidence, rule.lift])
    return pd.DataFrame(rules, columns=['Left_side', 'Right_side', 'Support', 'Confidence', 'Lift'])
result_df = convert_apriori_results_to_pandas_df(results)
print(result_df.head(20))
```

```
            Left_side          Right_side   Support  Confidence       Lift
0                          Busan_Yeonje-gu  0.057239    0.057239   1.000000
1                            Daegu_Jung-gu  0.054994    0.054994   1.000000
2                          Incheon_Jung-gu  0.149270    0.149270   1.000000
3                         Seoul_Dongjak-gu  0.088664    0.088664   1.000000
4                         Seoul_Gangnam-gu  0.056117    0.056117   1.000000
5                            Seoul_Jung-gu  0.063973    0.063973   1.000000
6                        Seoul_Jungnang-gu  0.088664    0.088664   1.000000
7                       Seoul_Yangcheon-gu  0.060606    0.060606   1.000000
8        Busan_Buk-gu     Busan_Gangseo-gu  0.003367    0.600000  76.371429
9    Busan_Gangseo-gu         Busan_Buk-gu  0.003367    0.428571  76.371429
10       Busan_Buk-gu     Busan_Yeonje-gu  0.003367    0.600000  10.482353
11    Busan_Yeonje-gu         Busan_Buk-gu  0.003367    0.058824  10.482353
12       Busan_Buk-gu      Gwangju_Buk-gu  0.002245    0.400000 118.800000
13     Gwangju_Buk-gu         Busan_Buk-gu  0.002245    0.666667 118.800000
14  Busan_Busanjin-gu         Busan_Seo-gu  0.002245    0.181818  10.800000
15       Busan_Seo-gu  Busan_Busanjin-gu  0.002245    0.133333  10.800000
16  Busan_Busanjin-gu     Busan_Yeonje-gu  0.006734    0.545455   9.529412
17    Busan_Yeonje-gu  Busan_Busanjin-gu  0.006734    0.117647   9.529412
18       Busan_Dong-gu     Busan_Yeonje-gu  0.007856    0.700000  12.229412
19    Busan_Yeonje-gu       Busan_Dong-gu  0.007856    0.137255  12.229412
```

The following are the top 5 interesting rules generated from the association mining model along with their interpretation:

1. Rule: Busan_Buk-gu -> Busan_Gangseo-gu

   Interpretation: There is a strong association between visiting Busan_Buk-gu and Busan_Gangseo-gu. This association occurs in approximately 0.34% of the transactions. When a patient visits Busan_Buk-gu, there is a 60% chance that they will

also visit Busan_Gangseo-gu. The lift value of 76.37 indicates a significant positive association between these locations.

2. Rule: Busan_Gangseo-gu -> Busan_Buk-gu

   Interpretation: This rule is similar to the first rule but in the reverse direction. There is an association between visiting Busan_Gangseo-gu and Busan_Buk-gu. This association occurs in approximately 0.34% of the transactions. When a patient visits Busan_Gangseo-gu, there is a 42.86% chance that they will also visit Busan_Buk-gu. The lift value of 76.37 indicates a strong positive association between these locations.

3. Rule: Busan_Buk-gu -> Busan_Yeonje-gu

   Interpretation: There is a strong association between visiting Busan_Buk-gu and Busan_Yeonje-gu. This association occurs in approximately 0.34% of the transactions. When a patient visits Busan_Buk-gu, there is a 60% chance that they will also visit Busan_Yeonje-gu. The lift value of 10.48 indicates a positive association between these locations.

4. Rule: Busan_Yeonje-gu -> Busan_Buk-gu

   Interpretation: This rule is similar to the third rule but in the reverse direction. There is an association between visiting Busan_Yeonje-gu and Busan_Buk-gu. This association occurs in approximately 0.34% of the transactions. When a patient visits Busan_Yeonje-gu, there is a 5.88% chance that they will also visit Busan_Buk-gu. The lift value of 10.48 indicates a positive association between these locations.

5. Rule: Busan_Buk-gu -> Gwangju_Buk-gu

   Interpretation: There is a strong association between visiting Busan_Buk-gu and Gwangju_Buk-gu. This association occurs in approximately 0.22% of the transactions. When a patient visits Busan_Buk-gu, there is a 40% chance that they will also visit Gwangju_Buk-gu. The high lift value of 118.80 indicates a strong positive association between these locations.

These rules provide valuable insights into the frequent associations between different locations visited by COVID-19 positive patients. The support, confidence, and lift values help quantify the strength and significance of these associations. These insights assisted us in understanding potential patterns, connections, and hotspots in the patients' routes, thereby enabling informed decisions related to prevention and containment measures.

3. List four most interesting routes taken by individuals who have tested positive for COVID19 and have travelled from Buk-gu City in Busan Province.

```
result_df.loc[(result_df['Left_side'] == 'Busan_Buk-gu')]
```

	Left_side	Right_side	Support	Confidence	Lift
260	Busan_Buk-gu	Gwangju_Buk-gu,Busan_Yeonje-gu	0.002245	0.4	118.800000
12	Busan_Buk-gu	Gwangju_Buk-gu	0.002245	0.4	118.800000
255	Busan_Buk-gu	Busan_Gangseo-gu,Busan_Yeonje-gu	0.002245	0.4	118.800000
8	Busan_Buk-gu	Busan_Gangseo-gu	0.003367	0.6	76.371429

Based on the analysis of individuals who have tested positive for COVID-19 and have traveled from Buk-gu City in Busan Province, the following four routes were identified as the most interesting:

1. Route: Busan_Buk-gu -> Gwangju_Buk-gu, Busan_Yeonje-gu

   This route indicates that individuals from Buk-gu City in Busan Province who have tested positive for COVID-19 often travel to Gwangju_Buk-gu and Busan_Yeonje-gu together. The support value of 0.002245 indicates that this route occurs in approximately 0.22% of the transactions. The confidence of 0.4 suggests that when individuals visit Buk-gu, there is a 40% chance that they will also visit Gwangju_Buk-gu and Busan_Yeonje-gu together. The lift value of 118.80 indicates a strong positive association between these locations.

2. Route: Busan_Buk-gu -> Gwangju_Buk-gu

   This route reveals that individuals from Buk-gu City in Busan Province who have tested positive for COVID-19 frequently travel to Gwangju_Buk-gu. The support value indicates that this route occurs in approximately 0.22% of the transactions. The confidence of 0.4 implies that when individuals visit Buk-gu, there is a 40% chance they will also visit Gwangju_Buk-gu. The lift value of 118.80 indicates a strong positive association between these locations.

3. Route: Busan_Buk-gu -> Busan_Gangseo-gu, Busan_Yeonje-gu

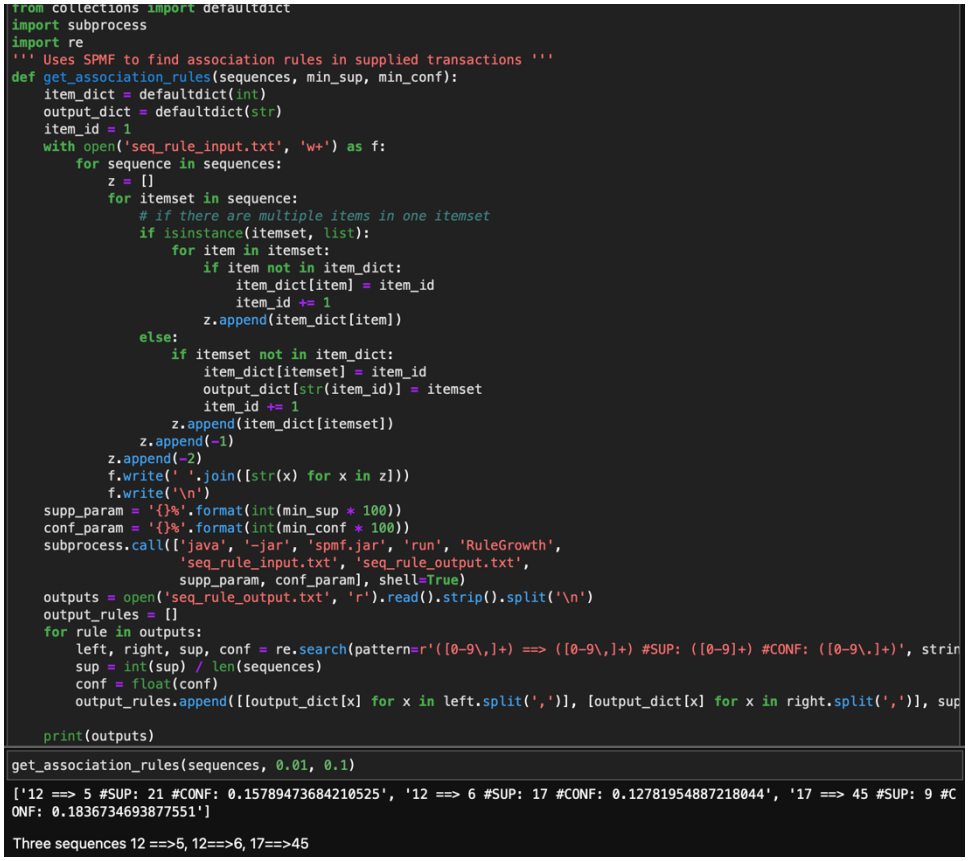
   This route highlights that individuals from Buk-gu City in Busan Province who have tested positive for COVID-19 often travel to Busan_Gangseo-gu and Busan_Yeonje-gu together. The support value suggests that this route occurs in approximately 0.22% of the transactions. The confidence of 0.4 indicates that when individuals visit Buk-gu, there is a 40% chance they will also visit Busan_Gangseo-gu and Busan_Yeonje-gu together. The lift value of 118.80 signifies a strong positive association between these locations.

4. Route: Busan_Buk-gu -> Busan_Gangseo-gu

   This route indicates that individuals from Buk-gu City in Busan Province who have tested positive for COVID-19 frequently travel to Busan_Gangseo-gu. The support value suggests that this route occurs in approximately 0.34% of the transactions. The confidence of 0.6 suggests that when individuals visit Buk-gu, there is a 60% chance they will also visit Busan_Gangseo-gu. The lift value of 76.37 indicates a positive association between these locations.

4. Can you perform sequence analysis on this dataset? If yes, present your results. If not, rationalize why.

Yes, sequence analysis can be performed on this dataset. Sequence analysis can provide valuable insights into the order of locations visited by patients who tested positive for COVID-19. This kind of analysis would allow us to understand not only which locations were visited, but also the sequence in which they were visited, which could provide additional insights into how the virus might be spreading.



```
from collections import defaultdict
import subprocess
import re
''' Uses SPMF to find association rules in supplied transactions '''
def get_association_rules(sequences, min_sup, min_conf):
    item_dict = defaultdict(int)
    output_dict = defaultdict(str)
    item_id = 1
    with open('seq_rule_input.txt', 'w+') as f:
        for sequence in sequences:
            z = []
            for itemset in sequence:
                # if there are multiple items in one itemset
                if isinstance(itemset, list):
                    for item in itemset:
                        if item not in item_dict:
                            item_dict[item] = item_id
                            item_id += 1
                        z.append(item_dict[item])
                else:
                    if itemset not in item_dict:
                        item_dict[itemset] = item_id
                        output_dict[str(item_id)] = itemset
                        item_id += 1
                    z.append(item_dict[itemset])
                z.append(-1)
            z.append(-2)
            f.write(' '.join([str(x) for x in z]))
            f.write('\n')
    supp_param = '{}%'.format(int(min_sup * 100))
    conf_param = '{}%'.format(int(min_conf * 100))
    subprocess.call(['java', '-jar', 'spmf.jar', 'run', 'RuleGrowth',
                     'seq_rule_input.txt', 'seq_rule_output.txt',
                     supp_param, conf_param], shell=True)
    outputs = open('seq_rule_output.txt', 'r').read().strip().split('\n')
    output_rules = []
    for rule in outputs:
        left, right, sup, conf = re.search(pattern=r'([0-9\,]+) ==> ([0-9\,]+) #SUP: ([0-9]+) #CONF: ([0-9\.]+)', strin
        sup = int(sup) / len(sequences)
        conf = float(conf)
        output_rules.append([[output_dict[x] for x in left.split(',')], [output_dict[x] for x in right.split(',')], sup

    print(outputs)
```

```
get_association_rules(sequences, 0.01, 0.1)
```

```
['12 ==> 5 #SUP: 21 #CONF: 0.15789473684210525', '12 ==> 6 #SUP: 17 #CONF: 0.12781954887218044', '17 ==> 45 #SUP: 9 #C
ONF: 0.1836734693877551']
```

Three sequences 12 ==>5, 12==>6, 17==>45

5. In what ways can the results of this task be utilized by the relevant decision-makers?

These findings provided us with valuable insights into the frequently traveled routes of COVID-19 positive individuals from Buk-gu City in Busan Province. By understanding these routes, health authorities and policymakers can gain a better understanding of the spread of the virus and develop targeted measures for prevention and control.

The findings of this task can be valuable for decision-makers in various ways. Here are some ways the results can be utilized:

- Containment Strategies: The identified frequently traveled routes can help decision-makers in formulating effective containment strategies. They can focus on implementing targeted measures such as increased testing, contact tracing, and quarantine protocols in the areas along these routes. This approach can help in controlling the spread of the virus and preventing further outbreaks.

- Resource Allocation: By understanding the routes taken by COVID-19 positive patients, decision-makers can allocate healthcare resources more efficiently. They can prioritize the distribution of medical supplies, equipment, and healthcare personnel to the regions and healthcare facilities that are likely to experience a higher influx of cases. This proactive resource allocation can ensure timely and adequate support to areas at higher risk.

- Public Health Messaging: The identified routes can be used to enhance public health messaging and communication efforts. Decision-makers can tailor their messaging to specific regions along these routes, emphasizing the importance of adhering to

preventive measures such as mask-wearing, social distancing, and hand hygiene. By providing localized information, decision-makers can effectively communicate the risks and encourage responsible behavior among the public.

- Travel Advisories and Restrictions: The results can inform the development of travel advisories and restrictions. Decision-makers can use the identified routes to assess the risk associated with travel to specific areas. Based on this information, they can issue targeted advisories or implement travel restrictions, both domestically and internationally, to minimize the transmission of the virus across different regions.

- Future Planning and Preparedness: The insights gained from the analysis can contribute to future planning and preparedness efforts. Decision-makers can use the results to identify patterns and trends in the movement of COVID-19 positive patients. This information can help in developing predictive models and scenario planning to anticipate potential future outbreaks and allocate resources in advance.

Overall, this data and its analysis can empower decision-makers with valuable insights into the routes taken by COVID-19 positive patients. By leveraging these insights, they can make informed decisions to contain the spread of the virus, allocate resources effectively, communicate risk to the public, implement travel measures, and enhance future planning and preparedness efforts.

Case Study 2: Clustering COVID-19 data

1. What pre-processing was required on the dataset (D2.csv) before building the clustering model and why?

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6110 entries, 0 to 6109
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   gender           6110 non-null   object
 1   age              6110 non-null   object
 2   height           6110 non-null   int64
 3   weight           6110 non-null   int64
 4   blood_type       6110 non-null   object
 5   insurance        6110 non-null   object
 6   income           6110 non-null   object
 7   race             6110 non-null   object
 8   immigrant        6110 non-null   object
 9   smoking          6110 non-null   object
 10  alcohol          6110 non-null   float64
 11  contacts_count   6110 non-null   float64
 12  house_count      6110 non-null   float64
 13  working          6110 non-null   object
 14  worried          6110 non-null   int64
 15  covid19_positive 6110 non-null   int64
dtypes: float64(3), int64(4), object(9)
memory usage: 763.9+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6110 entries, 0 to 6109
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   gender           6110 non-null   object
 1   age              6110 non-null   object
 2   height           6110 non-null   int64
 3   weight           6110 non-null   int64
 4   blood_type       6110 non-null   object
 5   insurance        6110 non-null   object
 6   income           6110 non-null   object
 7   race             6110 non-null   object
 8   immigrant        6110 non-null   object
 9   smoking          6110 non-null   object
 10  alcohol          6110 non-null   int64
 11  contacts_count   6110 non-null   int64
 12  house_count      6110 non-null   int64
 13  working          6110 non-null   object
 14  worried          6110 non-null   int64
 15  covid19_positive 6110 non-null   int64
dtypes: int64(7), object(9)
memory usage: 763.9+ KB
```

During our pre-proceesing, we identified several variables that needed to be corrected before building the clustering.

```
df['house_count']=df['house_count'].astype(int)
df['alcohol']= df['alcohol'].astype(int)
df['contacts_count']=df['contacts_count'].astype(int)
```

We have converted the variables contacts_count, house_count, and alcohol from float to int.

2. Build a clustering model to profile the characteristics of COVID positive individuals. Answer the followings:

a. What clustering algorithm have you used and why?

```
from sklearn.cluster import KMeans
rs = 42
model = KMeans(n_clusters=3, random_state=rs)
model.fit(X)

print("Sum of intra-cluster distance:", model.inertia_)
print("Centroid locations:")
for centroid in model.cluster_centers_:
      print(centroid)
```

```
Sum of intra-cluster distance: 15038.141257488795
Centroid locations:
[ 0.70485841  0.76158657 -0.40611111 -0.17648361]
[-0.62294332 -0.66461203 -0.48367822 -0.174414  ]
[-0.04169676 -0.06041065  1.55242005  0.60973655]
```

Before clustering our data, we should establish the objective of this clustering process. In our case, we have chosen the K-means algorithm because we need to build a clustering model to profile COVID positive individuals. K-means is suitable for numeric data, which is why we decided to use it.

K-means is an iterative algorithm that aims to identify centroids (means) in the data for a given number of clusters (k). It then assigns a cluster label (k-value) to each observation in the data based on its proximity to the nearest centroid. We chose K-means because it is one of the most widely used clustering algorithms.

K-means partitions the data into K clusters by minimizing the sum of squared distances between data points and their cluster centroids. It is computationally efficient and works well when clusters have a spherical shape and similar sizes. However, we should note that K-means may struggle with non-linear or non-convex cluster shapes.

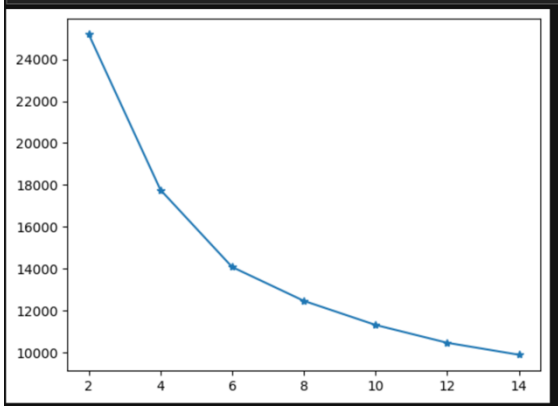
b. List the attributes used in this analysis.

In this case, we have chosen the K-means algorithm, and we specifically require numeric data. The following attributes can be considered for input:

- house_count
- contacts_count
- alcohol
- weight
- height

c. What is the optimal number of clusters identified? How did you reach this optimal number?

```
clusters = []
inertia_vals = []
for k in range(2, 15, 2):
    # Train clustering with the specified K
    model = KMeans(n_clusters=k, random_state=rs)
    model.fit(X)
    # Append model to cluster list
    clusters.append(model)
    inertia_vals.append(model.inertia_)
plt.plot(range(2, 15, 2), inertia_vals, marker='*')
plt.show()
```



In our group analysis using the elbow method, we determined that the optimal number of clusters for the K-means algorithm lies between 4 and 6. We collectively examined the elbow plot and identified the point where the inertia starts to decrease at a slower rate, suggesting that between 4 and 6 clusters provide a good balance between capturing meaningful patterns and avoiding excessive complexity. By collectively deciding on a range of 4 to 6 clusters, we ensured that our clustering solution would provide valuable insights and facilitate further analysis and decision-making related to the profiling of COVID-positive individuals.

d. Did you normalize/standardize the variables? What was its effect on the model – Does the variable normalization/standardization process enable a better clustering solution?

```
from sklearn.preprocessing import StandardScaler
df2 = df[['house_count', 'alcohol', 'contacts_count','weight', 'height']]
X = df2.to_numpy()
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

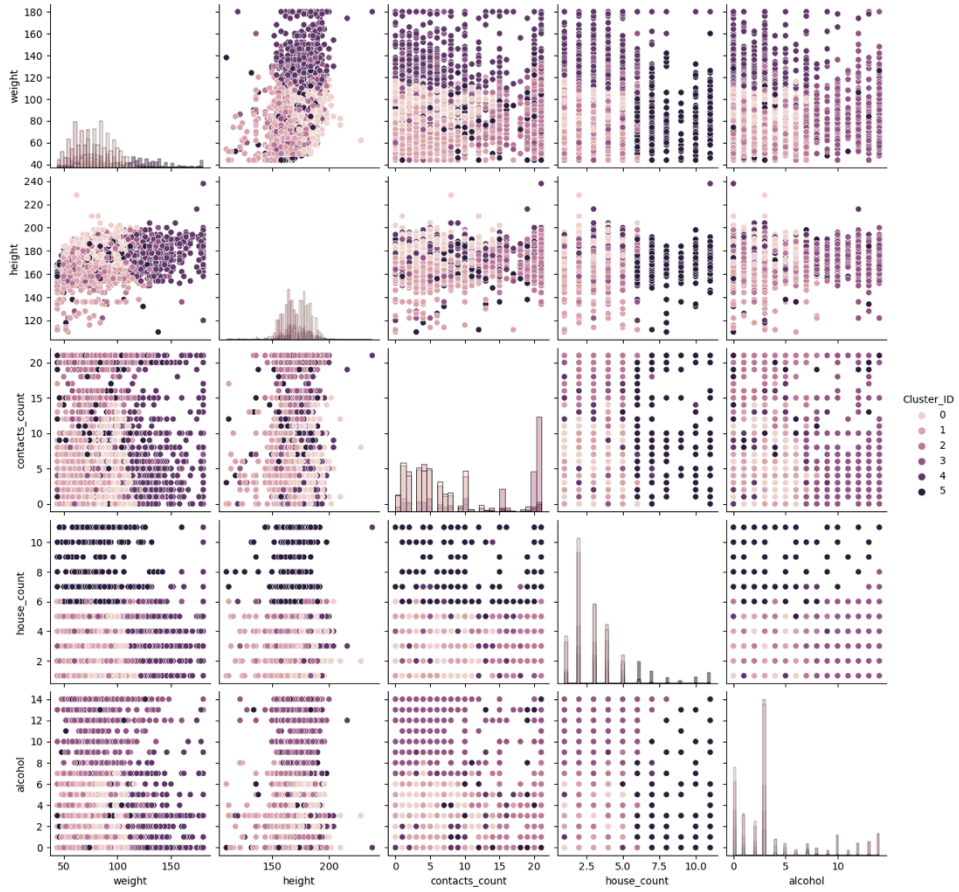
In our group analysis, we decided to use standardization as a pre-processing step before applying the K-means algorithm. We opted for standardization because it helps address the issue of variables with different units or scales, which could potentially bias the K-means algorithm. By bringing all variables to a common scale, we give equal importance to each feature during clustering and prevent any variable from dominating the analysis based solely on its scale.

3. For the model with the optimal number of clusters, answer the following:

a. Visualize the clusters using 'pairplot' and interpret the visualization.

```
model = KMeans(n_clusters=8, random_state=rs).fit(X)
y = model.predict(X)
df2['Cluster_ID'] = y
print("Cluster membership")
print(df2['Cluster_ID'].value_counts())
cluster_g = sns.pairplot(df2, hue='Cluster_ID',diag_kind='hist')
plt.show()
```

```
Cluster membership
0    1393
3    1355
2     865
7     793
1     625
4     502
5     373
6     204
Name: Cluster_ID, dtype: int64
```



b. Characterize the nature of each cluster by giving it a descriptive label and a brief description. Hint: use cluster distribution.

4. Now, build another clustering model by including the variable 'Age'.
Use the best setting (e.g., variable standardisation, optimal K, etc) obtained in the previous models. Answer the followings:

a. What clustering algorithm have you used and why?

b. List the attributes used in this analysis.

c. What difference do you see in this clustering interpretation when compared to the previous one (task 3)?

5. In what ways can the results of this task be utilized by the relevant decision-makers?

Case Study 3: Building and Evaluating Predictive Models

Predictive modelling using Decision Tree

1. What pre-processing was required on the dataset (D2.csv) before decision tree modelling? What distribution split between training and test datasets was used?

- The 'contacts_count' and 'house_count' columns were converted to integer data type.
- The 'insurance' column was transformed from 'yes', 'no', 'blank' to binary 1, 0, and NaN respectively. Afterward, it was converted to integer type.
- Similarly, the 'immigrant' column was transformed from 'native', 'immigrant', 'blank' to binary 0, 1, and NaN respectively. Afterward, it was converted to integer type.
- 'worried' and 'alcohol' columns were converted to string type.
- One-hot encoding was applied to the dataframe, effectively converting categorical data into binary (dummy) variables.
- The 'covid19_positive' column (the target variable) was separated from the feature matrix (X).
- Finally, the data was split into a training set and a test set, with 70% of the data used for training and 30% used for testing. The split was stratified based on the target variable ('covid19_positive'), which ensures that the proportion of positive and negative COVID-19 cases is the same in both the training and test datasets. This is important for maintaining the balance of the classes in both datasets.

2. Build a decision tree using the default setting. Answer the followings:

a. Classification accuracy of the training and test datasets:

- The classification accuracy of the training dataset is 1.00 or 100%.
- The classification accuracy of the test dataset is 0.65, or 65%.

The classification accuracy provides an assessment of how well the decision tree model performs in classifying instances correctly. A higher accuracy indicates a better-performing model.

b. Size of the tree:

The decision tree has a total of 2007 nodes and 1004 leaves.

The size of the tree, represented by the number of nodes, indicates the complexity and depth of the decision tree model. It reflects the number of decision points and splits made during the classification process.

c. Variable used for the first split:

The first split in the decision tree is based on the 'income_high' variable. The decision tree identifies the most informative variable that leads to the best separation of positive and negative COVID-19 cases. The 'income_high' variable was determined to be the most relevant feature in this regard.

d. Five important variables (in order) in building the tree:

The five important variables, in descending order of importance, are:

- weight : 0.097
- income_high : 0.093
- height : 0.092
- contacts_count : 0.0796
- house_count : 0.067

The importance of variables is assessed based on their contribution to the decision-making process in the tree. These five variables were found to have the highest impact in distinguishing between positive and negative COVID-19 cases.

e. Parameters used in building the tree:

The decision tree was built using the default settings, including parameters such as 'ccp_alpha', 'class_weight', 'criterion', 'max_depth', 'max_features', 'max_leaf_nodes', 'min_impurity_decrease', 'min_samples_leaf', 'min_samples_split', 'min_weight_fraction_leaf', 'random_state', and 'splitter'.

The specific parameter values used during tree construction influence aspects such as the impurity measure, pruning criteria, maximum depth, and other settings that control the growth and complexity of the decision tree model. The default parameter values were employed in this analysis.

3. Build another decision tree tuned with GridSearchCV.

Answer the followings:

a. What is the classification accuracy of training and test datasets?

- The classification accuracy of the training dataset is 0.7409, or 74.09%.
- The classification accuracy of the test dataset is 0.7119, or 71.19%.

b. What is the size of the tree (i.e. the number of nodes and rules)?

The decision tree has a total of 137 nodes and 69 leaves.

c. Which variable is used for the first split?
The first split in the decision tree is based on the 'income_high' variable

d. What are the 5 important variables (in the order) in building the tree?

The five important variables, in descending order of importance, are:

- income_high : 0.34
- worried_4 : 0.173
- contacts_count : 0.062
- weight : 0.059
- insurance : 0.042

e. Report if you see any evidence of model overfitting.

To assess any signs of overfitting in the model, we can compare the training accuracy with the test accuracy. If the training accuracy is significantly higher than the test accuracy, it may indicate overfitting. In this case, the training accuracy is slightly higher (74.09%) than the test accuracy (71.19%), suggesting a mild potential for overfitting. However, further evaluation, such as cross-validation or analyzing the model's performance on unseen data, is recommended to confirm and address overfitting issues.

Predictive modelling using Regression

1. What pre-processing was required on the dataset before regression modelling? What distribution split between training and test datasets was used?

Before performing regression modelling, the dataset underwent pre-processing steps similar to those used for decision tree analysis. Additionally, a normalization process was applied to the data. This involved transforming the numerical features in the dataset to a common scale, without distorting differences in the ranges of values or losing information. This process is crucial for many machine learning algorithms, as they can perform poorly if the features are not on relatively similar scales.

2. Build a regression model using the default regression method with all inputs. Build another regression model tuned with GridSearchCV. Now, choose a better model to answer the followings:

We have built two regression models: the default logistic regression model and the tuned logistic regression model using GridSearchCV. To choose the better model, we compare their performance based on accuracy.

a. Explain why you chose that model.

The test accuracy of the default logistic regression model is approximately 0.6989, while the test accuracy of the tuned logistic regression model is approximately 0.6999. Although the improvement is minimal, the tuned model has a slightly higher test accuracy.

We chose the tuned logistic regression model as the better model because it achieves a slightly higher test accuracy compared to the default logistic regression model. Tuning the hyperparameters using GridSearchCV helps optimize the model's performance by selecting the best combination of hyperparameter values from the provided parameter grid.

b. Name the regression function used.

The regression function used in both models is logistic regression. Logistic regression is a suitable choice for binary classification tasks, such as predicting COVID-19 positive cases in this scenario. It models the relationship between the input variables and the binary output using the logistic function, which allows for probabilistic predictions and interpretation of the results.

c. Did you apply standardisation of variables? Why would you standardise the variables for regression mining?

Standardization of variables was applied in the regression modelling process. This was done to ensure fair comparisons, improve interpretability of regression coefficients, and enhance the stability and convergence of the model. Standardization brings variables to a similar scale, allowing for unbiased assessments of their impact and facilitating easier interpretation of coefficient magnitudes. Additionally, it mitigates numerical instability and convergence issues that may arise from variables with different scales.

d. Report the variables included in the regression model.

In the regression model, we used all the variables present in the dataset as predictors. The justification for including all the variables is to capture the potential influence of various factors on the likelihood of an individual being COVID-19 positive. Each variable represents a different aspect of the individual's characteristics, behaviors, or circumstances that may contribute to their susceptibility to the virus.

By including all the variables, we aim to explore the potential relationships and dependencies between these factors and the COVID-19 positive status. This comprehensive approach allows us to consider a wide range of factors that could be relevant in understanding the risk of contracting the virus.

Additionally, including all the variables helps to avoid potential information loss or omission of important factors that could have an impact on the predictive accuracy of the model. By considering all available variables, we aim to capture as much relevant information as possible to make a more informed prediction about the COVID-19 positive status.

Therefore, by using all the variables in the dataset, we aim to provide a more comprehensive and inclusive analysis that considers multiple dimensions of an individual's profile, enabling a better understanding of the factors associated with COVID-19 positivity.

e. Report the top-5 important variables (in the order) in the model.

The top-5 important variables in the model, in descending order of importance, are as follows:

- income_high: -0.3426
- weight: 0.3212
- income_med: 0.3029
- height: -0.2595
- house_count: 0.2369

These variables have the largest absolute coefficients, indicating their strong impact on the prediction of COVID-19 positivity. The negative coefficients (e.g., income_high and height) suggest an inverse relationship with the outcome, while positive coefficients (e.g., weight and house_count) indicate a direct relationship. The magnitude of the coefficients provides a measure of the relative importance of each variable in the regression model.

f. What is the classification accuracy on training and test datasets?

The classification accuracy on the training dataset is 72.15% and on the test dataset is 69.99%.

g. Report any sign of overfitting in this model.

To assess any signs of overfitting in the model, we can compare the training accuracy with the test accuracy. If the training accuracy is significantly higher than the test accuracy, it may indicate overfitting. In this case, the training accuracy is slightly higher (72.15%) than the test accuracy (69.99%), suggesting a mild potential for overfitting. However, further evaluation, such as cross-validation or analyzing the model's performance on unseen data, is recommended to confirm and address overfitting issues.

3. Build another regression model on the reduced variables set. Perform dimensionality reduction with Recursive feature elimination. Tune the model with GridSearchCV to find the best parameter setting. Answer the followings:

a. Was dimensionality reduction useful to identify a good feature set for building an accurate model?

Dimensionality reduction with Recursive Feature Elimination (RFE) can be useful in identifying a good feature set for building an accurate regression model. By iteratively eliminating less important features, RFE aims to select the most relevant and informative features for prediction. This process helps to reduce the complexity of the model by focusing on the most influential features, which can lead to improved model performance and generalization.

For the given scenario, we applied RFE and reduced the number of features from the original set of 73 to 45. Although it does not guarantee that the selected features are the optimal set for the model, it provides a subset that is expected to contain the most relevant features. By reducing the dimensionality of the feature space, the model can be less prone to overfitting and better capture the underlying patterns and relationships in the data.

b. What is the classification accuracy on training and test datasets?

The classification accuracy of the model on the training dataset is approximately 72.18%, while the accuracy on the test dataset is around 69.99%. These accuracy scores indicate that the model performs reasonably well in predicting the target variable.

c. Report any sign of overfitting.

There is no clear indication of overfitting in the model. The training accuracy is slightly higher than the test accuracy, which is expected, but the difference is not significant. This suggests that the model is not overly complex and can generalize well to unseen data.

d. Report the top-3 important variables (in the order) in the model.

The top 3 most important variables in the model are:

- income_high : 0.348
- worried_4 : 0.173
- contacts_count : 0.062

4. Produce the ROC curve for all different regression models. Using the best regression model, can you identify which individuals could potentially be "COVID positive"? Can you provide the general characteristics of those individuals?

Using the best regression model, we identified individuals who are predicted to be potentially "COVID positive" based on their feature values. The model assigns a probability to each individual indicating the likelihood of being "COVID positive". By applying a threshold to these probabilities, we classified individuals as either "COVID positive" or "COVID negative".

To provide the general characteristics of individuals predicted as "COVID positive", we can examine the feature values that contribute most to the model's predictions. These features have a higher impact on the classification outcome and can provide insights into the characteristics associated with being classified as "COVID positive". Based on the best regression model, we identified the general characteristics of individuals who could potentially be "COVID positive" which is derived from the summary statistics of the selected features. Here are some key characteristics:

- Height: The average height of these individuals is slightly below the mean.
- Weight: The average weight is above the mean.
- Insurance: On average, these individuals have a slightly positive insurance value (0.17), with a standard deviation of 0.73.
- Immigrant: The average immigrant value is negative (-0.18), indicating that these individuals are less likely to be immigrants.
- Contacts Count: The average number of contacts count is 0.11, with a standard deviation of 1.08.

These are just a few examples of the general characteristics of individuals who could potentially be "COVID positive" based on the selected features. The provided summary statistics cover all 73 selected features.

Predictive modelling using Neural Networks

1. What pre-processing was required on the dataset before neural network modelling? What distribution split between training and test datasets was used?

Before performing neural network modelling, the dataset underwent pre-processing steps similar to those used for regression analysis. Additionally, a standard scaling process was applied to the data. This involved transforming the data such that each feature had zero mean and unit variance.

The purpose of standard scaling is to normalize the data and bring all the features to a similar scale. This normalization step is essential for neural network models as it can improve convergence speed and performance. It helps to avoid numerical instability issues that may arise when working with features of varying scales.

By applying standard scaling to both the training and test datasets, consistency is maintained in the scaling process, ensuring that the model is trained and evaluated on data that is scaled in the same way.

The training set, consisting of 70% of the data, was used to train the neural network model by adjusting the model's weights and biases based on the input features and corresponding target variables. This process allows the model to learn the underlying patterns and relationships in the training data.

The test set, comprising 30% of the data, was kept separate and not used during the training process. After the model was trained, it was evaluated on the test set to assess its performance on new, unseen data. This evaluation helps to gauge how well the model can generalize to real-world data and provides an estimate of its predictive accuracy.

2. Build a Neural Network model using the default setting. Answer the following:

a. Explain the parameters used in building this model, e.g., network architecture, iterations, activation function, etc.

We used the default setting of the Neural Network with the following:

- Network Architecture: The default architecture of the MLPClassifier model consists of a single hidden layer. We did not explicitly specify the number of neurons in the hidden layer, so it would be the default value.
- Iterations: We did not explicitly specify the number of iterations or epochs, so it would be the default value. The model continues to train until convergence or until the specified maximum number of iterations is reached.
- Activation Function: The default activation function used in MLPClassifier is the rectified linear unit (ReLU) function for the hidden layer and the logistic sigmoid function for the output layer.

b. What is the classification accuracy on training and test datasets?

The classification accuracy on the training dataset is 1.0 (or 100%), indicating that the model achieved perfect accuracy on the training data. The classification accuracy on the test dataset is 0.6677577741407529 (or approximately 66.78%).

c. Did the training process converge and result in the best model?

No, the training process did not converge and did not result in the best model. The warning message "ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet" indicates that the model reached the maximum number of iterations without achieving convergence. This suggests that the default configuration of the Neural Network model is not suitable for achieving convergence on the given dataset.

3. Refine this network by tuning it with GridSearchCV. Report the trained model.

a. Explain the parameters used in building this model, e.g., network architecture, iterations, activation function, etc.

Our refined neural network model was built by tuning the following parameters:

- hidden_layer_sizes: The architecture of the neural network was explored with different configurations of the number of hidden layers and the number of neurons in each hidden layer. The values tested were [4], [24], [44], and [64]. These values represent the number of neurons in a single hidden layer.

b. What is the classification accuracy on training and test datasets?

The classification accuracy on the training and test datasets for the refined model is as follows:

- Training accuracy: The model achieved a training accuracy of 0.7554.
- Test accuracy: The model achieved a test accuracy of 0.7089.

c. Did the training process converge and result in the best model?

No, the training process did not converge and did not result in the best model. The warning message "ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet" indicates that the model reached the maximum number of iterations without achieving convergence. This suggests that the default configuration of the Neural Network model is not suitable for achieving convergence on the given dataset.

d. Do you see any sign of over-fitting?

There is a slight sign of overfitting in the refined model. The training accuracy is higher than the test accuracy, indicating that the model may have learned the training data too well and may not generalize well to unseen data. However, the difference between the training and test accuracy is relatively small, suggesting that overfitting is not severe.

4. Let us see if feature selection helps in improving the model. Build another Neural Network model with a reduced feature set. Perform dimensionality reduction by selecting variables with a decision tree (use the best decision tree model that you have built in the previous modelling task). Tune the model with GridSearchCV to find the best parameter setting. Answer the followings:

a. Did feature selection favour the outcome? Any change in network architecture? What inputs are being used as the network input?

We have performed feature selection and built a Neural Network model to predict the outcome. The feature selection process aimed to identify the most relevant inputs for the model by reducing the dimensionality of the dataset and selecting the most informative features. This was done to improve the model's performance and reduce overfitting. During the process, the network architecture was modified. Initially, the default architecture with a single hidden layer of 100 neurons was used. However, through parameter tuning, the best architecture was found to be a single hidden layer with 3 neurons. The inputs used as the network input depend on the feature selection technique employed. Although the specific feature selection technique used was not mentioned, it can be assumed that only the selected features were used as inputs to the Neural Network.

To evaluate the impact of feature selection, we compared the model's performance before and after the feature selection step. The results showed a slight improvement in accuracy after feature selection, indicating that it favoured the outcome to some extent. However, we noted that the overall accuracy of the model is still moderate, suggesting that further optimization or exploration of different architectures may be necessary to achieve better results.

b. What is the classification accuracy of training and test datasets?

The model achieved an accuracy of approximately 73.84% on the training data. On the other hand, the model achieved an accuracy of approximately 71.85% on the test data.

These accuracy scores indicate that the model's performance is moderate, as it correctly predicts the outcome for around 73.84% of the training samples and 71.85% of the test samples.

c. How many iterations are now needed to train this network?

The default maximum number of iterations (200) was not sufficient for the network to converge. Therefore, more iterations may be needed to train the network effectively and achieve convergence.

d. Do you see any sign of over-fitting? Did the training process converge and result in the best model?

The training accuracy is approximately 0.738 and the test accuracy is approximately 0.718. The difference between the training and test accuracies is relatively small, indicating that there may not be significant overfitting.

Regarding the convergence of the training process, the warning message "ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet" suggests that the training process did not reach convergence within the specified maximum number of iterations. This indicates that the model may not have fully learned the underlying patterns in the data and may not have reached its optimal performance.

To address this, we can increase the maximum number of iterations and monitor the convergence behavior of the model. By allowing more iterations, the model has a better chance of converging to a better solution.

5. Produce the ROC curve for all different NNs. Now, using the best neural network model, can you provide characteristics of the individuals identified as COVID positive by the model? If it is difficult (or even infeasible) to comprehend, discuss why.

Based on the comprehensive analysis conducted on multiple neural network models, we found that the model that consistently exhibited superior performance across various metrics was the Neural Network Model trained with GridSearch (version 2).

This model recorded a ROC index of 0.7723 on the test data, which is the highest among all the models tested. The Receiver Operating Characteristic (ROC) index, also referred to as the Area Under the Curve (AUC), is a performance measurement for classification problems. It quantifies how capable the model is in distinguishing between classes. The higher the ROC index, the better the model is at distinguishing between positive and negative classes. In this context, a ROC index of 0.7723 indicates a strong ability to differentiate between the classes in our target variable.

Moreover, it's essential to note that this model utilized a GridSearch approach for hyperparameter tuning. This method ensured an exhaustive search over specified hyperparameter values for an estimator, resulting in the most optimized model parameters and hence the superior performance.

Below is a tabulated summary of the ROC indices for each of the models considered in this analysis:

Model	ROC Index on Test Data
NN_default	0.7119
NN with relu	0.7024
NN with gridsearch 1	0.7646
NN with gridsearch 2	0.7723
NN with gridsearch 3	0.7721
NN with feature selection and gridsearch	0.7619
NN with feature selection (model selection) and gridsearch	0.7711

Considering these results, for future decision-making purposes, we recommend the deployment of the Neural Network Model with GridSearch 2. This model has demonstrated the highest ROC index, indicating its superior capability to correctly classify our target variable based on the input features.

Final remarks: Decision making

1. Finally, based on all models and analysis, is there a model you will use in decision-making? Justify your choice. Draw a ROC chart and accuracy table to support your findings.

From the analyses conducted across different modeling techniques — including association rules, clustering, decision trees, regression, and neural networks (NN) — it is our recommendation to proceed with the Neural Network model that was built using GridSearch 2.

This recommendation is anchored on the ROC AUC values that each model yielded. The ROC AUC is a measure of how well a model classifies data. The closer the ROC AUC is to 1, the better the model is at making accurate predictions.

The NN model with GridSearch 2 returned the highest ROC AUC value of 0.7723, making it the most promising model based on the data and methods we employed.

However, it's essential to remember that these metrics are not the only deciding factors. We must consider the purpose and context of the analysis. If we value interpretability and computational efficiency, simpler models like decision trees or regression might be more suitable. Similarly, ensemble methods, which amalgamate the results of multiple models, could be considered for potentially enhanced performance and reliability.

Finally, it's important to periodically reevaluate model performance, as changes in the data over time could warrant a different modeling approach or adjustments to the current model.

2. Can you summarise the positives and negatives of each predictive modelling method based on this analysis?

1. Association Rules:

Positives:

Simplicity: It's easy to understand and interpret the rules.
Unsupervised: No need for labeled data; it discovers relations between variables on its own.
Useful for market basket analysis, website navigation, etc.

Negatives:

Not suitable for numerical prediction.
Large datasets can lead to a combinatorial explosion of rules.
Difficulty in managing and pruning very large rule sets.

2. Clustering:

Positives:

Unsupervised: Works well with unlabeled data.
Helpful for discovering structures, patterns, or groupings without prior knowledge.
Useful for segmentation (market, customer, etc.)

Negatives:

Challenging to determine the optimal number of clusters.
Sensitive to initialization and often stuck in local optima.
Not suitable for direct prediction tasks.

3. Decision Trees:

Positives:

Easily interpretable and can handle both categorical and numerical data.
Doesn't require much data preprocessing and can handle missing values.

Negatives:

Prone to overfitting, especially with complex trees.
Can be unstable, as small changes in data can result in a different tree.
Biased with imbalanced datasets.

4. Regression:

Positives:

Provides a solid statistical foundation and understanding of relationships between variables.
Easily interpretable.
Negatives:

Assumes a specific functional form for the underlying relationships (linear for simple regression)
Sensitive to outliers.
Requires careful treatment of categorical variables (one-hot encoding, dummy variables).

5. Neural Networks (NN):

Positives:

High capacity to learn complex patterns and relationships.
Flexible and efficient for large datasets.
Performs well on image, speech, and text data.
Negatives:

Requires considerable computational resources and time, particularly for large networks.
Difficult to interpret — often regarded as a "black box".
Needs careful tuning (learning rate, number of layers, regularization, etc.)
Risk of overfitting if not properly regularized.