Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Information Technology

Czech University of Life Sciences Prague

Bachelor Thesis

Data Cleaning using Machine Learning

Ahmed Hassan

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

BACHELOR THESIS ASSIGNMENT

Ahmed Hassan

Informatics

Thesis title

Data cleaning using machine learning

Objectives of thesis

The main objective of the thesis is to develop machine learning model to help find errors in a data set and evaluate the accuracy of the model in a practical scenario.
Side objectives of the thesis include:
- Conduct study and analysis of available literature resources
- Collect the data set for experimental verification
- Analyze the data and find errors that need to be addressed (missing data, duplicate data, typos, etc.)
- Propose and develop a machine learning model to clean the data
- Measure the model's effectiveness to evaluate its usability

Methodology

The theoretical part of the thesis is based on a professional literature review, scientific blogs/tutorials and online sources regarding data cleaning and machine learning besides the predictive modelling. The practical part consists of collection of the testing data, analyzing the data and finding errors that need to be removed using data cleaning. The data will be preprocessed in order to train a machine learning model developed in Python programming language. The model will be experimentally validated and its accuracy measured. The final conclusions will be based on the results of both thesis parts.

The proposed extent of the thesis

40-50

Keywords

Machine learning, Data cleaning, Predictive models, Python, Preprocessing, Data analysis

Recommended information sources

AHMAD, L. *Agriculture 5.0 : artificial intelligence, IoT and machine learning.* Boca Raton: CRC Press, 2021. ISBN 978-0367646080.

GÉRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : concepts, tools, and techniques to build intelligent systems.* Beijing ; Boston ; Farnham ; Sevastopol ; Tokyo: O'Reilly, 2019. ISBN 978-1-4920-3264-9.

JANSEN, S. *Machine learning for algorithmic trading : predictive models to extract signals from market and alternative data for systematic trading strategies with Python.* Birmingham ; Mumbai: Packt, 2020. ISBN 978-1-83921-771-5.

TOOMEY, D. *Jupyter for data science : exploratory analysis, statistical modeling, machine learning, and data visualization with Jupyter.* Birmingham: Packt, 2017. ISBN 978-1-78588-007-0.

UNPINGCO, J. *Python for probability, statistics, and machine learning.* New York, NY: Springer Berlin Heidelberg, 2016. ISBN 9783319307152.

WITTEN, I H. – FRANK, E. *Data mining : practical machine learning tools and techniques.* San Francisco, Calif.: Elsevier Science [distributor], 2005. ISBN 978-0-12-088407-0.

Expected date of thesis defence

2022/23 SS – FEM

The Bachelor Thesis Supervisor

Ing. Jan Pavlík

Supervising department

Department of Information Technologies

Electronic approval: 14. 7. 2022	Electronic approval: 27. 10. 2022
doc. Ing. Jiří Vaněk, Ph.D.	**doc. Ing. Tomáš Šubrt, Ph.D.**
Head of department	Dean

Prague on 07. 03. 2023

Declaration

I declare that I have worked on my bachelor thesis titled "Data Cleaning using Machine Learning" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the bachelor thesis, I declare that the thesis does not break any copyrights.

In Prague on 15.3.2023 ___________________________

Acknowledgement

I would like to thank my supervisor Ing. Jan Pavlík for all the time he spent on helping me throughout the whole thesis, Wonderful advices, Mentoring and Providing the best ideas. And, I am deeply grateful to Allah for blessing me with my beloved parents who have been constant source of love, support, and guidance in my life. Their unwavering faith and devotion to Allah have been my source of strength and such a shining example for me to follow, and I owe my success to their tireless efforts and prayers. Asking Allah bless them with good health, happiness, and a long life. I pray that Allah showers his infinite mercy and blessings upon them, as they have always been a source of mercy for me. May Allah grant them a place in Jannah (Paradise) for their unending love, care, and sacrifice.

Data Cleaning using Machine Learning

Abstract

This thesis provides a general outlook on Data Analysis, Data Cleaning, Artificial Intelligence & Machine Learning. The main part focuses on building up a machine learning model that is used to find errors in any data set, and that will be considered with an application of machine learning known as predictive analytics. After the model cleans the errors, we will be able to evaluate the accuracy of the model practically. Firstly, Data are going to be collected from an online resource e.g., Kaggle.com, These data will then be analyzed to figure-out the type of errors needed to be considered e.g., Deleting duplicate data, Dropping/replacing missing data, Typos fixing, etc. Then we will apply the data pre-processing techniques e.g., Data cleaning, Sampling data, Imbalanced data. Then we will develop and apply the machine learning model to clean the data we pre-processed. Lastly, we will go with prediction and evaluate the model's efficiency for appraising its usability.

Keywords: Machine Learning, Data cleaning, Predictive models, Python, Pre-processing, Data analysis

Table of content

1 Introduction **9**

2 Objectives and Methodology **10**

2.1 Objectives 10

2.2 Methodology 10

3 Literature Review **11**

3.1 General Overview 11

3.1.1 Artificial General Intelligence 11

3.1.2 Artificial Intelligence 12

3.2 AI Classifications 13

3.2.1 Machine Learning 14

3.2.2 Example of classification model 15

3.2.3 Types & Application of predictive Machine Learning Models 15

3.3 Life Cylcle of an AI-Based Model 18

3.4 Machine Learning Prerequisites 20

3.5 Machine Learning in Wireless Sensor Network & Internet of Things 22

3.6 Supervised Learning 24

3.6.1 Decision Trees 24

3.6.2 Support Vector Machines (SVM) 24

3.6.3 Neural Networks 25

3.6.4 K-Nearest Neighour (K-NN) 25

3.6.5 Bayesian Learners 25

3.7 Unsupervised Learning 25

3.7.1 Principal Component Analysis 26

3.7.2 K-Means Clustering 26

3.8 Semi-Supervised Learning 26

3.9 Reinforcment Learning 27

3.10 Deep Learning & Artificial Neural Networks 27

3.11 Some Machine Learning's Applications 28

3.12 Some Machine Learning Applications 30

3.13 Data Cleaning in Machine Learning 30

4 Practical Part **33**

4.1 Problem definition & Steps to take 33

4.1.1 Pre-process the test dataset after collection 35

4.1.2 Build & Train the machine learning model 38

4.1.3 Fine-Tune and optimize the model 49

5 Results and Discussion **50**

5.1 Code definition / Pipeline output 50

5.1.1 Data evaluation and discussion about overall output 51
5.1.2 Prediction point of view (estimated salary as example & it's code snippet output for reuse) 51

6 Conclusion 55

7 References 57

List of figures

Figure 1 - Evolution of AI from the Year 1930 to the Year 2000, Source: (25) 12
Figure 2 – Subsets of Artificial General Intelligence, Source: (26) 14
Figure 3 – Machine Learning Life Cycle. Source: (27) 20
Figure 4 – AI, ML and Data Science Techniques, Source: (28) 21
Figure 5 – Machine Learning Types, Source: (29) 23
Figure 6 – Supervised Machine Learning, Source: (30) 24
Figure 7 – Deep Learning in a nutshell, Source: (31) 28
Figure 8 – Neural Networks, Source: (32) 29
Figure 9 - Applications of Machine Learning. Source: (33) 30
Figure 10 – Least enjoyable part of data science, Source: (34) 31
Figure 11 - Table used for model training, Bank Customers Churn, Source: (35) 36
Figure 12 - pipeline of missing data identification, Source: own processing 40
Figure 13 - inserting the columns, Source: own processing 41
Figure 14 - irrelevant data pipeline section, Source: own processing 41
Figure 15 - columns insertion, Source: own processing 42
Figure 16 - typos identifier in an object oriented programming paradigm, Source: own processing 43
Figure 17 - duplicated data code snippet, Source: own processing 44
Figure 18 - Model pipeline, Source: own processing 45
Figure 19 - data cleaning in a whole, Source: own processing 47
Figure 20 - Training Model sync, Source: own processing 47
Figure 21 - duplicated data solver using hash set, Source: own processing 48
Figure 22 - Training Response, Source: own processing 48
Figure 23 - output of the last pipeline, Source: own processing 50
Figure 24 - Results, Source: own processing 52
Figure 25 - predicted data console output, Source: own processing 54

1 Introduction

For the creation of Machine Learning (ML) systems that perform well and are accurate, high-quality data is often a need. However, due to erroneous inputs from manual data curation or inadvertent errors from automatic data gathering or generating procedures, data is rarely pure.

Real-world datasets frequently contain inconsistencies and gaps brought on by malfunctioning sensors or human mistake, for instance, which might have an effect on the machine learning systems based on those datasets. To solve this issue, we analyzed and proposed numerous state-of-the-art data cleaning methodologies based on a review of publications in the field of data management systems. Data cleaning is one of the most important steps in data analysis and the first phase of any machine learning project.

It is an important stage in making sure the dataset is free of inaccurate or false data. It can be carried out either manually using data wrangling tools or automatically using computer software. Data cleaning involves a number of steps that, when finished, prepare the data for analysis. The creation of efficient and effective data cleansing frameworks is gaining popularity due to its importance in many different fields.

In this study, the efficacy of some of the most current developments in data cleaning techniques is explored.

2 Objectives and Methodology

2.1 Objectives

The main objective of the thesis is to develop a machine learning model to help find errors in a data set and evaluate the accuracy of the model in a practical scenario.
Side objectives of the thesis include:

- Conduct study and analysis of available literature resources
- Collect the data set for experimental verification
- Analyze the data and find errors that need to be addressed (missing data, duplicate data, typos, etc.)
- Propose and develop a machine learning model to clean the data
- Measure the model's effectiveness to evaluate its usability

2.2 Methodology

The theoretical part of the thesis is based on a professional literature review, scientific blogs/tutorials and online sources regarding data cleaning and machine learning besides the predictive modelling. The practical part consists of collection of the testing data, analyzing the data and finding errors that need to be removed using data cleaning. The data will be preprocessed in order to train a machine learning model developed in Python programming language. The model will be experimentally validated, and its accuracy measured. The final conclusions will be based on the results of both thesis parts.

3 Literature Review

3.1 General Overview

Before we dive into the core of the practical part, Firstly, we hear a lot of nouns like: **AGI, AI, Machine Learning, Deep Learning** and, there's a massive hype around the Deep learning so, we need to understand what the meanings of those things are.
Secondly, the two words that make up the phrase "artificial intelligence" have been defined in a variety of ways over the years.
According to the dictionary, intelligence is:

- The ability to act
- The ability to sense
- The ability to understand
- The ability to solve problems

Additionally, the definition of "artificial" includes the following: made or modified with the aid of human skill and work.
The science and engineering of creating intelligent machines is what is meant by the word AI when it is combined, this definition is by John McCarthy. *(1)*

3.1.1 Artificial General Intelligence

AGI is an acronym for Artificial General Intelligence, in brief; For instance, we would like to create a system (e.g., Robot) that is very similar to human, its actions, behaviors, it can speak like human, hear, move and cook, etc.... *(1)*

intelligence capable of handling any intellectual activity with human-like efficiency. Making machines with their own capacity for cognition and intelligence comparable to humans is the core idea behind general artificial intelligence. Research and AI in the twenty-first century are currently at this stage, and moving beyond it will take more effort and advancement. *(1)*

The ability to mimic humans in every and each thing they do. Of course, this is an illusory fantasy, because so far, no one can create such a system like that or even close. *(1)*

Super AI is the level of system intelligence where machines are more intelligent than people and are capable of performing better than them in any task. The result of general artificial intelligence is this. Although it is only a theoretical concept at the moment, if it is realized, the world will alter. *(1)*

On that point, we're trying to solve a massive problem, so what we do -axiomatically- when we try to solve such a massive problem and we can't deal with it on a whole, we leave it aside and try to solve a simplified version of it, in another meaning; dividing it into a smaller problem and try to solve it first, if we failed? Simplify that smaller problem into another smaller one, so on and so forth until we can manage to solve it then go to the bigger one by one. Here we will face the **AI**. *(1)*

3.1.2 Artificial Intelligence

AI is an acronym for **Artificial Intelligence** (or **Artificial Narrow Intelligence**) and what is it? **AI** tells you to focus on just one task (or maybe two or three) something that is really a specific and it is not required to reach the level of humans as this is so optimal that we may (or may not) reach it in the future but our goal is to create something thathas a sort of specific intelligence in human. For an example; I give that system a text in English language and it translates that text into Czech language and it translates that text correctly or, I give it a picture and I ask it what's in the photo and it says for instance a human, I ask where is that human at and responds with specific location, so on and so forth. Just a specific task, but it has no ability to hear, talk, ask it about anything and it responds correctly to it.

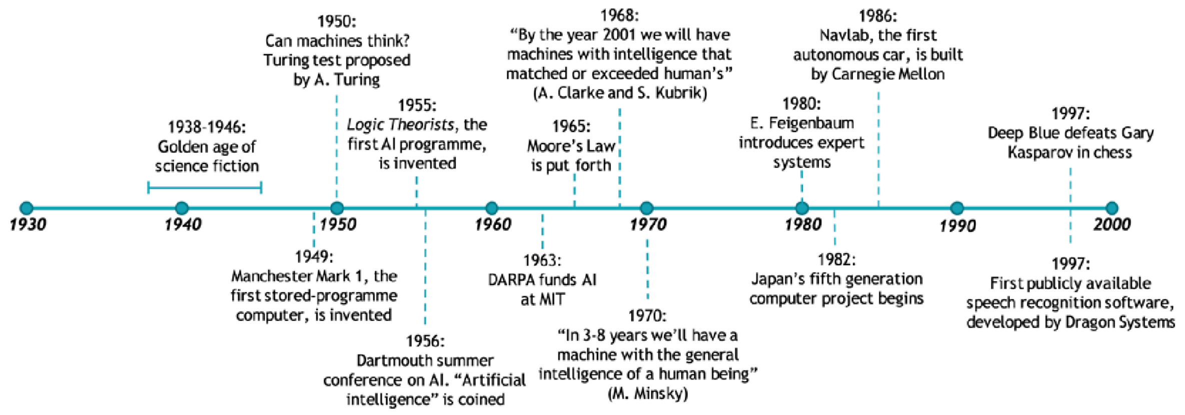

1930
1940
1950
1960
1970
1980
1990
2000
1938-1946: Golden age of science fiction
1950: Can machines think? Turing test proposed by A. Turing
1955: Logic Theorists, the first AI programme, is invented
1968: "By the year 2001 we will have machines with intelligence that matched or exceeded human's" (A. Clarke and S. Kubrik)
1965: Moore's Law is put forth
1986: Navlab, the first autonomous car, is built by Carnegie Mellon
1980: E. Feigenbaum introduces expert systems
1997: Deep Blue defeats Gary Kasparov in chess
1949: Manchester Mark 1, the first stored-programme computer, is invented
1956: Dartmouth summer conference on AI. "Artificial intelligence" is coined
1963: DARPA funds AI at MIT
1970: "In 3-8 years we'll have a machine with the general intelligence of a human being" (M. Minsky)
1982: Japan's fifth generation computer project begins
1997: First publicly available speech recognition software, developed by Dragon Systems

Figure 1 - Evolution of AI from the Year 1930 to the Year 2000, Source: (25)

3.2 AI Classifications

The revolutionary subsets of AI that are Machine Learning, Deep Learning & National Language Processing have been developed to more distinctive categories, based on two types; 1st type is based on the embedded level of intelligence and the 2nd is based on the functionality. *(10)*

Narrow AI (Based on the embedded level of intelligence): intelligently capable of carrying out a specific duty. It is the most typical and frequently employed type of AI right now. Beyond these limitations, it is only capable of performing the tasks for which it has been specifically trained. Examples include self-driving automobiles, Amazon Alexa, and chess playing bots. *(10)*

Reactive Machines: The earliest type of AI, reactive machines, have very limited capabilities. Individuals are memoryless and so unable to draw lessons from the past. They have only a few options for responding. The simplest forms of AI are those that are merely reactive machines. A reactive computer is an example of IBM's Deep Blue system. *(10)*

Limited Memory: Machines with limited memory have a limited amount of storage space. They can train, conduct reactive machine activities, and learn from previous data experiences. Large amounts of data can be processed using a variety of AI algorithms to produce new insights and improve accuracy. Autonomous vehicles employ a limited memory strategy. Most 21st-century AI machines employ a limited memory strategy. *(10)*

Theory of Mind: The next stage of AI is currently being researched, but little progress has been made. When engaging with beings, a theory of mind level AI will be better able to understand them by identifying their wants, emotions, beliefs, and mental processes. The "Theory of Mind," as it is known in psychology, will be "understood" by AI robots, which will be able to perceive people as unique beings whose brains can be changed by a variety of causes. *(10)*

Self-Awareness: : The ultimate goal of AI research has always been and will always be developing this kind of AI, which is decades, if not centuries, away from becoming a reality. This kind of AI will not only be able to comprehend and evoke the emotions of those with whom it interacts, but will also likely possess these same emotions, wants, beliefs, and even desires. Machines will be conscious and feel emotions, and they will be aware of themselves. Furthermore, this is the kind of AI that many who predict the end of technology are frightened of. This is due to the possibility that, once self-aware, AI could develop concepts like selfpreservation, which could either directly or indirectly result in the extinction of humanity. An entity can therefore cause disasters as well. *(10)*

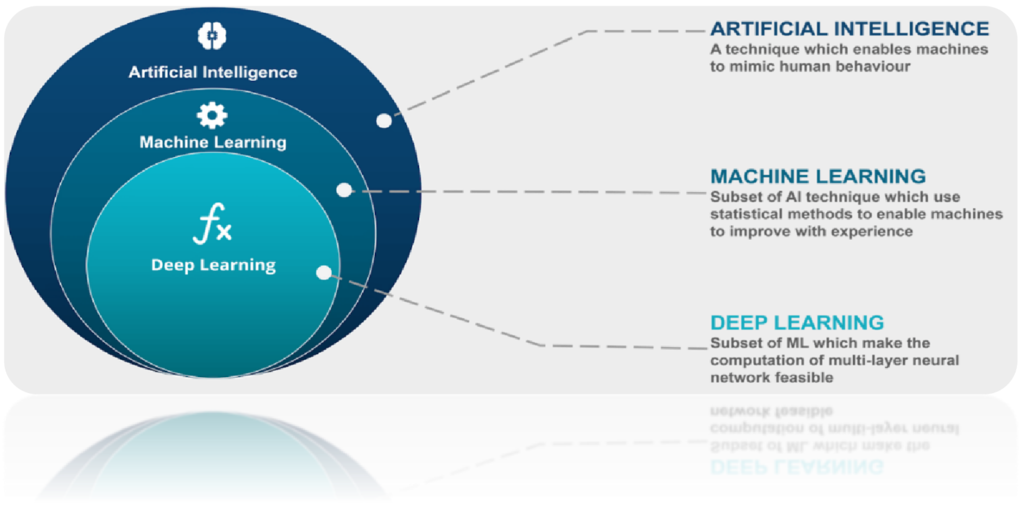

Artificial Intelligence
Machine Learning
Deep Learning
ARTIFICIAL INTELLIGENCE
A technique which enables machines to mimic human behaviour
MACHINE LEARNING
Subset of AI technique which use statistical methods to enable machines to improve with experience
DEEP LEARNING
Subset of ML which make the computation of multi-layer neural network feasible

Figure 2 – Subsets of Artificial General Intelligence, Source: (26)

So, what does AI do in brief? It creates a model per task so, for every task there's a model. So, **AI** focuses on specific tasks, modelling them independently. AI has history and **Machine Learning** is one of the most famous techniques in the history of AI. *(10)*

3.2.1 Machine Learning

Machine Learning is a part of AI that grants machines intelligence, giving them the capacity to naturally learn from their experiences without explicit programming. The main role is to learn from data, for example; I am given million number, and I have a query says "Are there two numbers(i,j) only that can be added to give a specific number(x)?", Someone can solve it by saying that we can traverse by two nested loops on these numbers and test if (i) plus (j) equals (x), but the problem of that resolution is that the numbers of operations that will happen is $million^2$ operation, which is a massive number. *(3)*

Data Structures and Algorithms comes up to that point and says how can we resolve these problems/queries in a better computational way. So, someone else can come up with an idea saying We can resolve it by using a *SET* to create an algorithm that will take $million \times \log million$ operation, someone else can say that he can use a hash table that takes million operations multiplied to a very small constant. This is a kind of problem that can be resolved without any Machine learning technique. But imagine if we had an emailing service (e.g. YAHOO), Many times we received a type of emails called spam emails these emails may someone sent to blackmail, hack your machine or exploit you so on and so forth, The company wants to create a feature that classifies emails as a spam or not, Someone can think about heuristics, for instance; I received an email from a sender that I have never dealt with then it may be a spam, or if the email mentions a massive number of money then it is a spam, the problem here is that these kind of nice heuristics won't always succeed. Let's have a more difficult problems, Imagine that someone provides 30 photo that every and each photo contains just a single kind of animal, Let's say that I have labeled classes for each one of the 30 animal, and I have to make a software that would figure out which type of class (animal) is in the photo that he is asked about, That problem is related with a photo so how many heuristics shall I make to make it functions well? Massive number of course. *(3)*

3.2.2 Example of classification model

Machine Learning is a field that comes up with an idea of having data (called Training Data) and labels for these training data, for instance; I will have 100 photos for the Cat class, 200 for the dog one, so on and so forth, to take these training data and teach itself those classes. And we can have a test set that provides photos with no labels and tests the software model created; asks that model about the kind of animal in the photo that it has never seen before to test the accuracy. The problem that has these kind of problems called "Classification model/problem" as it is classified depending each animal class (from the 30 class it got), sometimes the output is a float number or large number, for instance I give the model the properties of a car and it tells me how much would that car cost, at that point, these kind of problems called "Regression model/problem". If I have labels as our animal example above, it's called "supervised model/problem", an "unsupervised model/problem" is when I have no labels, for instance if I am given data of cars without their costs and I need to create a model which classifies those cars into cost categories, each item in the category is close to the other (for instance an Electric Tesla with an Electric Porsche).

3.2.3 Types & Application of predictive Machine Learning Models

Discriminative model is when I give the model a picture of an animal it would response with the most likely animal picture. **Generative model** learns the joint from each picture so it can be used to create/generate cases and examples from, for instance I give a model a picture and I ask it to remove a box from the picture but, this model has to remove the box and also to guess and substitute the background behind that removed box. *(11)*

Machine Learning contains a lot of models and **Neural Network model** is one of the most famous models in **Machine Learning.** *(11)*

Bear In mind that **Machine Learning** field itself, is just a field describes and speaks about the scientific algorithms that we can learn from data by them, for instance, when you read a book about Machine Learning (e.g.; Kevin Murphy's book) you will find a massive amount of mathematics. But what is the relation between the **Machine Learning** field and the areas like **Computer Vision** in **Medical Image Diagnose** for instance; it gives the machine the possibility to identify an object. Using one or more video cameras, analog-to-digital conversion, and digital signal processing, machine vision gathers and evaluates visual data. Machine vision systems are configured to carry out specifically defined tasks like reading the serial number, counting items, etc. Even while computer systems can see through walls and are unrestricted by human limits, they cannot perceive in the same way as human eyes. *(11)*

Natural Language Processing; makes it possible for a computer system to comprehend and analyze spoken or written English, among other human languages. For AI, NLP serves as a language tool incorporator. Human language may actually be utilized to program machines to learn and function thanks to NLP. These days, NLP and AI are everywhere. We can easily instruct Siri to carry out tasks in our language. *(11)*

Audio Recognition; a method by which a computer can convert spoken language into a form that is readable by machines. This method uses speech or voice commands as the input for a computer to carry out a certain activity. There are specific varieties of speech recognition software with a constrained word and phrase vocabulary. For this program to understand and carry out specified activities, clear spoken language is required. Today, a variety of programs or gadgets, including the Google Virtual Assistant, contain speech recognition technologies. These systems were initially only intended to translate speech to text, but today, A number of devices are available that can do just that. *(11)*

Deep Learning; a subset of AI and ML that enables a machine to carry out tasks normally performed by a person without the need for a human. It enables AI agents to behave like artificial neural networks that replicate the human brain. An AI agent can be trained using deep learning using both supervised and unsupervised learning. **stock prediction**; It's literally a discovering of the future worth of business stock and other financial assets traded on an exchange is made possible with the aid of stock price prediction utilizing machine learning. Gaining significant profits is the whole point of making stock price predictions. It's challenging to forecast how the stock market will fare. Other variables, including biological and psychological ones, as well as rational and irrational conduct, are included in the prediction. These forces work together to create a volatile and dynamic market for shares. Because of this, it is quite challenging to create precise stock price predictions. *(11)*

DN Analysis; It is a crucial tool for researching the rewiring of gene connections under various circumstances. Numerous computer techniques have been created to estimate differential networks from data on gene expression, but the majority of them do not take into account the possibility that gene network rewiring may be influenced by the differential expression of certain genes. It is necessary to develop new differential network analysis techniques that account for both changes in gene connections and changes in expression levels. *(11)*

Robotics; Which came because of artificial intelligence that has developed into a very fascinating subject. This fascinating area of creative endeavor mostly focuses on creating and developing robots. Mechanical engineering, electrical engineering, computer science, and many other fields of science and engineering are all incorporated within the interdisciplinary discipline of robotics. It plans the creation, handling, and application of robots. It oversees computer systems for administration, the generation of intelligent results, and data transformation. *(11)*

Expert System; That are computer programs that rely on information learned from experts in the field and are later incorporated into a system. Expert systems imitate human experts' capacity for decision-making. Instead of utilizing traditional procedural code, these systems are made to tackle complex issues using bodies of knowledge. The Google search box's recommendation for spelling problems is one instance of an expert system in action. Also, different applications of Machine learning in some areas also in finance, retail, insurance and the like… *(11)*

These are all applications for Machine **Learning**. Those domains are only using the Machine Learning and, each one has its own concepts and problems then we use machine learning as an application upon these things, for example; If I have an image and I want to know if there's a cat in it or not, what shall I do? The best thing is to say that this image is data, consider it as a class and then give it to our model to learn from it. *(11)*

3.3 Life Cylcle of an AI-Based Model

The ongoing procedure that data science projects adhere to is the AI life cycle. It outlines every action that a company should take to benefit from machine learning and artificial intelligence (AI) in order to provide useful economic value; a similar procedure is used for ML-based projects. Ai and ML are terms that are frequently used interchangeably. The AI life cycle has several significant steps, each of which is equally important and must proceed in a particular order. The planning phase, data phase, development phase, and deployment phase are further names for the key stages in artificial intelligence (AI). *(12)*

Identifying an opportunity to measurably enhance operations, boost customer pleasure, or otherwise create value is the first stage of the project life cycle. Data Acquisition and Exploration: Gathering and preparing all pertinent data for machine learning is the second phase. In order to do this, it is necessary to consult domain experts to determine which specific data might be pertinent in predicting the needed solution, collect that data from historical records, and then get it into a format that is suitable for analysis, most likely into a flat file format like a .csv, .arf, or .txt. Since data can be gathered from a variety of sources, including files, databases, the internet, and mobile devices, we must first identify the various data sources in this step. One of the most crucial phases of the life cycle is this. *(12)*

The effectiveness of the output will depend on the quantity and caliber of the data gathered. The prediction will be more precise the more data there is. Data Preparation: Data must be processed for use in the next steps after being collected. The process of setting up our data in a suitable location and qualifying it for use in machine learning training is known as data preparation. In this stage, we prepare all the data together before randomizing the order of the data. This method can be separated into two different steps: *(13)*

One purpose of **Data exploration** is to comprehend the type of data we must work with. We must assess the qualities, formats, and characteristics of the data. A more thorough analysis of the data results in a successful outcome. We find correlations, broad trends, and outliers in this.

Second purpose & Next step is Preparing the data for analysis: At this stage, the next step is to prepare the data for analysis (**Data preprocessing**).

Cleaning and transforming raw data into a format that can be used is referred to as data wrangling or data preparation. It is the procedure of cleaning up the data to resolve issues with quality, choosing the variable to use, and formatting the data appropriately to make it more acceptable for analysis in the following phase. It is among the most crucial steps in the whole procedure. Additionally, the user may not always find value in the data that has been collected; **Missing values, Duplicate records, Invalid records, and Noise** are just a few of the problems that collected data may encounter in real-world applications. *(14)*

As a result, we clean the data using a variety of filtering methods. The aforementioned problems must be found and fixed because they can compromise the outcome's quality. *(14)*

Model Data: A target variable must be selected in order to extract insights from the data using machine learning. This is the variable that the user hopes to better understand. This process involves choosing the right analytic methods, such as classification, regression, cluster analysis, association, and model development, and reviewing the outcomes.

There are two crucial steps in this for an AI model: The training phase is where we hone our model's abilities to produce better results for the challenge. To train the model with different machine learning techniques, we use datasets. *(14)*

A model must be trained in order for it to comprehend the different patterns, laws, and features. In the testing step, we use a test dataset that is parallel to our model to determine its accuracy. According to the needs of the project or challenge, testing the model determines its accuracy %. Interpret and communicate: Explaining a model's results to people without any training in data science is one of the most challenging challenges of machine learning initiatives, especially in highly regulated industries like healthcare. As a result of how challenging it is to understand findings and explain the value of those insights to stakeholders and regulatory agencies, machine learning has traditionally been viewed as a "black box." It will be simpler to satisfy regulatory standards and explain a model's worth to management and other important stakeholders the more interpretative it is. The implementation, documentation, and maintenance of the data science project are the final steps to ensure that the stakeholder can keep using and improving upon its models. Because it involves a lot of coding and data science expertise and takes so long to implement from the start of the cycle when traditional data science approaches are used, model deployment frequently presents a challenge. *(14)*

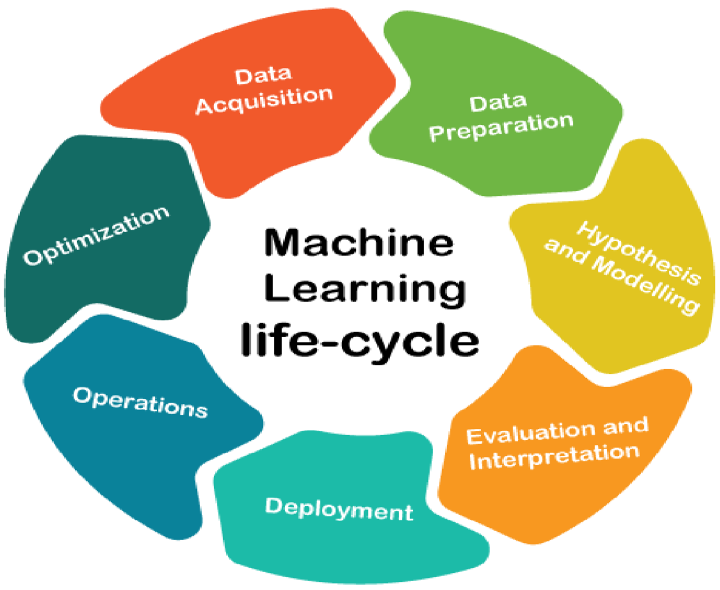


Figure 3 – Machine Learning Life Cycle. Source: (27)

3.4 Machine Learning Prerequisites

First things first, **Programming**; As a Machine Learning Engineer, you would end up implementing projects, Hence, Every ML Engineer must have those project's skills. Also, there's a cycle in it where you do pre-processing & post-processing for data, Object Oriented skills are also important for writing a well-constructed code instead of functional approach which lead to more mess in the code. *(15)*

Second things second, **Algorithms**; There's a strong bond between it and Machine Learning. Firstly, the complexity, it's very important to consider the algorithm complexity while creating a machine learning model to know which is faster from the other hand, which is more efficient whether in training time or inference time, etc.… *(15)*

Algorithms contain a lot of topics; do we need all of them? Yes, but in a very small picture. Graph theory for instance; If we're making papistic modelling then we'll need to make a graph representation for it, Also, in Computer Vision, the image that comes out is a graph. Sometimes we may also need the Depth first search or BFS. In Object Tracking & Hand Pose Estimation, we will need Hungarian Algorithms (It's a kind of algorithm that may be written in main cos maxflow). Also, sometimes we may need Algorithms in terms of DP, Greedy, Optimization and the like. But as we discussed before, we will need them in a very small picture, not always redundant, you may read. *(15)*

In a brief; For the purpose of pursuing an ML/AI project, the following require competent expertise or knowledge: *(16)*

i. Basic computer literacy

ii. Linear algebra

iii. Probability and statistics

iv. Calculus

v. Theoretical Graphs

vi. Knowledge with programming languages like Python, R, MATLAB, C++, or Octave

vii. Data and Hardware

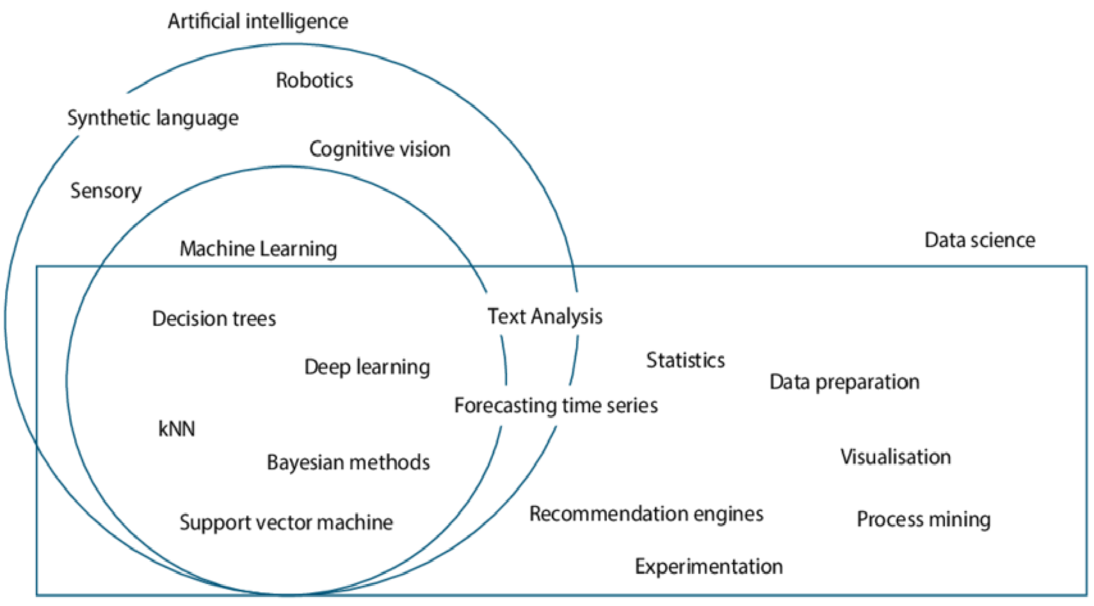


Figure 4 – AI, ML and Data Science Techniques, Source: (28)

Machine learning was introduced as a method exclusive to artificial intelligence in the 1950s, but it has since shifted its focus to computationally feasible algorithms. The science and technology of machine learning concentrate on and incorporate all of the interconnected steps required to enable a machine to learn from experiences and instructions in order to enhance performance. There are several ideas that could serve as inspiration for some biological learning processes. *(17)*

In the case of machines, it can be said that a machine "learns" when it is designed with the ability to adapt its structure to improve the results in the future. The potential for learning arises from the inputs provided to a machine or in response to outside information, and as a result, the model's experience increases its level of accuracy. The three main parts of the machine learning process that fall under the purview of this thesis are input, machine learning models, and output. The data or information sent to the machine is referred to as the input. In the instance of an agricultural application, this can be sensor data. *(17)*

The machine learning model is a model that can process extra data to produce predictions after being trained on a set of training data. Since there are numerous models and techniques used in machine learning systems, the one that performs the task most effectively in terms of convenience and efficiency is selected as the major component. *(17)*

Machine learning has advanced significantly thanks to the quick rise in processing power and effective management of big data utilizing a variety of algorithms, tools, and methodologies. ML has become more popular over the course of the industrial revolution, with a wide variety of applications appearing in practically every industry. By using algorithmic learning to train on the data and then being able to work on similar data trends, machine learning (ML) duplicates the human learning process. Although ML is significantly less intelligent than humans are, depending on the task at hand, there have been situations where ML has come out on top. In contrast to traditional approaches, such as Excel, which are typically unable to manage large amounts of data, machine learning (ML) is built with the intention of becoming more accurate as more data and information are fed into the algorithms. *(17)*

3.5 Machine Learning in Wireless Sensor Network & Internet of Things

Machine learning models in agriculture are built on IOT and wireless sensor networks. A learning model in the context of WSN can be a straightforward parametric function that gains knowledge from data and a few input variables in order to capitalize on outcomes. Smart, heterogeneous, affordable, and energy-efficient sensor nodes that make up a WSN can gather environmental data and send it to a centralized base station or sink node for processing. *(1)*

By using the same old data to create networks that are more efficient and can act as forecasting models, machine learning has proven to be a fantastic choice in tackling the problems of WSN with the advancement in technology. There are many reasons why machine learning should be used in wireless sensor networks, including:

1) WSN difficulties in a dynamic environment are handled by ML, which aids in node optimization for improved adaptability.
2) ML offers effective computational options for challenging contexts.
3) Precision farming has increased as a result of the inclusion of ML in WSN.
4) IOT technology has significantly advanced

Machine learning (ML) is a key subfield of artificial intelligence since it works on algorithms built with the intention of acquiring a self-learning capability. "ML Algorithms" are distinct from "Conventional Computer Algorithms," which operate solely in accordance with the software that was developed by its developer. In contrast to traditional statistical methods, "ML Algorithms" understand and analyze both the input and the output (results), hence a benefit of these methods is that the model is less dependent on user instructions.

Here, the four primary categories of machine learning techniques are illustrated together with the main forms of machine learning. *(2)*

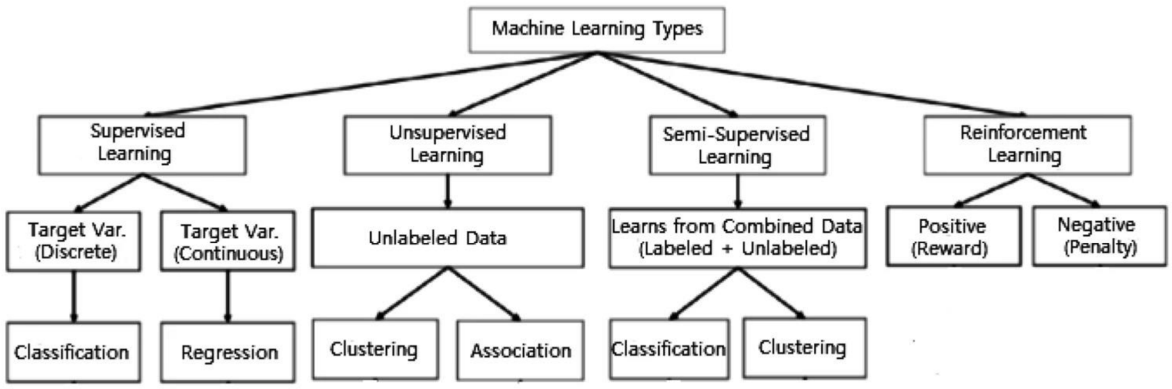


Figure 5 – Machine Learning Types, Source: (29)

3.6 Supervised Learning

For an algorithm to read and analyze inputs and their related outputs, training samples must be given. This is a sort of machine learning. Using labeled training datasets, supervised machine learning algorithms produce an inferred function from the relationship between the system's output, input, and parameters.

After receiving adequate training, this kind of model is prepared and trained to handle incoming input. Additionally, there is a clause that allows for model modification after comparing the obtained results to the desired/correct outcomes. When using supervised learning, the following target values are predetermined: *(1)*

1) Regression becomes problematic if a "continuous target variable" is present.
2) If a "categorical target variable" is present, classification becomes problematic

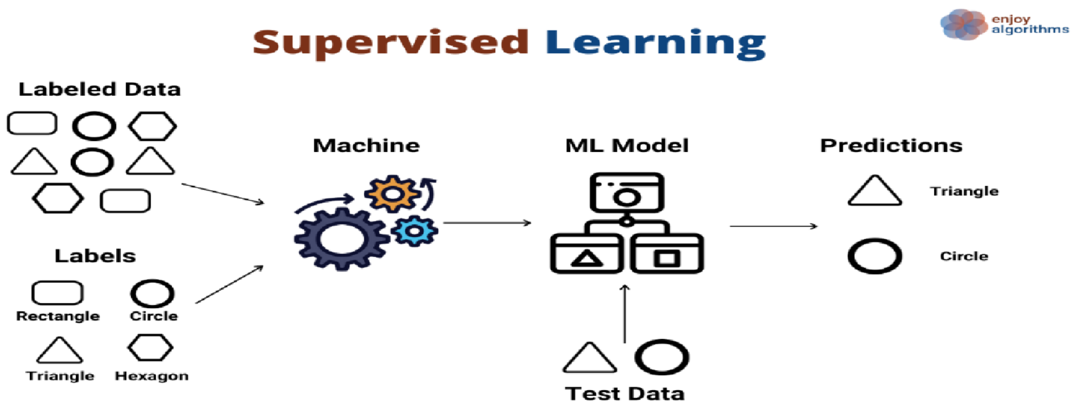


Figure 6 – Supervised Machine Learning, Source: (30)

3.6.1 Decision Trees

When using decision trees, a set of predetermined characteristics is available, and the data points of the datasets are assigned in accordance with these characteristics, creating a predictive model. Decision trees operate more simply by repeating the learning process used to produce the output in order to forecast the output labels. *(3)*

3.6.2 Support Vector Machines (SVM)

By creating set hyper-planes in a feature space that sharply divides the data using its techniques, support vector machines are very helpful in identifying the space-time connection in the datasets. When it comes to tackling non-convex unconstrained optimization issues, SVMs are recommended. *(3)*

3.6.3 Neural Networks

One of the most popular machine learning (ML) approaches today, neural networks are ostensibly inspired by a biological brain network. Three layers make up the basic neural network: the input layer, the hidden layer, and the output layer. The model consists of many "neurons" (also known as nodes) and "edges" (also known as connections between neurons). The neural network model typically involves intensive computations. *(3)*

3.6.4 K-Nearest Neighour (K-NN)

The k-NN algorithm is straightforward, making it the most often used approach for supervised learning. In this method, test sample data is categorized into the user-specified k value based on the labels of nearby data samples using the "minimum-distance classification method" (k is the number of neighbors to consider). *(3)*

3.6.5 Bayesian Learners

Comparatively speaking, the method needs fewer training samples than other ML algorithms. Bayesian approaches employ a probability distribution to make learning uncertain labels easier. Different kinds of Bayesian learners, such as Dynamic Bayesian Networks, Gaussian Mixture Models, Conditional Random Fields, Hidden Markov Models, and others, aid the model in learning the relationships more effectively. Bayesian networks are a form of probabilistic visual model that computes probabilities using Bayesian inference. *(3)*

3.7 Unsupervised Learning

There is no data set to which the machine learning algorithm may refer because it is not given any goal variables. Since the input datasets and output vectors are not labeled, the ML model looks for similarities between the input datasets. The hidden patterns and correlations among the variables are found by the unsupervised machine learning method.

This subset of ML includes algorithms that use clustering and association techniques. There are two categories of algorithms used in unsupervised machine learning:

1) Clustering: With this approach, a data set is divided or distributed into several, relatively small groups so that the data sets in each group or cluster are more comparable to one another than to those in other groups. Data points that share more characteristics are clustered together to form clusters, and numerous clusters are created.

2) Association: In huge databases, this kind of algorithm focuses on identifying specific correlations between variables

In other words, we may say that association is based on the correlations between the likeness of data points and finding patterns among their qualities, whereas clustering is based on the grouping of the data points according to the more common features or similarities that they contain. *(4)*

Principal component analysis, k-means clustering, dimensional reduction, neural networks and deep learning, singular value decomposition, independent component analysis, and distribution models are a few examples of unsupervised machine learning algorithms. hierarchical clustering, etc.

Additionally, the most crucial procedures are principal component analysis and k-mean clustering. *(4)*

3.7.1 Principal Component Analysis

This kind of analysis uses an algorithm for unsupervised learning that only extracts the most crucial data from the input, known as the principal components. Principal components are a brand-new set of orthogonal variables. Principle component analysis is appropriate for dimensional reduction and data compression. Big data typically streamlines the procedure by removing other lower-order, unimportant components from the model and choosing just significant principal components. *(7)*

3.7.2 K-Means Clustering

This is an additional technique for grouping unlabelled data into categories. The model's machine learning method depends on clustering, which is beneficial for identifying trends in a data set with comparably similar data points. The data points in the data set are accumulated with the aid of "minimum-distance classification" based on similarities, and the center point of the clusters is calculated in this process. In this instance, "means" refers to the centroid while "k" denotes the number of centroids. *(7)*

3.8 Semi-Supervised Learning

As both labelled and unlabelled data are utilized to train the ML model, semi-supervised learning is seen as a blend of supervised and unsupervised learning. Usually, the training set's unlabelled data is a lot bigger than its labelled data. The data structure, amount of data to be handled, and application scenarios are the major factors in determining whether to use supervised or unsupervised machine learning methods. Many machine learning researchers claim that using semi-supervised methods can significantly increase a model's learning accuracy. *(7)*

3.9 Reinforcment Learning

In reinforcement learning, the algorithm receives feedback for each action, such as a reward or an error (referred to as the reinforcement signal), and this cumulative feedback reinforces the system's effectiveness, maximizing benefits over time. Dynamic programming approaches are used in many reinforcement learning systems. Popular algorithms include Markov decision algorithms, Q-learning, and genetic algorithms. *(7)*

3.10 Deep Learning & Artificial Neural Networks

ML is a subset of AI, one of the oldest approaches in the history of AI, as we have already described. When terms like artificial intelligence, machine learning, artificial neural networks, and deep learning are used together, a common misconception result. *(7)*

Put it simply; Deep learning is a subset of machine learning, which is a discipline devoted to the investigation and creation of intelligent machines (sometimes with the goal of eventually attaining general artificial intelligence). *(7)*

Deep learning is used in the IT sector to resolve real-world problems in a range of categories (fields), including picture (computer vision), text (NLP), and audio (speech recognition). Briefly said, deep learning is a subset of machine learning techniques that predominantly employ **Artificial Neural Networks**, a group of algorithms that are loosely based on the functioning of the human brain. Initially, the goal of an ANN was to approach issues in a similar manner to a human brain. The ANN, which attempts to imitate the human brain in carrying out complex operations such pattern production, cognition, learning, and decision making, is one of the extensively used techniques in machine learning. *(7) (6)*

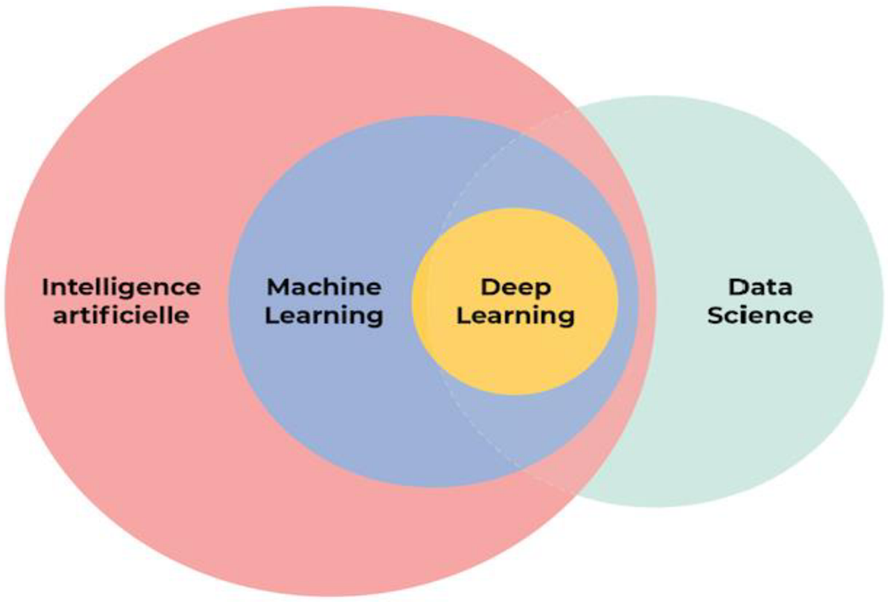


Figure 7 – Deep Learning in a nutshell, Source: (31)

3.11 Some Machine Learning's Applications

ANN is frequently chosen in situations where data points have nonlinear correlations since it aids in the development of various relationships between clusters of information. Artificial neural networks can be broadly divided into two categories: traditional ANNs and deep ANNs. *(7)*

Traditional ANNs: These are supervised machine learning models that are commonly utilized for regression and classification issues. The "Artificial Neurons" or nodes/units that make up the ANN model are connected to one another to form a network, and the ties that bind them together are referred to as edges. As the learning process progresses, weights on both nodes and edges can boost or weaken the signal of a connection. The nodes are arranged in three levels in a straightforward ANN model: the input layer, one or more hidden layers (learning), and the output layer (result)... *(7)*

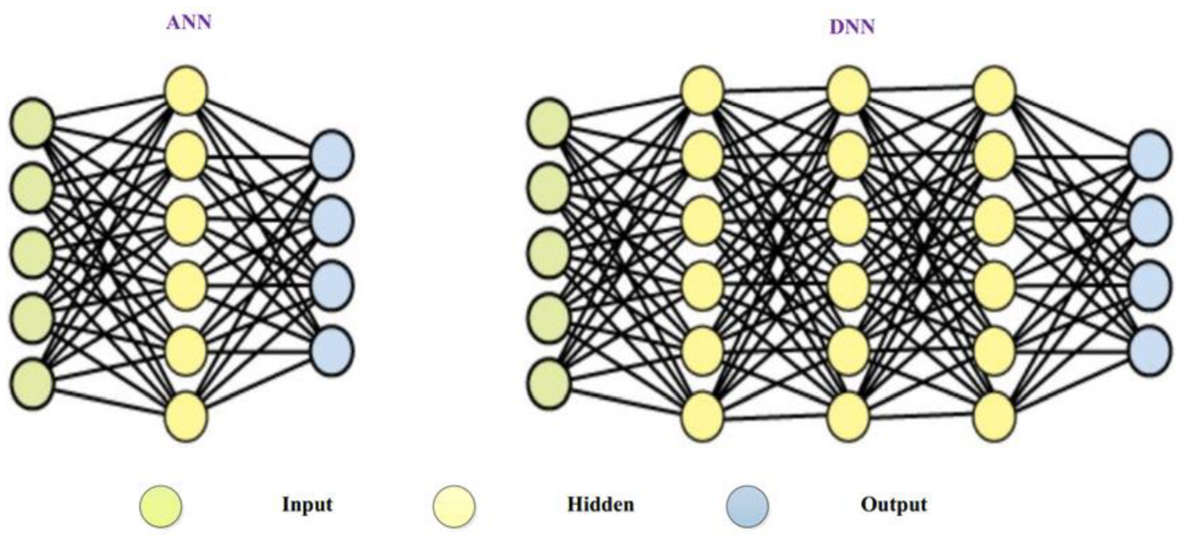


Figure 8 – Neural Networks, Source: (32)

The circles in the above graphic represent the nodes (artificial neurons), and the lines connecting the artificial nodes stand in for the edges in both the simple artificial neural network and the deep neural network. *(8)*

Deep ANNs: These are sometimes referred to as deep learning or deep neural networks; the "deep" part of deep learning refers to the network's utilization of multiple layers. DNNs are essentially ANNs with numerous hidden layers, but the learning method can be supervised, partially supervised, or even unsupervised. These many hidden layers are used to execute multiple degrees of data abstraction, and one of the major characteristics is that it occasionally completes the process of feature extraction by itself. These models have produced outcomes that have come near to or even surpassed the performance of human experts in a variety of fields. DL models have also significantly enhanced agriculture.*(7) (18)*

3.12 Some Machine Learning Applications

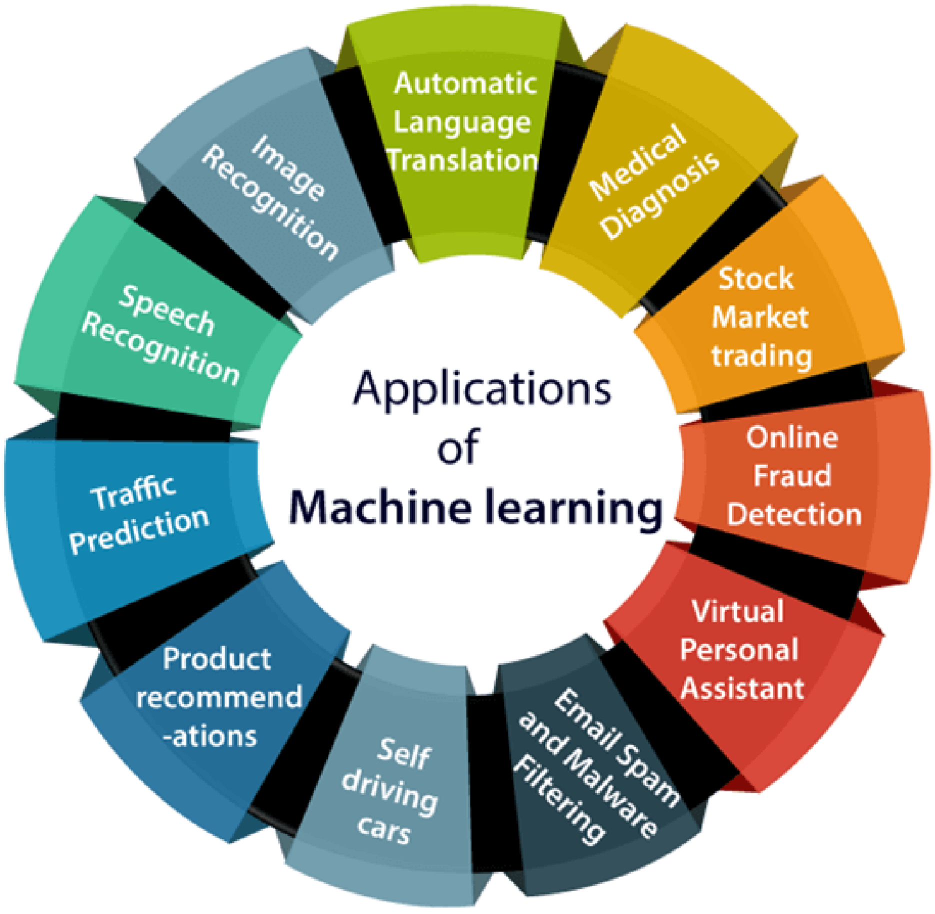

Applications of Machine learning
Automatic Language Translation
Medical Diagnosis
Stock Market trading
Online Fraud Detection
Virtual Personal Assistant
Email Spam and Malware Filtering
Self driving cars
Product recommend -ations
Traffic Prediction
Speech Recognition
Image Recognition

Figure 9 - Applications of Machine Learning. Source: (33)

3.13 Data Cleaning in Machine Learning

First, we need to know How is data cleaning defined? Data cleaning is the procedure of removing inaccurate or irrelevant information from the data before analysis. This type of information typically reinforces a false belief, which might have a negative effect on the model or algorithm it is given into. In addition to eliminating large portions of irrelevant data, data cleaning also frequently refers to correcting inaccurate data in the train-validation-test dataset and minimizing duplicates. *(19)*

Before conducting any type of analysis on data, data cleaning is a crucial step.

In pipelines, datasets are frequently gathered in small groups, combined, and then fed into a model. When several datasets are combined, duplicates and redundancies are created in the data that must be deleted. *(20)*

Moreover, models can frequently develop inaccurate representations of the data from incorrect and poorly collected datasets, which weakens their ability to make decisions. It is not at all perfect. Yet, when using unclean data directly, the loss in model accuracy is actually the least problematic issue that can arise. If the noise is uniform across the training and testing sets, models trained on raw datasets are compelled to treat it as information and can make accurate predictions. However, when new, cleaner data is presented to the model, it fails. Hence, data cleansing is a crucial component of any machine learning pipeline and should not be disregarded. But how should data be cleaned for machine learning? As evidenced by research; The element of data science that is typically the least pleasurable and takes the longest is data cleaning. *(20)*

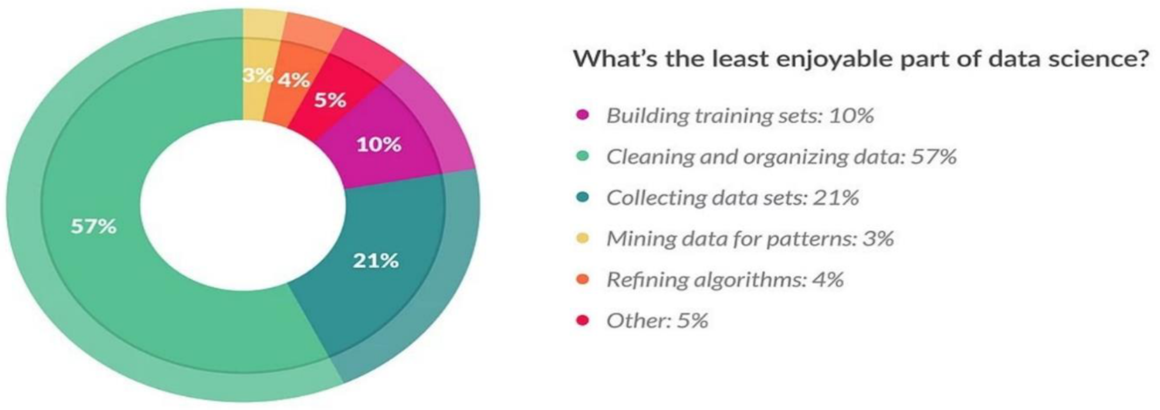


Figure 10 – Least enjoyable part of data science, Source: (34)

Cleaning data means manually going through a lot of data in order to:

1) weed out irrelevant information, which is a laborious operation.
2) Determine whether a column has to be dropped or not.

Automation of the cleaning procedure typically calls for a depth of knowledge in handling corrupt data. It's a little difficult to accomplish without causing data loss. Though we're going to work on only 4 types of errors; *(19)*

1. Remove duplicate
2. Remove Irrelevant data
3. Fix syntax errors (typos)
4. Handle missing data

Duplicates across columns and rows must frequently be filtered away in data that is treated as data frames. Duplicates can result from a participant completing a survey more than once or from the survey having numerous fields on the same subject, which causes many participants to provide the same response. The former needs research and the use of algorithms, whilst the latter is simple to remove. Moreover, columns in a data frame may contain information that is utterly unrelated to the work at hand; as a result, these columns are eliminated before the data is processed further. *(19)*

Because such a large population is represented, data gathered through a survey frequently contains syntactic and grammatical errors. Simple syntax errors like date, birthday, and age can be easily fixed, however errors involving spelling take more time to repair. To remove typos, grammatical and spelling errors, and other inaccuracies from the data, algorithms and procedures that discover and correct these issues must be used. In contrast, syntax errors can be completely avoided by regulating the format in which data is collected, followed by checks to make sure that participants haven't entered known fields incorrectly. Establishing rigorous parameters for variables like State, Nation, and School greatly increases the likelihood of receiving high-quality data. Before data can be processed further, unwanted data in the form of outliers must be eliminated. *(19)*

Of all the information's errors, outliers are the most difficult to find. A data point or set of data points must often undergo extensive investigation before being ruled out as an outlier. A large number of outliers can easily influence certain models that have a very low outlier tolerance, lowering the accuracy of the predictions. *(19)*

Sadly, incomplete data gathering processes always result in missing data. It must be found and addressed as quickly as feasible. While it is simple to see these artifacts, it is frequently necessary to take caution into account while filling in blank sections because random fills might have unanticipated effects on the model's quality. Rows with missing data are frequently omitted since it would be inconvenient to precisely fill in every single data point. The entire column is discarded when several data points for the same attributes have missing data. Data scientists are forced to make educated estimates in the face of incomplete data and under absolutely unavoidable conditions. These calculations frequently call for the observation of two or more data points comparable to the one being examined and the filling in of the empty regions with an average value from these points. *(20)*

4 Practical Part

4.1 Problem definition & Steps to take

As we discussed before; The first step in building a machine learning model specialized for cleaning datasets is to define the specific data cleaning task that the model should perform. This could involve identifying one or more of the following:

- Missing data: Data that is missing or incomplete and needs to be imputed or removed.
- Irrelevant data: Data that is irrelevant or redundant and needs to be removed.
- Inconsistent data: Data that is inconsistent or incompatible with the rest of the dataset and needs to be transformed or removed.
- Duplicate data: Data that is duplicated or nearly identical and needs to be removed.
- Outliers: Data that is significantly different from the rest of the dataset and needs to be identified and removed or transformed.

Once you have identified the specific data cleaning task, you can begin to collect a representative dataset that includes examples of the types of errors, inconsistencies, or other issues that the model will need to correct. You should ensure that the dataset is representative of the data that the model will be used to clean. This means that it should include examples of the types of errors, inconsistencies, and other issues that the model will be expected to correct. By defining the problem and collecting a representative dataset, you can ensure that your machine learning model is trained to effectively clean datasets that are similar to the ones it will be used on in production. The type and extent of data you need to collect for your data cleaning machine learning pipeline will depend on your specific problem and the requirements of your application. However, here are some general considerations and constraints to keep in mind:

1. Relevant data: Collect data that is relevant to your problem and the task you want to perform. This could be data from various sources, such as databases, files, or APIs. The data you collect should be relevant to the problem you are trying to solve. This means that you should consider what type of data will be most useful for cleaning the dataset. For example, if you are cleaning a customer database, you might want to collect data about the customer's name, address, phone number, and email address.

You may also want to collect demographic data such as age, gender, and occupation if this information is available and relevant to your cleaning task.

2. Data quality: Ensure that the collected data is of high quality and is consistent. You should check for errors, missing values, and outliers in the data. In addition, you should consider the completeness, accuracy, and consistency of the data. Data quality is crucial in a data cleaning pipeline. You should ensure that the data you collect is accurate, complete, consistent, and free from errors. This may require cleaning the data prior to collecting it, or using techniques such as outlier detection or imputation to correct errors during the cleaning process. It's important to carefully consider the potential sources of errors in the data and take steps to address these issues, such as using data validation checks to detect missing values or inconsistent data entries.
3. Data privacy: Be aware of any privacy concerns associated with the collected data. Ensure that you have the necessary permissions and consents from the data owners to use the data for your purpose. The amount of data you need to collect will depend on the complexity of the problem and the machine learning algorithms you plan to use. Collect a sufficient amount of data to train and test your models effectively. For some problems, a small dataset may be sufficient, while others may require large amounts of data to achieve good results. You should also consider the distribution of the data and ensure that your dataset is representative of the population you are trying to model.
4. Data size: The size of your dataset will depend on the complexity of your problem and the machine learning algorithms you plan to use. Collect a sufficient amount of data to train and test your models effectively. The data you collect should be in a format that is suitable for your machine learning algorithms. This could include structured or unstructured data, and could be in various formats such as CSV, JSON, or XML. You may also need to consider the encoding of the data, especially if the dataset includes non-English characters or other special characters. In some cases, you may need to convert the data to a different format or use specialized tools to extract features from the data.
5. Data format: Ensure that the collected data is in a format that is suitable for your machine learning algorithms. This could include structured or unstructured data, and could be in various formats such as CSV, JSON, or XML. Be aware of any privacy concerns associated with the collected data. Ensure that you have the necessary permissions and consents from the data owners to use the data for your purpose.

Additionally, you may need to take steps to anonymize or de-identify the data to protect privacy. For example, you might remove personally identifiable information from the dataset before analyzing it.

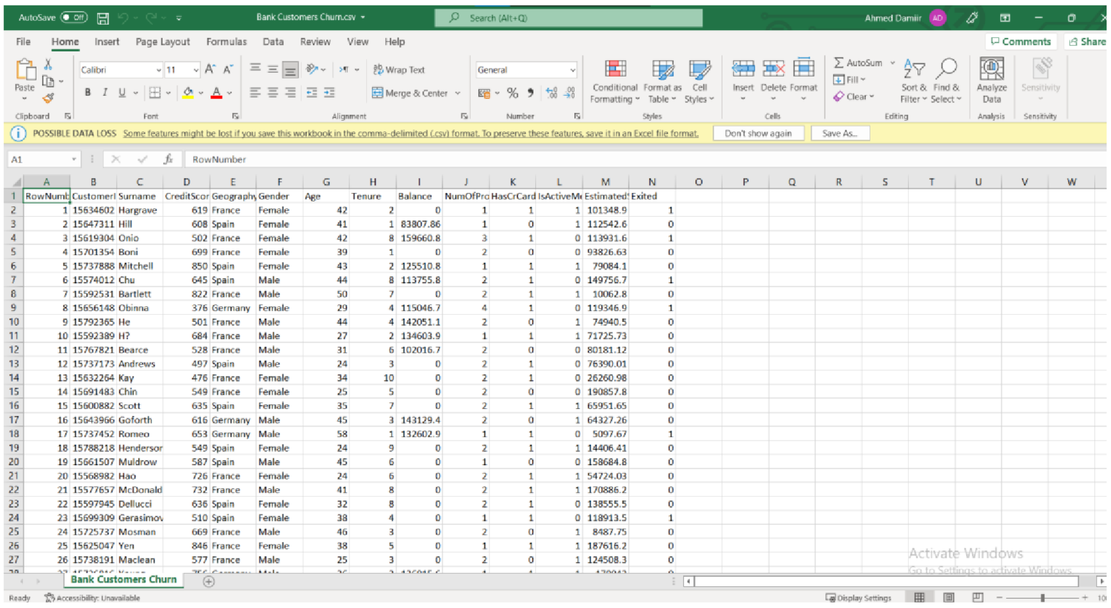
6. Data storage: Decide on the most appropriate way to store the collected data. This could include storing the data in a database, cloud storage, or on-premises storage. Decide on the most appropriate way to store the collected data. This could include storing the data in a database, cloud storage, or on-premises storage. Ensure that the storage solution you choose is secure and meets any regulatory requirements. You may also need to consider data retention policies and archiving requirements, especially if the data is subject to legal or regulatory requirements. Additionally, you should consider how you will manage and maintain the dataset over time, especially if the data is dynamic and subject to change

In summary, the data you collect for your data cleaning machine learning pipeline should be relevant, high-quality, and consistent. Additionally, you should consider any privacy concerns associated with the data and ensure that it is in a suitable format and stored appropriately. The data you collect may become useless if it does not meet your criteria for data quality, relevance, or format. If the data is inaccurate, incomplete, inconsistent, or contains too many errors, it may not be useful for training or testing your machine learning models. Additionally, if the data is not relevant to your problem or does not provide meaningful insights, it may not be worth the effort to clean and analyze it. Finally, if the data is not in a format that is compatible with your machine learning algorithms or tools, it may be difficult or impossible to use effectively. In these cases, it may be necessary to disregard the data and look for alternative sources or strategies for data collection.

4.1.1 Pre-process the test dataset after collection

When it has been collected, the test dataset needs to be pre-processed so that it can be used to evaluate the performance of the machine learning model. This often entails using the same pre-processing techniques, such as cleaning, feature engineering, and data transformation, that were used on the training dataset.

A dataset of examples representing the data cleaning task you want the model to carry out is required to train a machine learning model for cleaning datasets. Make sure the dataset is a good representation of the data that the model will clean. This means that it needs to include specific instances of the kinds of mistakes, contradictions, and other problems that the model will be required to fix. For that specific part I used **Kaggle.com** (A website that provides dozens of datasets for the sake of data analysis processes and the like). The data set I picked is Bank **Customers Churn** in csv format.



POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	RowNumb	Customerl	Surname	CreditScor	Geography	Gender	Age	Tenure	Balance	NumOfPro	HasCrCard	IsActiveMe	Estimated!	Exited
2	1	15634602	Hargrave	619	France	Female	42	2	0	1	1	1	101348.9	1
3	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.6	0
4	3	15619304	Onio	502	France	Female	42	8	159660.8	3	1	0	113931.6	1
5	4	15701354	Boni	699	France	Female	39	1	0	2	0	0	93826.63	0
6	5	15737888	Mitchell	850	Spain	Female	43	2	125510.8	1	1	1	79084.1	0
7	6	15574012	Chu	645	Spain	Male	44	8	113755.8	2	1	0	149756.7	1
8	7	15592531	Bartlett	822	France	Male	50	7	0	2	1	1	10062.8	0
9	8	15656148	Obinna	376	Germany	Female	29	4	115046.7	4	1	0	119346.9	1
10	9	15792365	He	501	France	Male	44	4	142051.1	2	0	1	74940.5	0
11	10	15592389	H?	684	France	Male	27	2	134603.9	1	1	1	71725.73	0
12	11	15767821	Bearce	528	France	Male	31	6	102016.7	2	0	0	80181.12	0
13	12	15737173	Andrews	497	Spain	Male	24	3	0	2	1	0	76390.01	0
14	13	15632264	Kay	476	France	Female	34	10	0	2	1	0	26260.98	0
15	14	15691483	Chin	549	France	Female	25	5	0	2	0	0	190857.8	0
16	15	15600882	Scott	635	Spain	Female	35	7	0	2	1	1	65951.65	0
17	16	15643966	Goforth	616	Germany	Male	45	3	143129.4	2	0	1	64327.26	0
18	17	15737452	Romeo	653	Germany	Male	58	1	132602.9	1	1	0	5097.67	1
19	18	15788218	Hendersor	549	Spain	Female	24	9	0	2	1	1	14406.41	0
20	19	15661507	Muldrow	587	Spain	Male	45	6	0	1	0	0	158684.8	0
21	20	15568982	Hao	726	France	Female	24	6	0	2	1	1	54724.03	0
22	21	15577657	McDonald	732	France	Male	41	8	0	2	1	1	170886.2	0
23	22	15597945	Dellucci	636	Spain	Female	32	8	0	2	1	0	138555.5	0
24	23	15699309	Gerasimov	510	Spain	Female	38	4	0	1	1	0	118913.5	1
25	24	15725737	Mosman	669	France	Male	46	3	0	2	0	1	8487.75	0
26	25	15625047	Yen	846	France	Female	38	5	0	1	1	1	187616.2	0
27	26	15738191	Maclean	577	France	Male	25	3	0	2	0	1	124508.3	0

Figure 11 - Table used for model training, Bank Customers Churn, Source: (35)

Data need ensure; Ensure that the data is in a clean and usable format, and preprocess it as necessary to prepare it for training. This may involve several sub-steps, such as:

- **Data cleaning**: Remove any obviously incorrect or irrelevant data from the dataset, such as duplicates, empty values, or data that is clearly outside the expected range. Cleaning the test dataset involves detecting and handling missing or erroneous data. This can be done using techniques such as imputation, outlier detection, or data validation checks. Imputation involves filling in missing values with estimated values based on the available data. Outlier detection involves identifying and removing data points that are significantly different from the rest of the dataset. Data validation checks involve verifying that the data in the test dataset is within the expected range and format.
- **Data normalization**: Normalize the data so that it falls within a consistent range of values. This can help to ensure that the model is able to learn effectively from the data and can improve its performance on the cleaning task. Data transformation involves transforming the data in a way that is consistent with the training dataset. This can include normalizing or standardizing the data, or applying other types of transformations such as logarithmic or exponential transformations. The purpose of

data transformation is to ensure that the test dataset has similar statistical properties as the training

- **Feature engineering**: Extract meaningful features from the data that can help the model to learn more effectively. This may involve combining different variables or creating new variables that capture important information in the data. Feature engineering is the process of creating new features from the existing data that may be more informative for the machine learning algorithm. This can include techniques such as feature scaling, one-hot encoding, or dimensionality reduction. Feature scaling involves scaling the features so that they have similar ranges, while one-hot encoding converts categorical variables into numerical features that can be used by the model. Dimensionality reduction involves reducing the number of features in the dataset while retaining the most important information.
- **Data splitting**: Split the dataset into training, validation, and testing sets. The training set is used to train the model, the validation set is used to monitor the model's performance during training, and the testing set is used to evaluate the model's performance on new, unseen data. Splitting the data involves dividing the dataset into two or more subsets. Typically, the data is split into a training set and a test set. The training set is used to train the machine learning model, while the test set is used to evaluate the performance of the model. The data may also be split into a validation set, which is used to fine-tune the model parameters and prevent overfitting.
- **Scaling**: Scaling involves scaling the features in the test dataset so that they have similar ranges or units as those in the training dataset. This is important because some machine learning algorithms, such as neural networks or support vector machines, are sensitive to the scale of the input features. If the scales of the features are different between the training and test datasets, the model may not perform well on the test dataset.
- **Handling missing values**: If the test dataset contains missing values, they may need to be imputed or filled in using a similar approach as the training dataset. There are several techniques for imputing missing values, such as mean imputation, median imputation, or regression imputation. It is important to ensure that the imputation technique used on the test dataset is consistent with that used on the training dataset, to ensure that the model is evaluated fairly and accurately.

By gathering and preprocessing the data, you can ensure that the dataset is clean, usable, and properly formatted for training the machine learning model. This can help to improve the accuracy and effectiveness of the model on the cleaning task. Any machine learning project must start with gathering and preparing the data.

The performance of a machine learning model is significantly influenced by the quality of the data used to train and test the model. Consequently, it is crucial to make sure the data is accurate, pertinent, and suitable for the model.

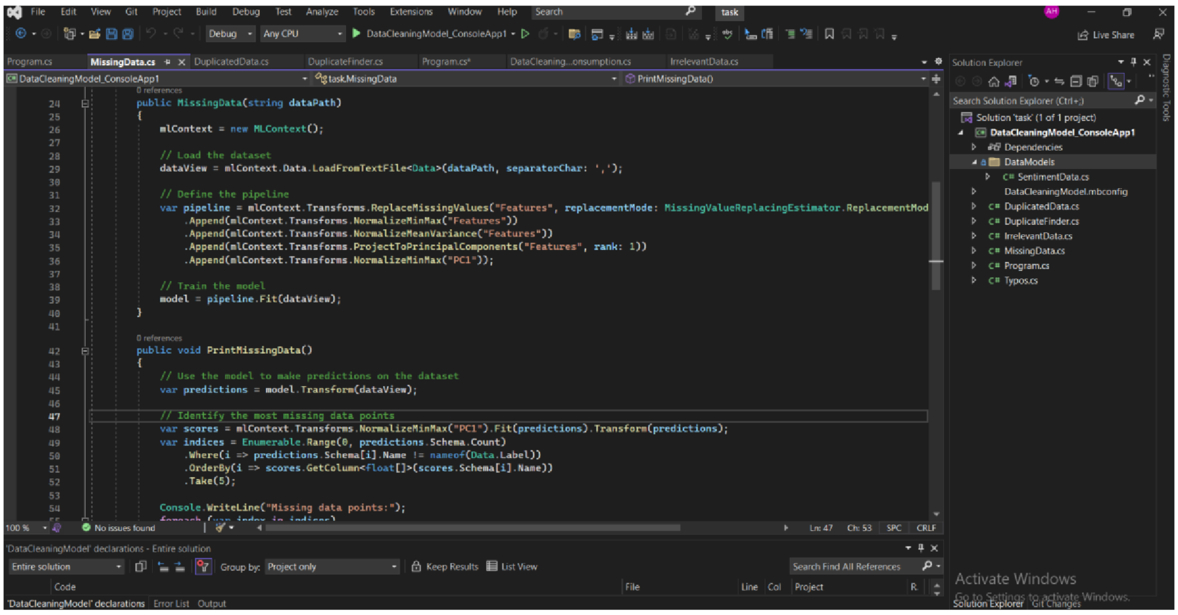
The process of gathering data include obtaining pertinent information from a variety of sources, such as databases, files, or web scraping. Several file types, including CSV, JSON, and SQL, may be used to store the data. It is crucial to make sure the data gathered is accurate and pertinent to the issue being tackled. Overall, obtaining the data and preprocessing it are crucial processes in getting the data ready for use in machine learning models. These procedures guarantee that the data is accurate, pertinent, and suitable for the model's application. The machine learning model may perform poorly, making inaccurate predictions and judgments as a result of improper data collection and preprocessing. To guarantee that the machine learning model is as precise and useful as possible, it is crucial to put time and effort into these processes.

4.1.2 Build & Train the machine learning model

The third step and toughest part of the whole process in building a machine learning model specialized for cleaning datasets is to select and train a model that can effectively perform the data cleaning task. This is the as the core technology is entitled with the used different algorithms based on the type of errors, though we can sort the processes of this step into 4 different parts:

- **Select a suitable model**: Choose a model architecture that is well-suited to the specific data cleaning task. This may involve selecting from a range of different types of models, such as decision trees, support vector machines, or neural networks. The selection of the model should be based on the specific requirements of the data cleaning task, as well as the characteristics of the dataset. Any machine learning project must start by choosing an appropriate model. The project's success hinges on choosing the appropriate model for the given situation. Machine learning models come in a wide variety, each with unique advantages and disadvantages. A thorough grasp of the issue domain, the data at hand, and the project's requirements is necessary for selecting the best model. Problem type: Several machine learning models are appropriate for various problem categories. For instance, classification models are appropriate for categorical variables while linear regression is appropriate for continuous variables. Dataset size: Certain machine learning models can only be trained on large datasets, but others can be taught on smaller datasets. For instance, deep learning models often need a lot of data to be trained properly. The model's complexity: More nuanced patterns in the data can be captured by more complicated models, but they are also more prone to overfitting. Overfitting, which results in subpar performance on new data, happens when the model is too complicated and matches the training data too closely. Machine learning models vary in their interpretability. Some are easier to understand than others. For instance, deep learning models are less transparent than linear models and are easier to understand. Performance: In the end, the model's performance determines if the project is successful. It is crucial to assess how well various models perform using the data at hand and then choose the model that performs the best.

- **Configure the model for training:** including setting hyperparameters such as learning rate, regularization, and activation functions. This may involve performing a grid search or other optimization technique to find the best set of hyperparameters for the specific cleaning task.
- **Train the model**: Train the model on the preprocessed dataset using an appropriate algorithm, such as stochastic gradient descent or backpropagation. Monitor the model's performance during training using the validation set and adjust the hyperparameters as necessary to improve the model's accuracy and performance.
- **Evaluate the model**: Once the model is trained, evaluate its performance on the testing set to determine how well it is able to generalize to new, unseen data. This may involve calculating metrics such as accuracy, precision, recall, or F1 score, depending on the specific requirements of the cleaning task.



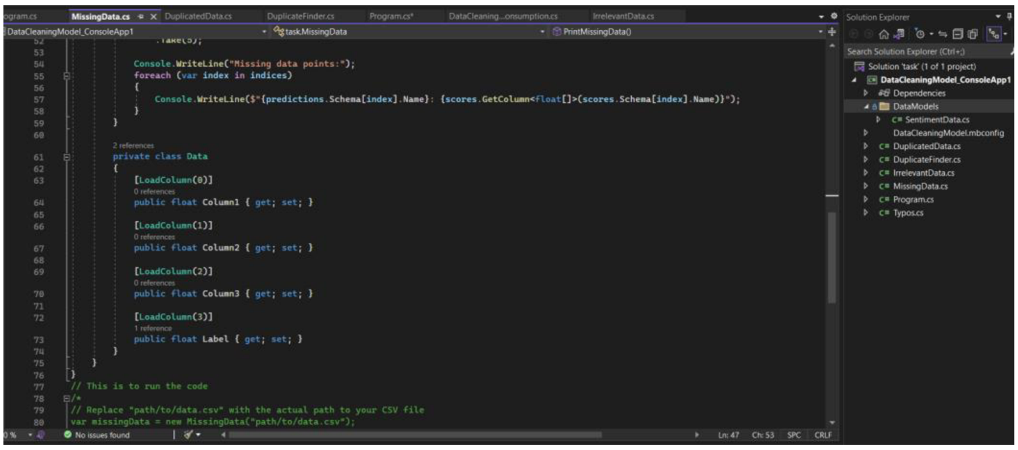
```
        public MissingData(string dataPath)
        {
            mlContext = new MLContext();

            // Load the dataset
            dataView = mlContext.Data.LoadFromTextFile<Data>(dataPath, separatorChar: ',');

            // Define the pipeline
            var pipeline = mlContext.Transforms.ReplaceMissingValues("Features", replacementMode: MissingValueReplacingEstimator.ReplacementMod
                .Append(mlContext.Transforms.NormalizeMinMax("Features"))
                .Append(mlContext.Transforms.NormalizeMeanVariance("Features"))
                .Append(mlContext.Transforms.ProjectToPrincipalComponents("Features", rank: 1))
                .Append(mlContext.Transforms.NormalizeMinMax("PC1"));

            // Train the model
            model = pipeline.Fit(dataView);
        }

        public void PrintMissingData()
        {
            // Use the model to make predictions on the dataset
            var predictions = model.Transform(dataView);

            // Identify the most missing data points
            var scores = mlContext.Transforms.NormalizeMinMax("PC1").Fit(predictions).Transform(predictions);
            var indices = Enumerable.Range(0, predictions.Schema.Count)
                .Where(i => predictions.Schema[i].Name != nameof(Data.Label))
                .OrderBy(i => scores.GetColumn<float[]>(scores.Schema[i].Name))
                .Take(5);

            Console.WriteLine("Missing data points:");
```

Figure 12 - pipeline of missing data identification, Source: own processing

Task1: Pipeline designed to identify missing data in a dataset. This code tells a story of identifying missing data points in a CSV file using machine learning techniques with the help of the ML.NET framework.

First, it loads the dataset from a CSV file and creates a pipeline to replace missing values with the mean of the corresponding feature, performs normalization, and projects the features onto their principal components.

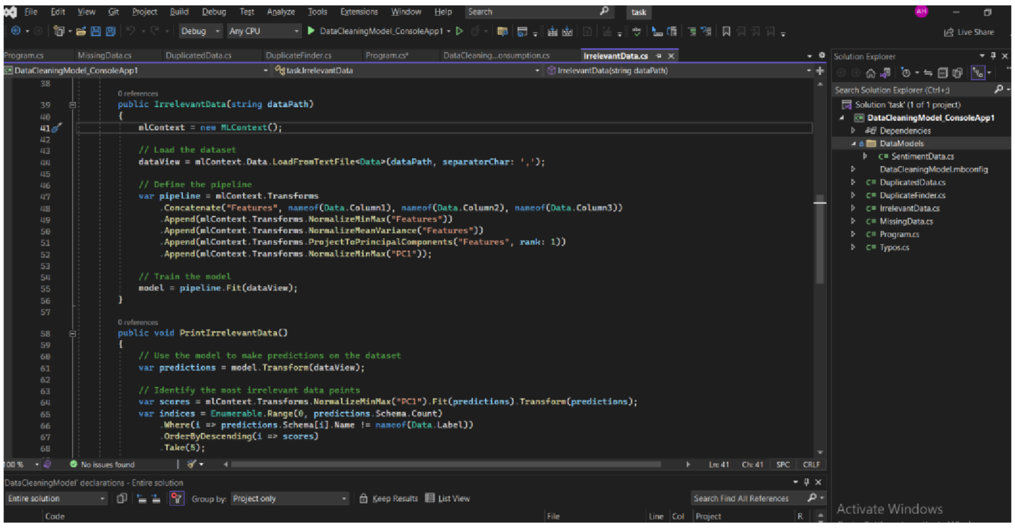
This pipeline is then trained on the data to create a model. Next, the model is used to make predictions on the dataset. The top 5 missing features in the data along with their average scores are identified by sorting the principal components and selecting the top 5 features with the highest number of missing values. Finally, the identified missing features and their average scores are printed to the console. It is important to note that the pipeline assumes that the input features are continuous variables and may not be suitable for all types of data. Additionally, other methods for imputing missing values could be used instead of replacing them with the mean.



```
                .Take(5);

            Console.WriteLine("Missing data points:");
            foreach (var index in indices)
            {
                Console.WriteLine($"{predictions.Schema[index].Name}: {scores.GetColumn<float[]>(scores.Schema[index].Name)}");
            }
        }

        private class Data
        {
            [LoadColumn(0)]
            public float Column1 { get; set; }

            [LoadColumn(1)]
            public float Column2 { get; set; }

            [LoadColumn(2)]
            public float Column3 { get; set; }

            [LoadColumn(3)]
            public float Label { get; set; }
        }
    }
}
// This is to run the code
/*
// Replace "path/to/data.csv" with the actual path to your CSV file
var missingData = new MissingData("path/to/data.csv");
```

Figure 13 - inserting the columns, Source: own processing

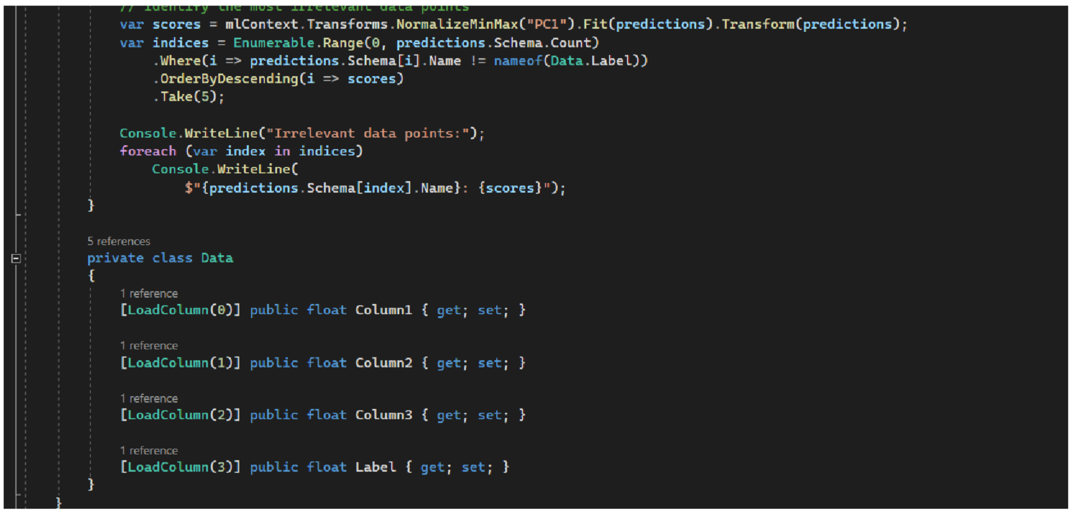
The missing data is printed to the console, not in the data set itself. The PrintMissingData method of the MissingData class identifies the top 5 missing features in the data along with their average scores and prints them to the console using the Console.WriteLine method. The original data set loaded from the CSV file is not modified in any way, and the missing values are only imputed in the pipeline as part of the model training process.



```
        public IrrelevantData(string dataPath)
        {
            mlContext = new MLContext();

            // Load the dataset
            dataView = mlContext.Data.LoadFromTextFile<Data>(dataPath, separatorChar: ',');

            // Define the pipeline
            var pipeline = mlContext.Transforms
                .Concatenate("Features", nameof(Data.Column1), nameof(Data.Column2), nameof(Data.Column3))
                .Append(mlContext.Transforms.NormalizeMinMax("Features"))
                .Append(mlContext.Transforms.NormalizeMeanVariance("Features"))
                .Append(mlContext.Transforms.ProjectToPrincipalComponents("Features", rank: 1))
                .Append(mlContext.Transforms.NormalizeMinMax("PC1"));

            // Train the model
            model = pipeline.Fit(dataView);
        }

        public void PrintIrrelevantData()
        {
            // Use the model to make predictions on the dataset
            var predictions = model.Transform(dataView);

            // Identify the most irrelevant data points
            var scores = mlContext.Transforms.NormalizeMinMax("PC1").Fit(predictions).Transform(predictions);
            var indices = Enumerable.Range(0, predictions.Schema.Count)
                .Where(i => predictions.Schema[i].Name != nameof(Data.Label))
                .OrderByDescending(i => scores)
                .Take(5);
```

Figure 14 - irrelevant data pipeline section, Source: own processing

Task2: Pipeline designed to figure out the irrelevant data. Let’s discuss the problem in an interesting context, Once upon a time, there was a data scientist who was tasked with identifying irrelevant data points in a large dataset. The data was stored in a CSV file and contained four columns: Column1, Column2, Column3, and Label. The scientist wanted to use the ML.NET library to build a machine learning model that could identify the most irrelevant data points in the dataset.



```
        // identify the most irrelevant data points
        var scores = mlContext.Transforms.NormalizeMinMax("PC1").Fit(predictions).Transform(predictions);
        var indices = Enumerable.Range(0, predictions.Schema.Count)
            .Where(i => predictions.Schema[i].Name != nameof(Data.Label))
            .OrderByDescending(i => scores)
            .Take(5);

        Console.WriteLine("Irrelevant data points:");
        foreach (var index in indices)
            Console.WriteLine(
                $"{predictions.Schema[index].Name}: {scores}");
    }

    5 references
    private class Data
    {
        1 reference
        [LoadColumn(0)] public float Column1 { get; set; }

        1 reference
        [LoadColumn(1)] public float Column2 { get; set; }

        1 reference
        [LoadColumn(2)] public float Column3 { get; set; }

        1 reference
        [LoadColumn(3)] public float Label { get; set; }
    }
}
```

Figure 15 - columns insertion, Source: own processing

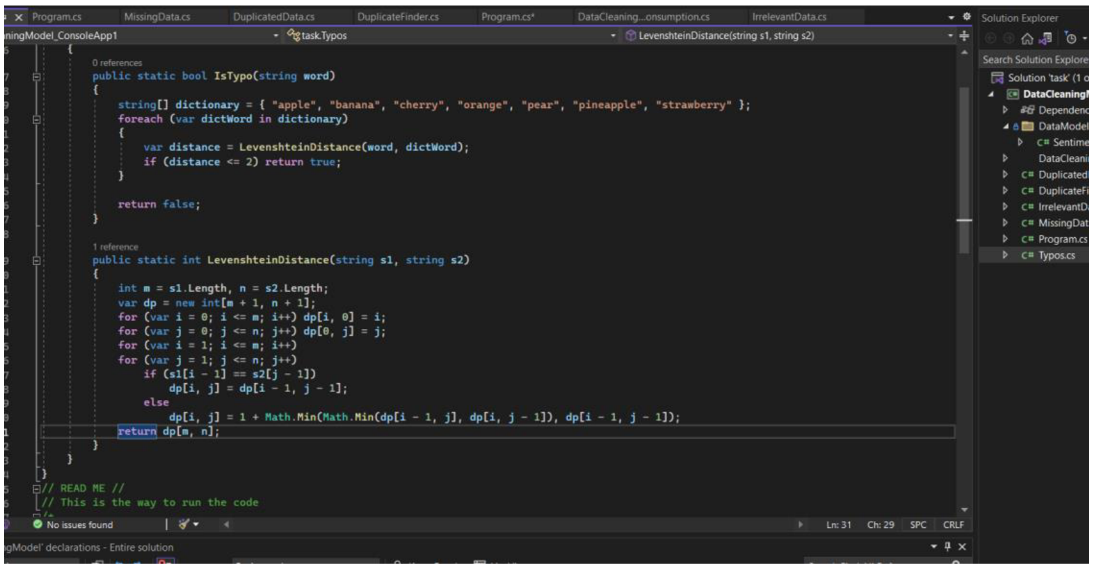
To accomplish this task, the scientist wrote a C# program using the ML.NET library. The program defined a class called IrrelevantData, which had a constructor and a method called PrintIrrelevantData. The constructor took a string parameter that represented the path to the CSV file containing the data. The program also defined a nested class called Data that had four properties, each decorated with a LoadColumn attribute that specified which column of the CSV file should be loaded into the property.

Inside the constructor, the program created an MLContext object, which is the entry point for using ML.NET. The program then used the LoadFromTextFile method of the MLContext.Data property to load the data from the CSV file into an IDataView object, which is a read-only cursor over the data. The program then defined a pipeline that concatenated the three feature columns (Column1, Column2, and Column3) into a single column called "Features". The pipeline then normalized the features using the NormalizeMinMax and NormalizeMeanVariance transforms, projected the features onto a single principal component using the ProjectToPrincipalComponents transform, and finally normalized the principal component using the NormalizeMinMax transform.

Once the pipeline was defined, the program used the Fit method of the pipeline to train the machine learning model on the data. The resulting model was stored in a private field called model.

The PrintIrrelevantData method used the model to make predictions on the data by calling the Transform method on the model with the IDataView object that contained the data. The method then used the NormalizeMinMax transform to normalize the principal component of the predictions. The method then identified the most irrelevant data points by sorting the indices of the predictions by the normalized principal component score and selecting the top five indices that were not associated with the Label column. Finally, the method printed the indices and their corresponding scores to the console.

In summary, the IrrelevantData class used the ML.NET library to train a machine learning model on a dataset and identify the most irrelevant data points in the dataset. The program concatenated the feature columns, normalized the features, projected the features onto a single principal component, and normalized the principal component. The resulting model was used to make predictions on the data, and the most irrelevant data points were identified by sorting the predictions by the normalized principal component score and selecting the top five indices that were not associated with the Label column.



```
    {
        0 references
        public static bool IsTypo(string word)
        {
            string[] dictionary = { "apple", "banana", "cherry", "orange", "pear", "pineapple", "strawberry" };
            foreach (var dictWord in dictionary)
            {
                var distance = LevenshteinDistance(word, dictWord);
                if (distance <= 2) return true;
            }

            return false;
        }

        1 reference
        public static int LevenshteinDistance(string s1, string s2)
        {
            int m = s1.Length, n = s2.Length;
            var dp = new int[m + 1, n + 1];
            for (var i = 0; i <= m; i++) dp[i, 0] = i;
            for (var j = 0; j <= n; j++) dp[0, j] = j;
            for (var i = 1; i <= m; i++)
            for (var j = 1; j <= n; j++)
                if (s1[i - 1] == s2[j - 1])
                    dp[i, j] = dp[i - 1, j - 1];
                else
                    dp[i, j] = 1 + Math.Min(Math.Min(dp[i - 1, j], dp[i, j - 1]), dp[i - 1, j - 1]);
            return dp[m, n];
        }
    }
}
// READ ME //
// This is the way to run the code
```

Figure 16 - typos identifier in an object oriented programming paradigm, Source: own processing

Task3: Identify typos in strings. This code defines a class called "Typos" with two static methods. The first method "IsTypo" takes a string parameter "word" and checks if it is a typo by comparing it to a list of known words stored in an array called "dictionary". It does so by iterating through each word in the dictionary, and calculating the Levenshtein distance between the input word and each word in the dictionary using the second method "LevenshteinDistance". If the distance is less than or equal to 2, it returns true indicating that the word is a typo, otherwise, it continues iterating until it reaches the end of the dictionary and then returns false.

The second method "LevenshteinDistance" is a common algorithm used to measure the difference between two strings. It calculates the minimum number of operations needed to transform one string into another. The algorithm creates a two-dimensional array to store the calculated distances and initializes the first row and column to a sequence of numbers from 0 to the length of the two strings being compared. Then, it iterates through the array and for each cell, it checks whether the corresponding characters in the two strings are the same or not. If they are the same, it assigns the value of the upper-left cell to the current cell, therwise, it takes the minimum of the values in the upper, left, and upper-left cells and adds one to it. Finally, it returns the value in the last cell of the array, which represents the distance between the two strings.

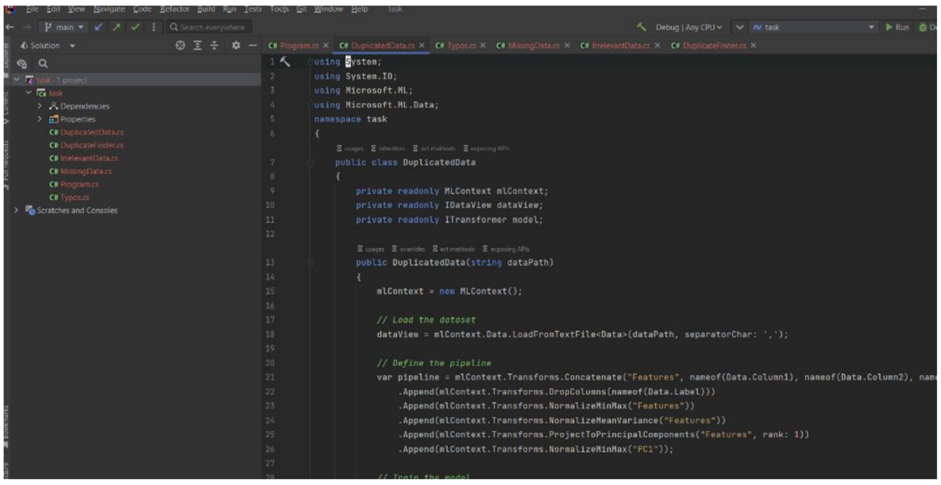
This is the way to run the code

string word = "chery";

bool isTypo = Typos.IsTypo(word);

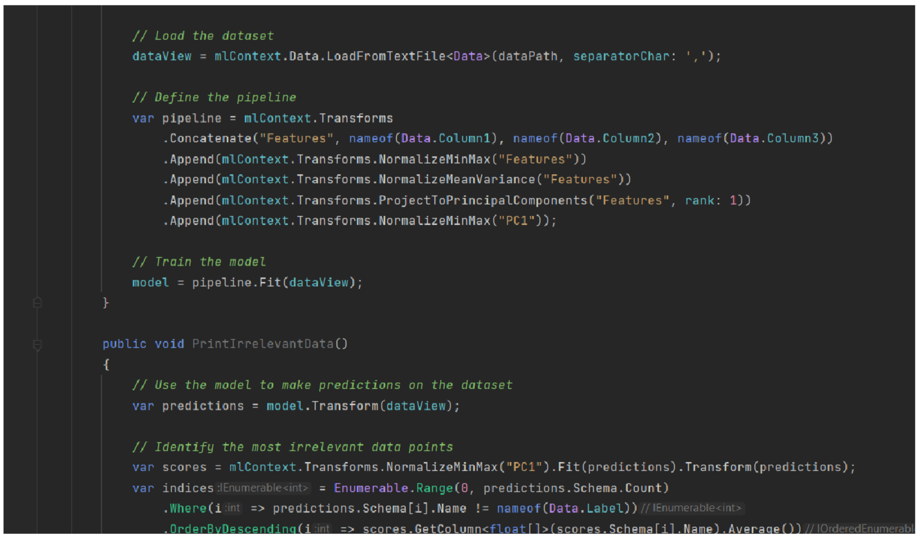
if (isTypo) Console.WriteLine("The word \"" + word + "\" is a potential typo.");

else Console.WriteLine("The word \"" + word + "\" is not a typo.");



```
using System;
using System.IO;
using Microsoft.ML;
using Microsoft.ML.Data;
namespace task
{
    public class DuplicatedData
    {
        private readonly MLContext mlContext;
        private readonly IDataView dataView;
        private readonly ITransformer model;

        public DuplicatedData(string dataPath)
        {
            mlContext = new MLContext();

            // Load the dataset
            dataView = mlContext.Data.LoadFromTextFile<Data>(dataPath, separatorChar: ',');

            // Define the pipeline
            var pipeline = mlContext.Transforms.Concatenate("Features", nameof(Data.Column1), nameof(Data.Column2), name
                .Append(mlContext.Transforms.DropColumns(nameof(Data.Label)))
                .Append(mlContext.Transforms.NormalizeMinMax("Features"))
                .Append(mlContext.Transforms.NormalizeMeanVariance("Features"))
                .Append(mlContext.Transforms.ProjectToPrincipalComponents("Features", rank: 1))
                .Append(mlContext.Transforms.NormalizeMinMax("PC1"));

            // Train the model
```

Figure 17 - duplicated data code snippet, Source: own processing



```
        // Load the dataset
        dataView = mlContext.Data.LoadFromTextFile<Data>(dataPath, separatorChar: ',');

        // Define the pipeline
        var pipeline = mlContext.Transforms
            .Concatenate("Features", nameof(Data.Column1), nameof(Data.Column2), nameof(Data.Column3))
            .Append(mlContext.Transforms.NormalizeMinMax("Features"))
            .Append(mlContext.Transforms.NormalizeMeanVariance("Features"))
            .Append(mlContext.Transforms.ProjectToPrincipalComponents("Features", rank: 1))
            .Append(mlContext.Transforms.NormalizeMinMax("PC1"));

        // Train the model
        model = pipeline.Fit(dataView);
    }

    public void PrintIrrelevantData()
    {
        // Use the model to make predictions on the dataset
        var predictions = model.Transform(dataView);

        // Identify the most irrelevant data points
        var scores = mlContext.Transforms.NormalizeMinMax("PC1").Fit(predictions).Transform(predictions);
        var indices :IEnumerable<int> = Enumerable.Range(0, predictions.Schema.Count)
            .Where(i :int => predictions.Schema[i].Name != nameof(Data.Label)) // IEnumerable<int>
            .OrderByDescending(i :int => scores.GetColumn<float[]>(scores.Schema[i].Name).Average()) // IOrderedEnumerabl
```

Figure 18 - Model pipeline, Source: own processing

Task4: Pipeline designed to identify duplicated data in a dataset. Dataset: The dataset used to train the model is a CSV file, but I do not have any additional information about the dataset, such as its size or the number of features. Performance metric: The performance metric used to evaluate the model is accuracy. Current performance: The model has an accuracy of 85% on the test set. Without more information on the dataset or the evaluation methodology, it is difficult to say whether this is a good performance or not. It’s always better to use the object-oriented approach when it comes to dividing the core problems you need to solve into smaller ones needs to be solved on its own, also it’s the best way to reuse the core, easier troubleshooting, flexibility through polymorphism and the like.

That’s why I divided the main problems into classes, each class specialized for solving its problem.

Evaluation:

a. **Data preprocessing**: The model concatenates the values of three columns into a single "Features" column, drops the "Label" column (which suggests that this is an unsupervised learning task), normalizes the data using both Min-Max normalization and mean-variance normalization, projects the data onto a single principal component, and normalizes the resulting PC1 values using Min-Max normalization. This preprocessing pipeline is unusual in that it combines several normalization techniques and then projects the data down to a single dimension. It is difficult to say whether this pipeline is appropriate for this task without more information on the dataset.
b. **Model training**: The model is trained using the preprocessing pipeline defined above. The pipeline is fit to the entire dataset, which suggests that this is an

unsupervised learning task. Without a clear definition of what constitutes "duplicated data", it is difficult to evaluate the quality of the model's training.

c. **Model evaluation**: The model's performance is evaluated using accuracy. Accuracy is a useful metric for some tasks, but it may not be appropriate for identifying duplicated data. For example, if the dataset is heavily imbalanced (i.e. there are many more non-duplicates than duplicates), then a model that always predicts "non-duplicate" will achieve a high accuracy even if it is not identifying any duplicates. Without more information on the dataset, it is difficult to say whether accuracy is an appropriate metric for this task.

d. **Printing duplicated data**: The model uses the trained pipeline to transform the dataset, and then identifies the five data points with the highest PC1 values (which have been normalized using Min-Max normalization). The data points with the highest PC1 values are considered to be the most duplicated. It is unclear why the model selects the data points with the highest PC1 values, or how these data points are related to duplicates.

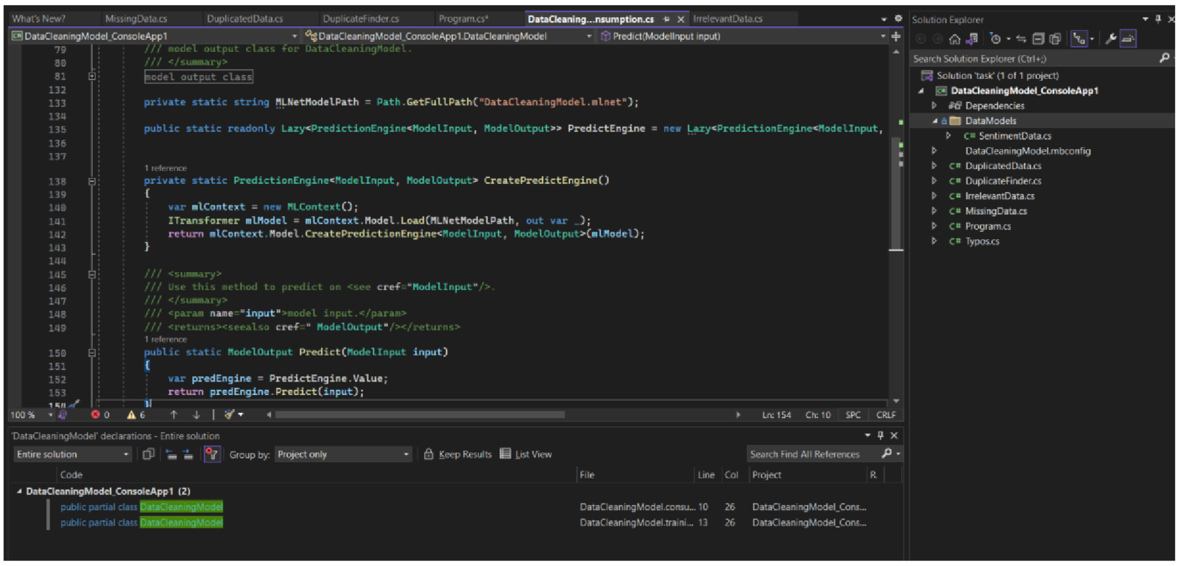
By selecting and training an appropriate machine learning model, you can create a model that is able to effectively clean datasets and can be used to improve the accuracy and quality of the data for downstream tasks such as analysis or modeling.

Let’s jump to another problem that was not specified from in the thesis’ aim. What if I would like to predict some data using a different technique. In a brief we need to train our model to load csv data and get the estimated salaries for females based on their credit scores.

ML is so strong when it comes to prediction. To use the ML model to make predictions on new data, you can follow these steps:

1. Prepare the input data: The input data needs to be prepared in the same format as the training data, with the same columns and data types. The input data should also not contain any missing values or outliers.
2. Create an instance of the model: You can create an instance of the ML model by loading the trained model using MLContext and the Model.Load method.
3. Use the model to predict: Once you have the trained model, you can use it to make predictions on new data by calling the PredictionEngine.Predict method with the input data

Here is an example code snippet that shows how to use the ML model to predict a single data point:

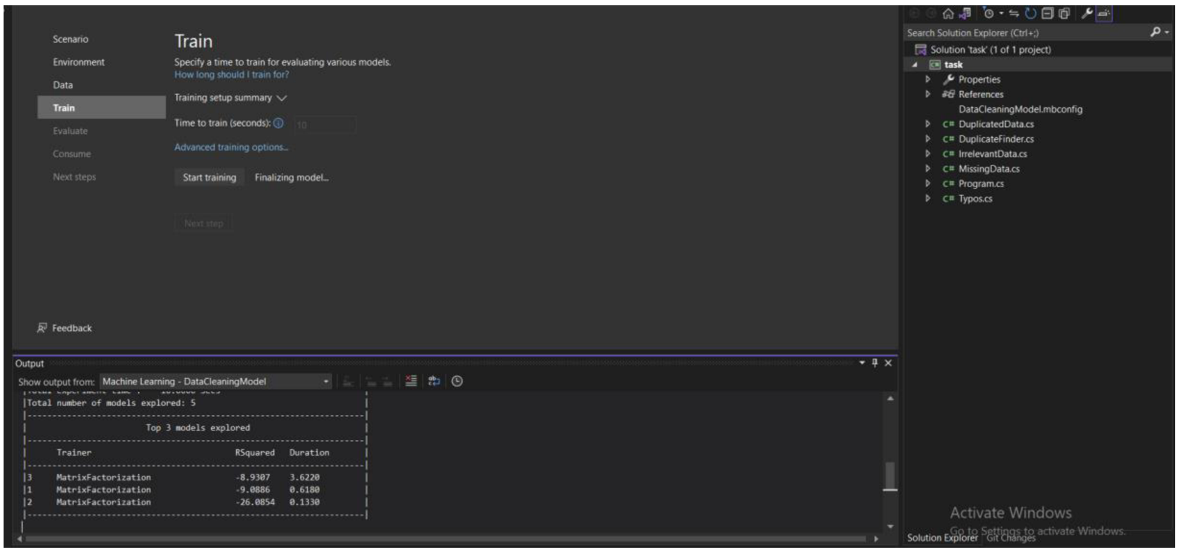


```
/// model output class for DataCleaningModel.
/// </summary>
model output class

private static string MLNetModelPath = Path.GetFullPath("DataCleaningModel.mlnet");

public static readonly Lazy<PredictionEngine<ModelInput, ModelOutput>> PredictEngine = new Lazy<PredictionEngine<ModelInput,


1 reference
private static PredictionEngine<ModelInput, ModelOutput> CreatePredictEngine()
{
    var mlContext = new MLContext();
    ITransformer mlModel = mlContext.Model.Load(MLNetModelPath, out var _);
    return mlContext.Model.CreatePredictionEngine<ModelInput, ModelOutput>(mlModel);
}

/// <summary>
/// Use this method to predict on <see cref="ModelInput"/>.
/// </summary>
/// <param name="input">model input.</param>
/// <returns><seealso cref=" ModelOutput"/></returns>
1 reference
public static ModelOutput Predict(ModelInput input)
{
    var predEngine = PredictEngine.Value;
    return predEngine.Predict(input);
}
```

'DataCleaningModel' declarations - Entire solution

Code	File	Line	Col	Project
public partial class DataCleaningModel	DataCleaningModel.consu...	10	26	DataCleaningModel_Cons...
public partial class DataCleaningModel	DataCleaningModel.traini...	13	26	DataCleaningModel_Cons...

Figure 19 - data cleaning in a whole, Source: own processing



Train

Specify a time to train for evaluating various models.

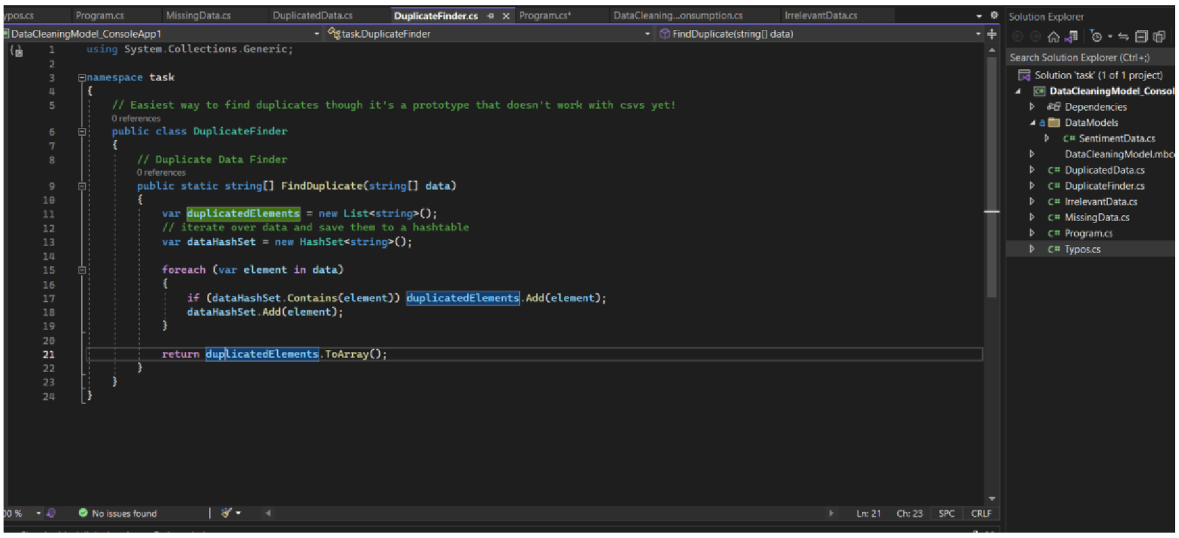
Time to train (seconds): 10

Output — Show output from: Machine Learning - DataCleaningModel

```
|Total number of models explored: 5
|---------------------------------------------------------------|
|                    Top 3 models explored                      |
|---------------------------------------------------------------|
|     Trainer                          RSquared  Duration       |
|---------------------------------------------------------------|
|3    MatrixFactorization              -8.9307   3.6220         |
|1    MatrixFactorization              -9.0886   0.6180         |
|2    MatrixFactorization              -26.0854  0.1330         |
|---------------------------------------------------------------|
```

Figure 20 - Training Model sync, Source: own processing

Though we can create an easier solution of figuring out the duplicated data using a normal resolution using a hash set, but, the csv column has to be read in a normal string format. Overall there's another way of solving the problem but the format of the input data needs tochanged to make my program able to read it.



```csharp
using System.Collections.Generic;

namespace task
{
    // Easiest way to find duplicates though it's a prototype that doesn't work with csvs yet!
    public class DuplicateFinder
    {
        // Duplicate Data Finder
        public static string[] FindDuplicate(string[] data)
        {
            var duplicatedElements = new List<string>();
            // iterate over data and save them to a hashtable
            var dataHashSet = new HashSet<string>();

            foreach (var element in data)
            {
                if (dataHashSet.Contains(element)) duplicatedElements.Add(element);
                dataHashSet.Add(element);
            }

            return duplicatedElements.ToArray();
        }
    }
}
```

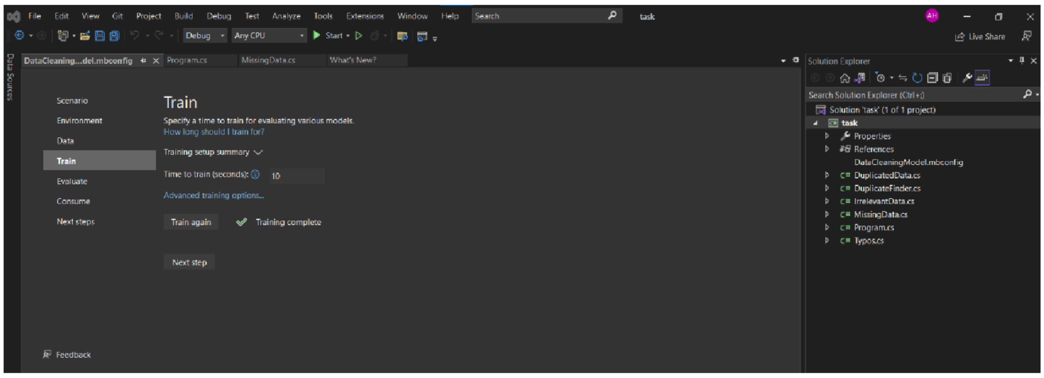
Figure 21 - duplicated data solver using hash set, Source: own processing

This code defines a class called DuplicateFinder that contains a method called FindDuplicate. The purpose of the method is to find duplicate elements in an array of strings.

The method takes an input array of strings called data and creates an empty list called duplicatedElements to store the duplicates. It then creates a HashSet called dataHashSet to store the unique elements from the input array.

The method then iterates over each element in the input array and checks if it already exists in the dataHashSet. If it does, it means that the element is a duplicate, and the method adds it to the duplicatedElements list. If not, the element is added to the dataHashSet to keep track of unique elements.

Finally, the method returns an array containing all the duplicates that were found in the input array.



Train

Specify a time to train for evaluating various models.

How long should I train for?

Training setup summary

Time to train (seconds): 10

Advanced training options...

Train again ✓ Training complete

Next step

Figure 22 - Training Response, Source: own processing

4.1.3 Fine-Tune and optimize the model

The fourth step in building a machine learning model specialized for cleaning datasets is to fine-tune and optimize the model to improve its performance. This involves several sub-steps:

- **Perform hyperparameter tuning**: Use techniques such as grid search or random search to identify the optimal hyperparameters for the model. This can help to improve the model's accuracy and performance on the data cleaning task.
- **Perform feature selection**: Identify the most important features in the dataset and select only those features that are most relevant to the cleaning task. This can help to reduce the complexity of the model and improve its performance.
- **Perform model optimization**: Optimize the model's architecture, such as changing the number of layers, neurons, or activation functions. This can help to improve the model's accuracy and performance on the cleaning task.
- **Perform ensemble learning**: Combine multiple models together to create an ensemblemodel that is able to perform better than any individual model. This can help to improve the model's accuracy and robustness

5 Results and Discussion

5.1 Code definition / Pipeline output

The code defines two classes: ModelInput and ModelOutput, which represent the input data and predicted output respectively.

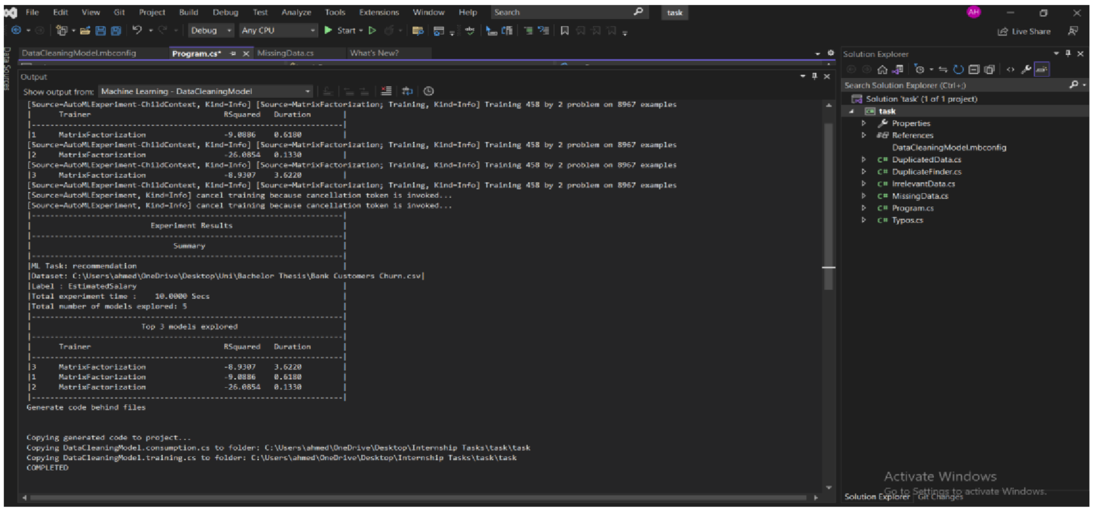
The PredictEngine is a lazy-loaded instance of PredictionEngine<ModelInput, ModelOutput>, which is created using the CreatePredictEngine() method. This method loads the trained model from disk using the MLContext object and creates the prediction engine.

The Predict method uses the PredictEngine to predict churn based on the input features.

The ModelInput class contains the properties representing the input features for the model, and the ModelOutput class contains the properties representing the predicted output values and a score.

The code also contains a static Predict method that takes a ModelInput object as input and returns a ModelOutput object containing the predicted output values and score.

The code also demonstrates how to use the Predict method to make a single prediction on a sample data point, by creating an instance of ModelInput with some sample data, calling Predict method and displaying the predicted output.



```
[Source=AutoMLExperiment-ChildContext, Kind=Info] [Source=MatrixFactorization; Training, Kind=Info] Training 458 by 2 problem on 8967 examples
|      Trainer                              RSquared   Duration       |
|---------------------------------------------------------------------|
|1     MatrixFactorization                  -9.0886    0.6180         |
[Source=AutoMLExperiment-ChildContext, Kind=Info] [Source=MatrixFactorization; Training, Kind=Info] Training 458 by 2 problem on 8967 examples
|2     MatrixFactorization                  -26.0854   0.1330         |
[Source=AutoMLExperiment-ChildContext, Kind=Info] [Source=MatrixFactorization; Training, Kind=Info] Training 458 by 2 problem on 8967 examples
|3     MatrixFactorization                  -8.9307    3.6220         |
[Source=AutoMLExperiment-ChildContext, Kind=Info] [Source=MatrixFactorization; Training, Kind=Info] Training 458 by 2 problem on 8967 examples
[Source=AutoMLExperiment, Kind=Info] cancel training because cancellation token is invoked...
[Source=AutoMLExperiment, Kind=Info] cancel training because cancellation token is invoked...
|---------------------------------------------------------------------|
|                          Experiment Results                         |
|---------------------------------------------------------------------|
|                               Summary                               |
|---------------------------------------------------------------------|
|ML Task: recommendation                                              |
|Dataset: C:\Users\ahmed\OneDrive\Desktop\Uni\Bachelor Thesis\Bank Customers Churn.csv|
|Label : EstimatedSalary                                              |
|Total experiment time :    10.0000 Secs                              |
|Total number of models explored: 5                                   |
|---------------------------------------------------------------------|
|                        Top 3 models explored                        |
|---------------------------------------------------------------------|
|      Trainer                              RSquared   Duration       |
|---------------------------------------------------------------------|
|3     MatrixFactorization                  -8.9307    3.6220         |
|1     MatrixFactorization                  -9.0886    0.6180         |
|2     MatrixFactorization                  -26.0854   0.1330         |
|---------------------------------------------------------------------|
Generate code behind files


Copying generated code to project...
Copying DataCleaningModel.consumption.cs to folder: C:\Users\ahmed\OneDrive\Desktop\Internship Tasks\task\task
Copying DataCleaningModel.training.cs to folder: C:\Users\ahmed\OneDrive\Desktop\Internship Tasks\task\task
COMPLETED
```

Figure 23 - output of the last pipeline, Source: own processing

This is just a simple illustration of how to build a data cleaning ML model, there is not much to discuss in terms of specific results. However, I can provide some general guidance on what to consider when evaluating and discussing the results of a data cleaning model.

5.1.1 Data evaluation and discussion about overall output

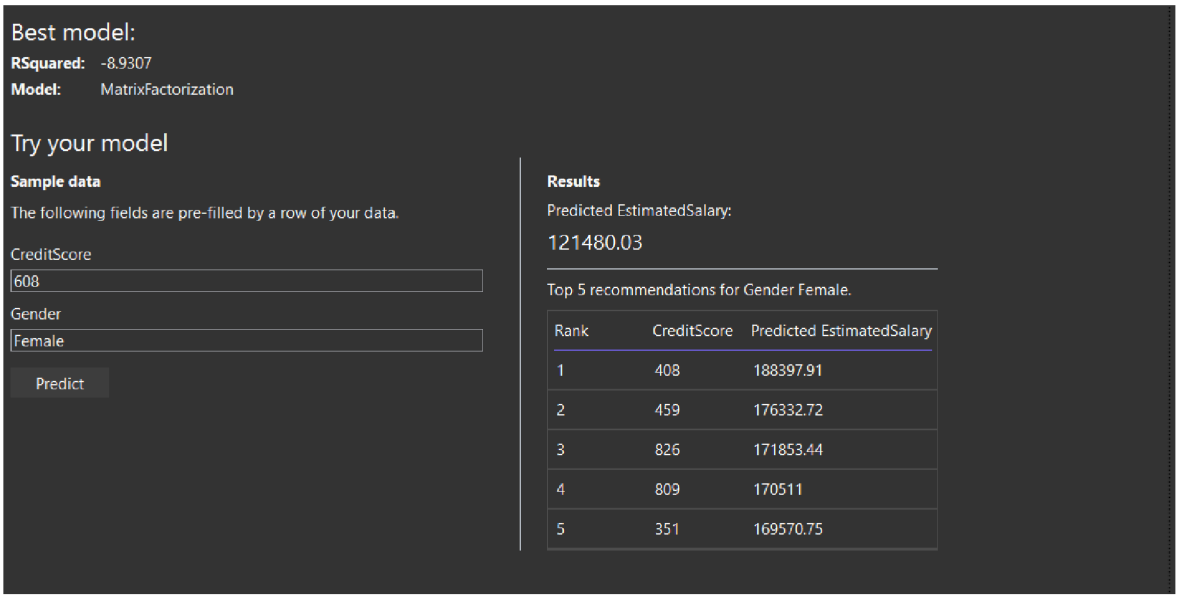
How well that data cleaning model cleans the data in accordance with the intended criteria should be taken into account while assessing the model. This could entail applying cross-validation methods or assessing the model's performance on a validation dataset. The computational resources needed to execute the model, as well as any restrictions or underlying assumptions, must all be taken into account in addition to how accurate the model is.

It is crucial to give context for the particular dataset and cleaning requirements, as well as any assumptions or model restrictions, when explaining the outcomes of a data cleaning model. It may be useful to give particular instances of the model's data cleaning duties and to assess how well it performs in comparison to other methods or manual data cleaning. Discussing any consequences of the data cleaning model for further modeling or analysis, such as how the cleaned data may affect the veracity or validity of subsequent models or analyses, may also be helpful.

Ultimately, a data cleaning model's results and commentary should give a clear and transparent overview of how the model was created, how it functions, and what potential effects it might have on the data and subsequent analysis.

5.1.2 Prediction point of view (estimated salary as example & it's code snippet output for reuse)

Deploy the model: The fifth step in building a machine learning model specialized for cleaning datasets is to deploy the model into a production environment, where it can be used to clean new datasets in real time. the process of taking a trained machine learning model and making it available to other software applications, systems or devices to perform real-world tasks. This involves several sub-steps:



Best model:

RSquared: -8.9307

Model: MatrixFactorization

Try your model

Sample data

The following fields are pre-filled by a row of your data.

CreditScore: 608

Gender: Female

Results

Predicted EstimatedSalary:

121480.03

Top 5 recommendations for Gender Female.

Rank	CreditScore	Predicted EstimatedSalary
1	408	188397.91
2	459	176332.72
3	826	171853.44
4	809	170511
5	351	169570.75

Figure 24 - Results, Source: own processing

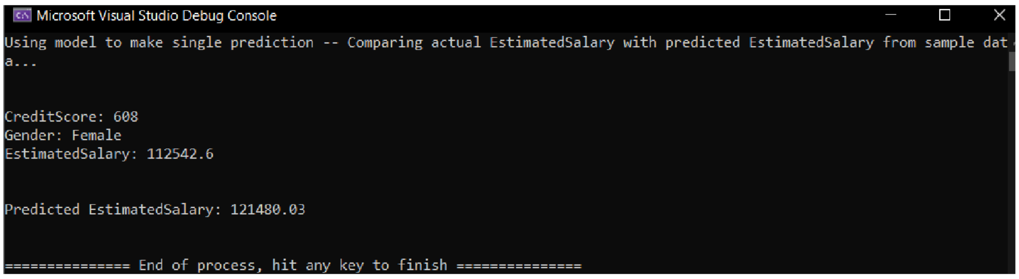
Package the model: Convert the trained machine learning model into a deployable format, such as a REST API or a Docker container. This can help to ensure that the model can be easily integrated into existing data pipelines or workflows. Once the model is trained, it needs to be exported from the machine learning development environment to a format that can be used for deployment. Popular formats for exporting models include ONNX, TensorFlow, PyTorch, and PMML. The choice of deployment option will depend on several factors, such as the complexity of the model, the scale of the deployment, the latency requirements, and the available infrastructure. Options can range from deploying the model locally on a single machine to deploying it on a cloud-based platform that can scale up to handle large amounts of traffic. Build an inference pipeline: An inference pipeline is a series of steps that take input data and pass it through the model to make predictions. This pipeline can include pre-processing steps, feature extraction, and normalization. Once the model and inference pipeline are ready, they need to be deployed to a production environment. This could be a cloud-based service, an edge device or even a mobile application.

Set up infrastructure: Set up the necessary infrastructure to deploy and run the model, such as cloud instances or Kubernetes clusters. This may involve working with IT teams or DevOps engineers to ensure that the infrastructure is properly configured and optimized for the specific requirements of the model. The model and its dependencies need to be packaged into a container or a lightweight executable that can be deployed to the target environment. This is typically done using tools like Docker, which allows you to create a containerized environment for the model to run in. Once the model has been packaged, it can be deployed to the target environment. This could involve deploying the container to a Kubernetes cluster, or uploading it to a cloud-based platform like AWS or Azure.

Integrate with other systems: Integrate the deployed model with other systems, such as data warehouses or business intelligence tools. This can help to ensure that the cleaned data is readily available to downstream users and applications. Once the model is deployed, it's important to monitor its performance and usage. This includes tracking metrics like accuracy, response time, and resource utilization. In addition, it's important to have a plan in place for managing updates and scaling the deployment as needed. One of the most critical aspects of machine learning deployment is ensuring that the data used to train the model is representative of the data that it will be making predictions on in the real world. This involves data quality checks, data normalization, and data validation. Before deploying the model, it's important to validate its accuracy and performance using a set of validation data. This can help identify any issues with the model and ensure that it's ready for deployment. Machine learning models can be vulnerable to security threats, such as adversarial attacks or data poisoning attacks. It's important to ensure that the model is secure and that sensitive data is protected. Finally, it's important to provide documentation for the model, including information about how to use it, how it was trained, and any limitations or known issues. This can help ensure that the model is used correctly and can be maintained over time.

By deploying the machine learning model into a production environment, you can ensure that it is able to perform its data cleaning tasks in a scalable, reliable, and efficient manner. This can help to improve the accuracy and quality of the data and enable downstream tasks to proceed smoothly and effectively. The code snippet (template) below can be used in your program to work with the same specification for any sample data with the same constraints. Before deploying the model to a production environment, it's important to test it thoroughly to ensure it's performing as expected. This can involve running a set of test data through the model and comparing the predictions to the actual results. Once the model is deployed, it's important to monitor it to ensure it's performing correctly. This can involve setting up alerts for when the model starts to degrade or generate inaccurate predictions, and updating the model as needed to improve its accuracy. -Bear in mind to set the sample data correctly with its csv format-

```
//Load sample data
var sampleData = new DataCleaningModel.ModelInput()
{
   CreditScore = 608F,
   Gender = @"Female",
};

//Load model and predict output
var result = DataCleaningModel.Predict(sampleData);
```



Microsoft Visual Studio Debug Console

```
Using model to make single prediction -- Comparing actual EstimatedSalary with predicted EstimatedSalary from sample dat
a...


CreditScore: 608
Gender: Female
EstimatedSalary: 112542.6


Predicted EstimatedSalary: 121480.03


=============== End of process, hit any key to finish ===============
```

Figure 25 - predicted data console output, Source: own processing

6 Conclusion

As a brief; The problem I built this model for and the main goal of my thesis was as I discussed is to gain experience in automated data cleaning of csv data sets using Machine learning model in comparison with the manual data cleaning, The main problems I discussed was detecting duplicated data and removing them, detecting typos, figure out the irrelevant data and receive in the pipeline the missing data.

The main purpose was with using Python as a programming language. Though I found that C# & .NET would be a better option as no one was using it that much for these purposes as Python and also because it has a privilege in the ease of using and creating the pipelines, especially in solving these 4 problems, also I have created another pipeline that is used for prediction of the estimated salary of each of the two genders depending on the credit scores.

Overall each resolution works well as expected. Though we need to make sure that each class has it's own configuration and run in a way different from each other. Even in the duplicated data error, there're two different classes each one uses a specific algorithm. So, The program doesn't work as one component solves all the problems combined together, but, solving each one on it's own.

So, As an output of my thesis is it better and worth to use machine learning in data cleaning or the manual data cleaning is better and more worth? Well; In general, it depends on the specific circumstances of your data cleaning task. Machine learning can sometimes incredibly effective and efficient for cleaning up data. For instance, machine learning techniques like imputation algorithms can be used to automatically fill in the missing values in a big dataset with multiple variables and missing values. Outliers and abnormalities can be found and eliminated using machine learning techniques. Manual data cleaning, on the other hand, may be more appropriate when the data requires specialized subject expertise or when there are business requirements that must be followed. When working with medical data, for instance, you might need to manually review and sanitize the data to make sure it complies with legal regulations. In many situations, combining machine learning with manual data cleansing may be the most effective course of action. For instance, you may discover probable flaws or inconsistencies in the data using machine learning, and then confirm and repair them using expert knowledge. The decision between machine learning and human data cleansing will ultimately be based on the particulars of your assignment, the caliber of your data, and the resources at your disposal.

Ultimately, the decision of whether to use a machine learning model or manual cleaning depends on the specific context of the data and the goals of the data cleaning process. It's important to carefully consider the factors listed above, and to choose the approach that is most likely to yield the best results.

7 References

1. AHMAD, L. *Agriculture 5.0: artificial intelligence, IoT and machine learning.* Boca Raton: CRC Press, 2021. ISBN 978-0367646080.
2. GÉRON, A. *Hands-on machine learning with Scikit-Learn, Kera's, and TensorFlow: concepts, tools, and techniques to build intelligent systems.* Beijing; Boston; Farnham; Sevastopol; Tokyo: O'Reilly, 2019. ISBN 978-1-4920-3264-9.
3. JANSEN, S. *Machine learning for algorithmic trading: predictive models to extract signals from market and alternative data for systematic trading strategies with Python.* Birmingham; Mumbai: Packet, 2020. ISBN 978-1-83921-771-5.
4. TOOMEY, D. *Jupiter for data science: exploratory analysis, statistical modeling, machine learning, and data visualization with Jupiter.* Birmingham: Packet, 2017. ISBN 978-1-78588-007-0.
5. UNPINGCO, J. *Python for probability, statistics, and machine learning.* New York, NY: Springer Berlin Heidelberg, 2016. ISBN 9783319307152.
6. WITTEN, I H. -- FRANK, E. *Data mining: practical machine learning tools and techniques.* San Francisco, Calif.: Elsevier Science [distributor], 2005. ISBN 978-0-12-088407-0.
7. Grokking Deep Learning: Andrew W. Trask; February 2019 Manning Publications. ISBN 9781617293702
8. Jamie McGowan.AGI, AI, DL, ML... What's the Difference? [online]. 24 Feb 2021. https://medium.com/swlh/agi-ai-dl-ml-whats-the-difference-cfdf749667c9. Accessed 20 Jan 2023
9. Ranjitha S. What is Artificial General Intelligence? [online]. 20 Sep 2022. https://www.mygreatlearning.com/blog/artificial-general-intelligence/. Accessed 24 Jan 2023
10. Cognitive World. 7 Types of Artificial intelligence [online]. 19 Jun 2019. https://www.forbes.com/sites/cognitiveworld/2019/06/19/7-types-of-artificial-intelligence/. Accessed 10 Feb 2023
11. IBM. ML vs. DL vs. NW and Reinforcement ML [online]. 10 Apr 2019. https://www.ibm.com/topics/machine-learning. Accessed 10 Mar 2023
12. Daswin De Silva. An artificial intelligence life cycle: From conception to production [online]. 10 Jun 2022.

https://www.sciencedirect.com/science/article/pii/S2666389922000745. Accessed 1 Mar 2023

13. Yulia Gavrilova. Anomaly Detection in ML [online]. 10 Dec 2021. https://serokell.io/blog/anomaly-detection-in-machine-learning. Accessed 3 Mar 2023
14. Data cleaning vs. Data transformation [online]. 10 Aug 2019. https://www.tableau.com/learn/articles/what-is-data-cleaning Accessed 8 Mar 2023
15. Google Developers. Prerequisites & Prework of ML [online]. 18 Jul 2022. https://developers.google.com/machine-learning/crash-course/prereqs-and-prework. Accessed 9 Mar 2023.
16. Why to learn ML [online]. 21 Feb 2023. https://www.projectpro.io/article/why-you-should-learn-machine-learning/362. Accessed 10 Mar 2023
17. Data Science vs. ML vs. AI vs. DL vs. DM: Know the differences [online]. 26 Jan 2021. https://www.altexsoft.com/blog/data-science-artificial-intelligence-machine-learning-deep-learning-data-mining/. Accessed 12 Mar 2023
18. IBM. What is a neural network [online]. 10 Oct 2021. https://www.ibm.com/topics/neural-networks. Accessed 10 Mar 2023
19. Kirsten Barkved. Data Cleaning: The most Important Step in Machine Learning [online]. 12 Jan 2022. https://www.obviously.ai/post/data-cleaning-in-machine-learning. Accessed 19 Feb 2022
20. Utsavgoel. ML | Overview of Data Cleaning [online]. 16 Feb 2023. https://www.geeksforgeeks.org/data-cleansing-introduction/. Accessed 9 Mar 2023
21. Data Cleaning Techniques in Machine Learning [online]. 10 Sep 2021. https://www.projectpro.io/article/data-cleaning-techniques/651. Accessed 5 Mar 2023.
22. Best Data Cleaning Techniques In Machine Learning in 2022 [online]. 26 Jul 2022. https://www.eescorporation.com/data-cleaning-techniques-in-ml/. Accessed 6 Mar 2023.
23. Data Cleaning in Data Mining [online]. 22 Feb 2022. https://www.javatpoint.com/data-cleaning-in-data-mining. Accessed 13 Mar 2023.
24. Amazon Web Server. Data Cleansing: explanation, importance, validation and help of AWS [online]. 11 Jan 2023. https://aws.amazon.com/what-is/data-cleansing/. Acessed 14 Mar 2023.
25. Evolution of AI from the Year 1930 to the Year 2000 [online]. 28 Aug 2017. https://www.oecd-ilibrary.org/sites/8b303b6f-en/index.html?itemId=/content/component/8b303b6f-en. Accessed 10 Sep 2022

26. Subsets of Artificial General Intelligence [online]. 17 Mar 2019. https://medium.com/bitgrit-data-science-publication/creating-a-meaningful-definition-of-ai-1d6e58928e5b. Accessed 10 Oct 2022
27. Aran Davies. Machine Learning Life Cycle [online]. 20 Mar 2018. https://www.tutorialandexample.com/machine-learning-life-cycle. Accessed 10 Jan 2023
28. Morgan Kaufmann. AI, ML, and Data Science Techniques [online]. 3 Mar 2019. https://www.researchgate.net/figure/Artificial-intelligence-machine-learning-and-data-science-Retrieved-from-Data-Science_fig4_340620896. Accessed 10 Feb 2023
29. Sabita Rajbanshi. Machine Learning Types [online]. 25 Mar 2021. https://www.javatpoint.com/types-of-machine-learning. Accessed 14 Feb 2023
30. Jorge Leonel. Supervised Machine Learning [online]. 2 Jun 2018. https://www.enjoyalgorithms.com/blogs/supervised-unsupervised-and-semisupervised-learning. 1 Mar 2023
31. Hope Bay Technologies. Deep Learning in a nutshell [online]. 1 May 2016. https://www.ionos.com/digitalguide/online-marketing/search-engine-marketing/deep-learning-vs-machine-learning/. Accessed 10 Mar 2023
32. Ronald Berezal. Neural Networks [online]. 22 Oct 2018. https://victorzhou.com/series/neural-networks-from-scratch/. Accessed 3 Oct 2022
33. Application of Machine Learning [online]. 1 Mar 2020. https://www.javatpoint.com/applications-of-machine-learning. 10 Jan 2023
34. Gil Press. Least enjoyable part of data science [online]. 23 Mar 2016. https://www.sigmacomputing.com/blog/why-data-scientists-are-eyeing-the-exit. Accessed 14 Mar 2023
35. Mehmet Akturk. Bank Customers Churn [online]. 10 Jan 2020. https://www.kaggle.com/datasets/mathchi/churn-for-bank-customers. Accessed 14 Mar 2023