Database Lab Summer Semester 2022

Group: 3

Members: Sagnik Sarkar,
Adeel Ahmed,
Muhammad Usman

1. Data Preparation and Understanding

Below is the example of loading the dataset in the Jupyter Notebook using the pandas library.

```
In [1]: import pandas as pd
```

```
In [4]: energyTrain_df = pd.read_csv("D:\RCSE_WS21\RCSE_S22\Lab_Training_DB\energy_train_1d_dirty\energy_train_1d_dirty.csv", sep=';')
        energyTrain_df.head(5)
```

Out[4]:

	eqp_id	value_max	value_avg	value_min	from_ts	to_ts
0	4531	8260.667969	6139.326660	3917.091309	1999-08-01 00:00:05	1999-08-01 00:00:10
1	4531	8282.814453	5756.913086	4269.579102	1999-08-01 00:00:10	1999-08-01 00:00:15
2	4531	8261.721680	6040.891113	3910.502441	1999-08-01 00:00:15	1999-08-01 00:00:20
3	4531	7906.354004	5284.814941	3521.319824	1999-08-01 00:00:20	1999-08-01 00:00:25
4	4531	7532.439941	5586.362793	3552.864746	1999-08-01 00:00:25	1999-08-01 00:00:30

```
In [7]: wip2d_df = pd.read_csv("D:\RCSE_WS21\RCSE_S22\Lab_Training_DB\wip_2d_dirty\wip_2d_dirty.csv", sep=';')
        wip2d_df.head(5)
```

Out[7]:

	segment_name	duration	valid_from	valid_to	conditioning_recipe	eqp_id	ei_ctrljob_id	entity_id	wafer_id	lot_id	prodspec_i
0	TheHappeningHappened	0.001100	1999-08-01 05:28:32	1999-08-01 05:30:06.999999	1.001096e+09	1463.0	1001096028	872362953	1.001096e+09	1.001096e+09	Nal
1	NothingToDoHere	0.004005	1999-08-01 05:28:32	1999-08-01 05:34:18	1.001096e+09	1463.0	1001096028	872362953	1.001096e+09	1.001096e+09	Nal
2	TheHappeningHappened	0.001100	1999-08-01 05:38:28	1999-08-01 05:40:02.999999	1.001096e+09	1463.0	1001096028	882226468	1.001096e+09	1.001096e+09	Nal
3	NothingToDoHere	0.000891	1999-08-01 05:38:28	1999-08-01 05:39:44.999999	1.001096e+09	1463.0	1001096028	882226468	1.001096e+09	1.001096e+09	Nal
4	TheHappeningHappened	0.001100	1999-08-01 05:28:33	1999-08-01 05:30:07.999999	1.001096e+09	1463.0	1001096028	1382710593	1.001096e+09	1.001096e+09	Nal

2. Theory - Data Cleaning and Analytics

1. *What are typical data cleaning steps? Name and explain five of them.*

Data cleaning or cleansing, is the process of correcting and deleting inaccurate records from a database or a table. With effective data cleansing, all data sets should be consistent and devoid of any errors which will result in proper analysis of the data. According to a study done by IBM, poor quality data costs 3.1 trillion dollars per year in the US alone. The cost for fixing the issues increases exponentially with time(1-10-100 quality principle). In order to achieve a cleaner data, below mentioned steps can be followed:

Step 1: Manage duplicate data

Data cleansing checks for duplicate data and either removes them or merges them using the deduplication measures.

Duplicate observations take place when we are trying to syndicate data from multiple sources.
Duplicate data can lead to skewed results which in turn can lead to wrong analysis of the data.

Step 2: Fix Structural Errors
Structural errors mainly occur during measurement, data transfer or maintenance activities. It may include strange naming conventions, typos, or incorrect capitalization which can lead to mislabelled categories, inconsistent data, duplicates.
For instance, we may find “N/A” and “Not Applicable” however, both should be analyzed as the same category.

Step 3: Filter unwanted outliers
Outliers are values that stand out and are significantly different from the others and can provide more insights into our model as compared to other observations. Outliers can be a result of scraping a bigger dataset or it can also come from measurement error that is unlikely to be real data.
It is highly recommended not to remove an outlier unless the user knows that it’s a mistake.
For example, a user is researching their app users’ age, and finds entries like 72 and 2.The first one might be a senior citizen who is up to date with the technology however, the second case is most likely an error since toddlers don’t use apps.

Step 4: Handle Missing Data
We can deal with it in several ways:
1. Firstly, we can drop observations that have missing values, but doing this will drop or lose information, so we need to be very careful before removing the data.

2. Secondly, although we can input missing values based on other observations, there is an opportunity to lose integrity of the data as we tend to operate from assumptions and not actual observations.

3. Thirdly, by altering the mechanism of using the data we can effectively navigate null values.

4. Finally, by flagging a particular data, it could become informative, especially if there is a pattern in play.

For instance, we conduct a survey, and most women refuse to answer a particular question. In that case we can flag the data in order to deal with those insights

1. For numeric data – just put in 0.
2. For categorical data – introduce the ‘missing’ category.

Step 5: Validate and QA
At the end of the data cleaning process, you should be able to answer these questions as a part of basic validation:

- Does the data make sense?
- Does the data follow the appropriate rules for its field?
- Does it prove or disprove your working theory, or bring any insight to light?
- Can you find trends in the data to help you form your next theory?
- If not, is that because of a data quality issue?

False conclusions because of incorrect or "dirty" data can inform poor business strategy and decision-making.

2. *Why is it important to choose and enforce optimal column data types? Please describe the impact of data types for both data cleaning and data analytics*

Importance of choosing and enforcing optimal column data types ensures optimum storage and performance by ensuring the correct execution plan.

Below are the constraints that can be set to to a column:

Data-Type Constraints: values in a particular column must be of a particular data type, e.g., boolean, numeric, date, etc.

Range Constraints: typically, numbers or dates should fall within a certain range.

Mandatory Constraints: certain columns cannot be empty.

Unique Constraints: a field, or a combination of fields, must be unique across a dataset.

Set-Membership constraints: values of a column come from a set of discrete values, e.g. enum values. For example, a person's gender may be male or female.

Foreign-key constraints: as in relational databases, a foreign key column can't have a value that does not exist in the referenced primary key.

Regular expression patterns: text fields that have to be in a certain pattern. For example, phone numbers may be required to have the pattern (999) 999–9999.

The impact of data types for both data cleaning and data analytics are mentioned below:

When dealing with datasets, the category of data type plays an important role to determine which pre-processing strategy would work for a particular set to get the appropriate results or which type of statistical analysis should be done in order to obtain the best results.
Also, datatypes are an important concept because statistical methods can only be used with certain data types. For example, we have to analyze numerical (continuous data) differently than categorical data otherwise it would result in a wrong analysis. Therefore, knowing the types of data, we are dealing with, enables us to choose the correct method of analysis.

3. *Why is the order and implementation of operations of a query so important (justification for query optimizer)? Which are typical relational operators where this is crucial and what are the basic optimization rules?*

A single query can be written in different forms and can be executed through different algorithms. Hence, the query optimizer comes into picture. The query optimizer evaluates different possible query plans and analyzes the most efficient way to execute a given query.

The steps for the query optimization are mentioned below:

Firstly, the optimizer decides if any change to the form of the query is needed or not in order to generate a better execution plan.

Secondly, it estimates the cost of each plan based on statistics in the data dictionary and by examining multiple access methods, such as full table scan or index scans, different join methods, different join orders, and possible transformations.

Finally, the optimizer chooses the lowest-cost plan among all the different cost plans which is known as the execution plan.

The relational operators are mainly as: Projection, Selection, Union, Minus, Intersection, Join

Basic optimization rule mainly comprises Rule based optimization and Cost based optimization.

Rule Based optimization: This is an old technique which uses a set of rules to define how to execute a query.

Cost Based optimization: Its target is to produce the cheapest execution plan for each SQL statement.

4. *What are the ways to handle relations or results that are larger than the memory budget?*

There are two mainly two straightforward approaches that can be followed in order to handle large data sets:

Money costing solution: If a faster CPU and larger RAM is used which can handle the entire dataset or using a cloud memory can be used to handle the workload. But this solution is not a cost effective solution since as the data set increases, cost also increases.

Time costing solution: If the RAM is too small to handle the dataset, then the HDD can be used to handle the dataset which will be cost effective but it will be much slower than using RAM irrespective of using SSDs or Spinning Disks.

But if the above two solutions are not viable for the project where the budget is fixed and also the performance cannot be compromised, then the following techniques can be used:

1. **Compression:** Compressing a data means representing a data in a certain way which will use less memory. There are basically two types of compression: lossless compression and lossy compression.

    - **Lossless Compression:** Data is not lost in case of Lossless compression.

      This can be achieved by loading specific columns where we load only the relevant columns instead of the entire dataset and can reduce the memory used by the data.

      Another way to achieve this is by manipulating the data types. For instance if we know that in certain columns, Integer values do not exceed 32767, then *int16* and *int32* can be used which results in saving 75% of the memory.

      Sparse column representation is also another way to perform a lossless compression. Sparse column skips the rows containing '*NaN*'.

    - **Lossy Compression:** If Lossless compression is not enough, then another approach is there to tackle the memory budget issue but accuracy of the data is compromised. Lossy compression can be performed in two ways:

- **Modifying numeric values:** Sometimes full accuracy of numeric data is not needed for the analysis. Hence, in those cases the data can be truncated to *int16* or *int32* from *int64*.
- **Sampling:** Sometimes data is required for particular cases from the overall set. Then in only those cases, the rows are fetched and for other cases(rows) the data is ignored.

2. **Chunking:** Chunking is similar to Divide and Conquer strategy where the whole data is divided into smaller chunks and those individual chunks are analyzed independently and later results are compared and evaluated finally.

3. **Indexing:** Indexing just needs a part of the data instead of the whole data. Chunking is useful where multiple datasets are loaded separately. Indexing is similar to the index of a book, where you get necessary information about an aspect without needing to read the entire book.

Theory - Data Prediction

1. *What is meant by training data, validation data and test data? How should these typically be chosen?*

Machine learning data algorithms are data hungry algorithms as they learn from the available data. These algorithms find the relationships between data and make decisions around it.

Training data
Training data is the data that is used to teach the machine learning algorithm what is the corresponding expected value to a given input value. The model that is being trained uses this data again and again to learn the characteristics of the input data by training itself to produce the output that is expected by the user. The model adjusts itself after going over the data many times until it can understand the input data and gives out results that are expected by the user.

Validation data
During training when the model thinks that it has learnt what it is supposed to learn by going over all the training data we check our model accuracy by passing through new data into the model that it hasn't evaluated before. Validation data is the first test against unseen data. This allows us to evaluate how well the model makes predictions based on the new data. This gives us a good idea about how the learning process is going and does our model need more alterations.

Test data
After the model has done training and it gives out somewhat acceptable accuracy we put this model to another test. Here the testing data once again validates that it can make accurate predictions. The test data is completely new to the network and the network has not seen it during training and validation. The test data provides a final, of an unseen dataset to confirm that the ML algorithm was trained effectively.

Good Practice is that at least 60 percent of data should be used in training.
In General Cases we used 80 percent of the dataset in training and 10 percent for training and validation respectively [1].

2. *What is over- and underfitting? Why is this a problem and how can it be avoided?*

ML algorithms require training data to achieve an objective. The algorithm will analyze this training dataset, classify the inputs and outputs, then analyze it again. Training the model long enough will cause the algorithm to memorize all of the inputs and outputs in a training dataset. This memorizing problem is what we call overfitting as the network has just memorized everything and it will fail dramatically when new data is used as input to the network.
To avoid overfitting validation data is useful. Validation data provides an initial check that the model can return useful predictions in a real-world setting, which training data cannot do.

Underfitting happens when our model is unable to capture the relationship between the input and output variables accurately, generating a high error rate on both the training set and unseen data. The model cannot establish the dominant trend within the data, resulting in training errors and poor performance of the model.
To avoid underfitting we need to either increase the training time of the model or by increasing the number of features.

3. What is the difference between accuracy and precision?

Precision is defined as from all the actual expected outputs (positive), the percentage our model correctly predicted as positive. The precision measures the model's accuracy in classifying a sample as positive.

$$\boldsymbol{Precision} = \frac{\boldsymbol{True_{positive}}}{\boldsymbol{True_{positive}} + \boldsymbol{False_{positive}}}$$

Precision is a measure to determine if the false positives are safe to ignore or it causes catastrophic results to our system. For example; In spam detection, a false positive means that an email that is non-spam has been identified as spam. The email user might lose important emails if the precision is not high for the spam detector to detect. So by seeing the low precision rate we can conclude that the model is not suitable for this task..

Accuracy is the measure of how well our model is doing in predicting the output correctly. It is calculated as a percentage of all the data points positives and negatives and how much percentage our model is predicting correctly.

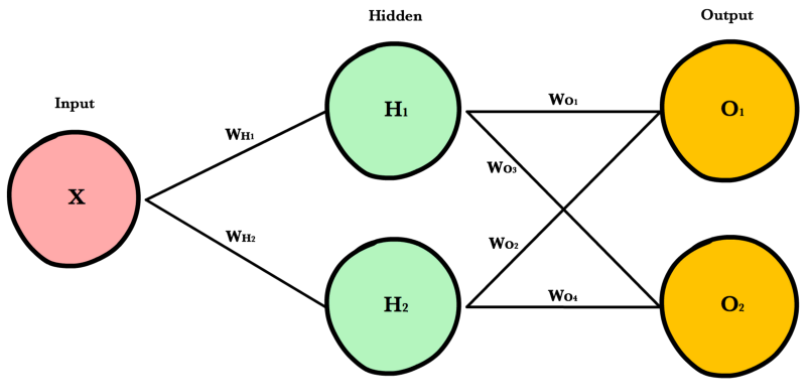
It is calculated as the ratio between the number of correct predictions to the total number of predictions.

$$Accuracy = \frac{True_{positive} + True_{negative}}{True_{positive} + True_{negative} + False_{positive} + False_{negative}}$$

Accuracy is used widely in checking the performance of the network and as high the accuracy of the network the better the network is performing.

4. What are the typical building blocks of a neural network? Please give a brief overview considering the terms neuron, activation function, layers and weights.

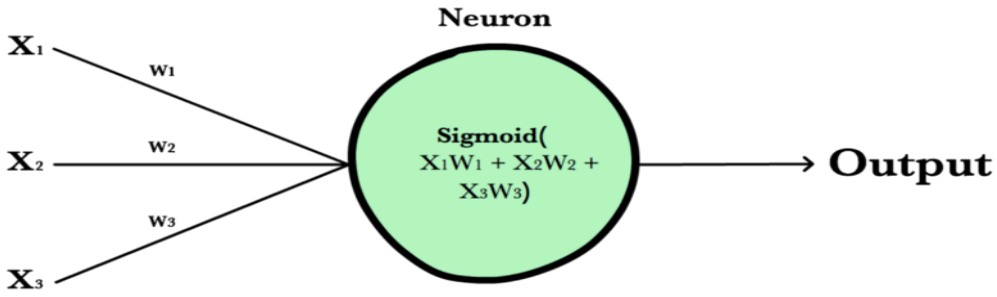
A neural network is a set of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates A neural network takes an input, passes it through multiple layers of hidden neurons, and outputs a prediction or result which represents the combined input of all the neurons.



[2]

Neural networks are trained iteratively in cycles and an error is calculated which is the difference between prediction and target. After computing this error each neuron weights are then updated relative to how much they contributed to the total error. This process is repeated again and again until the network error drops below an acceptable threshold.

Neuron:
A neuron as shown below intakes a group of weighted inputs, applies the activation function on it and then returns the output.



[2]

The neurons inputs can either be the features from a training dataset or the output from a previous layer's neurons. Each neuron is given a weight and then a non linear activation function is applied to the sum of weighted inputs and the result is passed to the next layer.

Weights:

Weights are values that control the strength of the connection between two neurons. We can interpret weight as the importance of one feature with respect to another one. The inputs are multiplied by weights, and that defines how much influence the input will have on the output.

Layers:

Input Layer

Hold the data your model will train on. Each neuron in the input layer represents a unique attribute in the dataset (e.g. hair color, height, completion etc.).

Hidden Layer

The layer/s between the input and output of the network are termed as 'hidden layers'. There are often multiple hidden layers in a network. Hidden layers are mostly fully-connected layers where each neuron receives input from all the previous layers till the last layer.

Output Layer

The last layer in a network is called the output later. It receives input from the previous hidden layer and it returns an output such as probability value representing the model's prediction.

Activation Functions:
An activation function defines how the weighted sum of the input is transformed into an output from a node in a layer of the network. Activation functions help the neural networks to model complex non-linear relationships. By modifying inputs with non-linear functions neural networks can model highly complex relationships between features. The purpose of the activation function is to introduce non-linearity into the output of a neuron. Popular activation functions include relu and sigmoid.

5. How does a neural network determine the output to a given input? What is the dominating type of operation in this computation and what hardware should be used as a consequence?

The neural network takes a number of inputs as explained above and passes it through a number of neurons in a hidden layer; where each neuron focuses on a specific trait of the data. So when we train our model what we are doing is that we are taking these traits and adjusting the weights and bias to an extent where our results are as close to the one expected.
For example we are tasked to make a medical diagnosis system.We have 5 symptoms as input which are blood pressure, temperature, cholesterol levels, glucose level and heartbeat. The output of what we expect is either the patient has heart disease or not, so we have two outputs. So the network will pass through a number of hidden neurons and where a weight will be calculated and final probability will be computed. This will be compared with the groundtruth value and the error will be calculated. This error will be back propagated to the network and weights of the neurons will be updated hence improving our network.
The dominating factor is the error computation and the backpropagation of the error to update the weights. These are computed in a form of matrix and these operations run in parallel to each other. The hardware required to do these computations fast and in parallel are by using GPUs. GPUs can do computations in parallel and much faster than ordinary CPUs.

6. Which ML model types might be appropriate for the data sets at hand? Please explain why you think so.

The algorithm that seems appropriate with the task in hand is K-Nearest Neighbors. In K-Nearest Neighbors the predictions are made by searching through the entire dataset for K most similar instances (neighbors). It then gives out the mean value as the output variable which is somewhat a very good estimation.
This is suitable for us as we have the data already and all we need to do is to predict the future value point and K nearest neighbors does a decent job in predicting the value by looking at all its neighboring values.

References

[1] https://www.applause.com/blog/training-data-validation-data-vs-test-data
[2] https://ml-cheatsheet.readthedocs.io/en/latest/nn_concepts.html