Building Digital Twins

Mr Ashlin Darius Govindasamy
University of South Africa
Department of Mathematics and Computer Science

July 4, 2022

Abstract

A digital twin is a virtual representation of a physical object that is designed to exactly reflect it. The object under investigation — say, a wind turbine — is equipped with a variety of sensors that monitor various aspects of its operation. These sensors collect data on the energy production, temperature, weather conditions, and other characteristics of the physical object's performance. This information is subsequently sent to a processing machine, where it is applied to a digital copy. Once the virtual model has been given this information, it may be used to run simulations, investigate performance concerns, and suggest improvements, all with the purpose of gaining important insights that can later be applied to the original physical device. In this paper methods of machine learning will be discussed, statistics, forecasting, mathematical equations, real-world examples, and methodology for building digital twins.

Keywords: machine-learning; digital twins; forecasting; statistics; equations; dashboard

Contents

Chapter 1

Introduction

A digital twin is a representation of a physical thing, process, or service in digital form. A digital twin is a computer representation of a real thing, such as a jet engine or wind farm, or even larger objects, such as skyscrapers or entire towns. The digital twin technology can be used to reproduce processes in order to collect data and anticipate how they will perform, in addition to real assets. A digital twin is a computer program that creates simulations based on real-world data to forecast how a product or process will perform. To improve the output, these applications can use the internet of things (Industry 4.0), artificial intelligence, and software analytics. These virtual models have become a standard in modern engineering to drive innovation and enhance performance, thanks to advancements in machine learning and elements such as big data. In brief, having one can help improve strategic technological trends, prevent costly breakdowns in physical objects, and test processes and services utilizing superior analytical, monitoring, and predictive capabilities. The development of a mathematical model that simulates the original begins with professionals in applied mathematics or data science examining the physics and operational data of a physical object or system. The creators of digital twins make certain that the virtual computer model may get feedback from sensors that collect data from the real-world version. This allows the digital version to mimic and emulate what is happening with the original version in real-time, allowing for the collection of data on performance and potential issues. A digital twin can be as complicated or as basic as you need it to be, with the quantity of data used deciding how closely the model mimics the actual version in the real world. The twin can be used in conjunction with a prototype to provide feedback on the product as it is

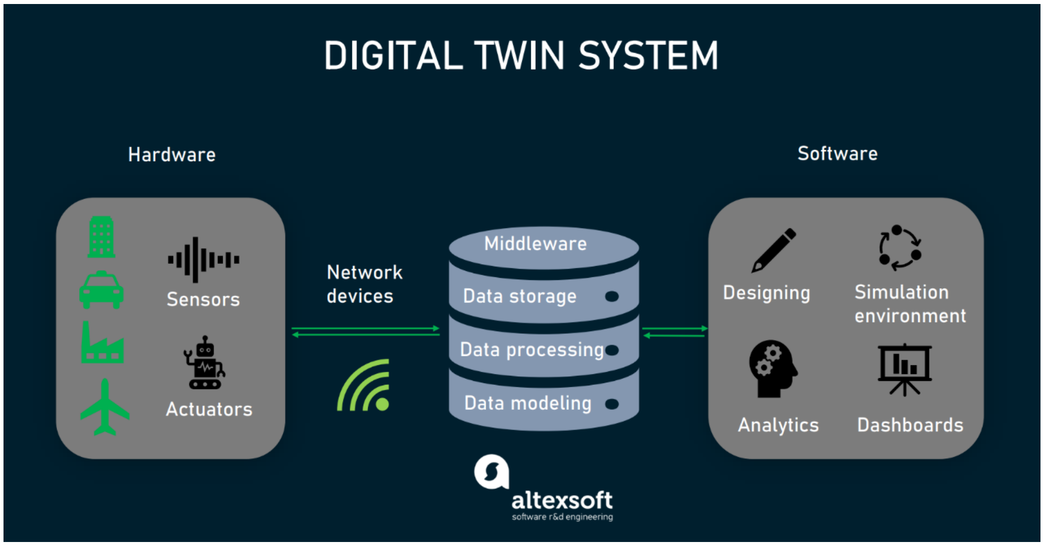


Figure 1.1: Components of a Digital Twin.

being created, or it can be used as a prototype in and of itself to simulate what might happen when the real version is built.

1.0.1 Background Knowlege of Digital Twins

You must first comprehend what you are attempting to construct when creating a Digital Twin. When creating a digital twin of a water plant or a power plant, for example, you'll need to understand how the facility works in real life. To run analytics on that plant, you'll need to gather information about it. Companies will make it difficult for you to obtain data because exposing the info to the public could be dangerous. Let's say you obtained the information by obtaining permission or collecting it yourself or with the help of coworkers. You must be aware of the variables, qualities, and influences on plant changes. You must ask yourself this question. What we want to foretell or what we need to forecast is the question you must now ask yourself. We can start building our digital twin after we know what our goals are and have a good understanding of our data. I've included some questions below that may assist you in asking the right questions as you begin to develop your digital twin.

- Do we understand our dataset?

- Is our data useful to make predictions?
- Does our dataset have outliers? Yes, then the software should flag/alert outliers or remove them. When running predictions?
- What behavior are we trying to predict or forecast?
- Are there any relationships in our dataset's parameters/variables?
- What machine learning methods are we going to use to run analytics/predictions?

1.0.2 Aim of this study and motivation

If you look at organizations that offer or construct Digital Twin software, you'll see that they charge tens of thousands of dollars to build these solutions. A corporation approached me and told me about the plant's digital twin problem statement. I promised them I could build it for hundreds of dollars in a few weeks. There aren't a lot of resources available on digital twins. I'm filling in the blanks with my experience and knowledge from a solid background in Data Science, Mathematics, and Computer Science. This paper is intended to inspire you to make your businesses smarter and more inventive. It is the fourth Industrial Revolution, and it is time for a shift. The goal of this research is to teach you how to create digital twins. The process for creating digital twins and the knowledge I gained on my own while creating a digital twin of a plant with no prior understanding of digital twins but a background in statistics, data science, and machine learning.

1.0.3 Prerequisites

- You will need to have a good understanding of a good statistical/data science language example Python or R
- Understanding of statistics and mathematics (if creating custom machine learning models or neural networks) for optimizing machine learning models
- Experience in Web Frameworks (used for creating Web Dashboards), Machine Learning Modules (required to forecast and predict data), and a graphing library (MATLAB, Seaborn, etc)

Chapter 2

Methodology

2.1 Data Dashboards in Data Science

A data dashboard is a tool that businesses use to track, analyse, and display data, usually to gain a better understanding of the overall health of the organization, a department, or even a specific process. Dashboards connect a variety of metrics, data sources, APIs, and services behind the scenes, assisting businesses in extracting relevant information from those sources and displaying it in user-friendly ways. Data dashboards, similar to a car's dashboard, organize and display important information at a glance to help you understand your company's most valuable data and find answers to critical questions. By linking dashboards to specific metrics or key performance indicators (KPIs), you gain valuable business intelligence as well as the ability to delve deeply into specific pieces of information to continuously monitor success. Dashboards, like those in cars, show how far you are along your journey and how long it may take to get to your destination. A dashboard's ability to provide up-to-date information and context to help inform business decisions and empower employees is one of its most powerful features. A dashboard, for example, could be used by an IT team to help detect signs of a security breach. Alternatively, a company could incorporate the dashboard into an app or mobile device for first-line workers in the field, ensuring that they have access to the data they require at all times. As for the digital twin case in this paper, our paper will be used visualize data, forecast data for Nth number of years, make predictions on a given date, find out the impacts of parameters in your dataset and find out how to optimize the production

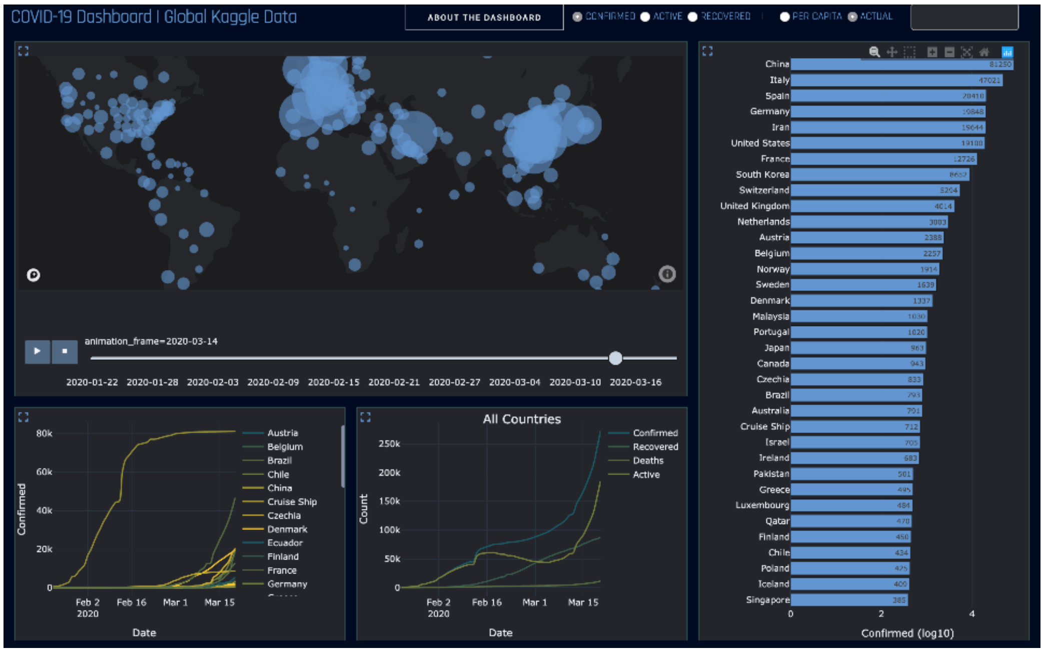


Figure 2.1: Example of a Covid-19 Dashboard just visualizing data with filters.

line or produce the best outcome for your plant/company. Our parameters (dataset headers example .csv) can be described as a set of

$$\{p_n | n = p_1, p_2, p_2, \dots\}$$

where p_n is a parameter.

Let's say for example in a water plant our p_1 can be DateAndTime , p_2 = PropertyMeasurement , p_3 = SomeChemicalCompound . . . p_n = our dataset parameters for a given index We could sketch graphs on this data and predict values based on parameter behaviors. We can play around, modify some parameters and see what effect that makes on our plant. Our AI can also optimize the plant for the best-fine-tuned results for our plant. It can generate insights for us and tell us how we can improve the quality of our plant or object. Below are examples of the dashboard and the functionality it presents to us.

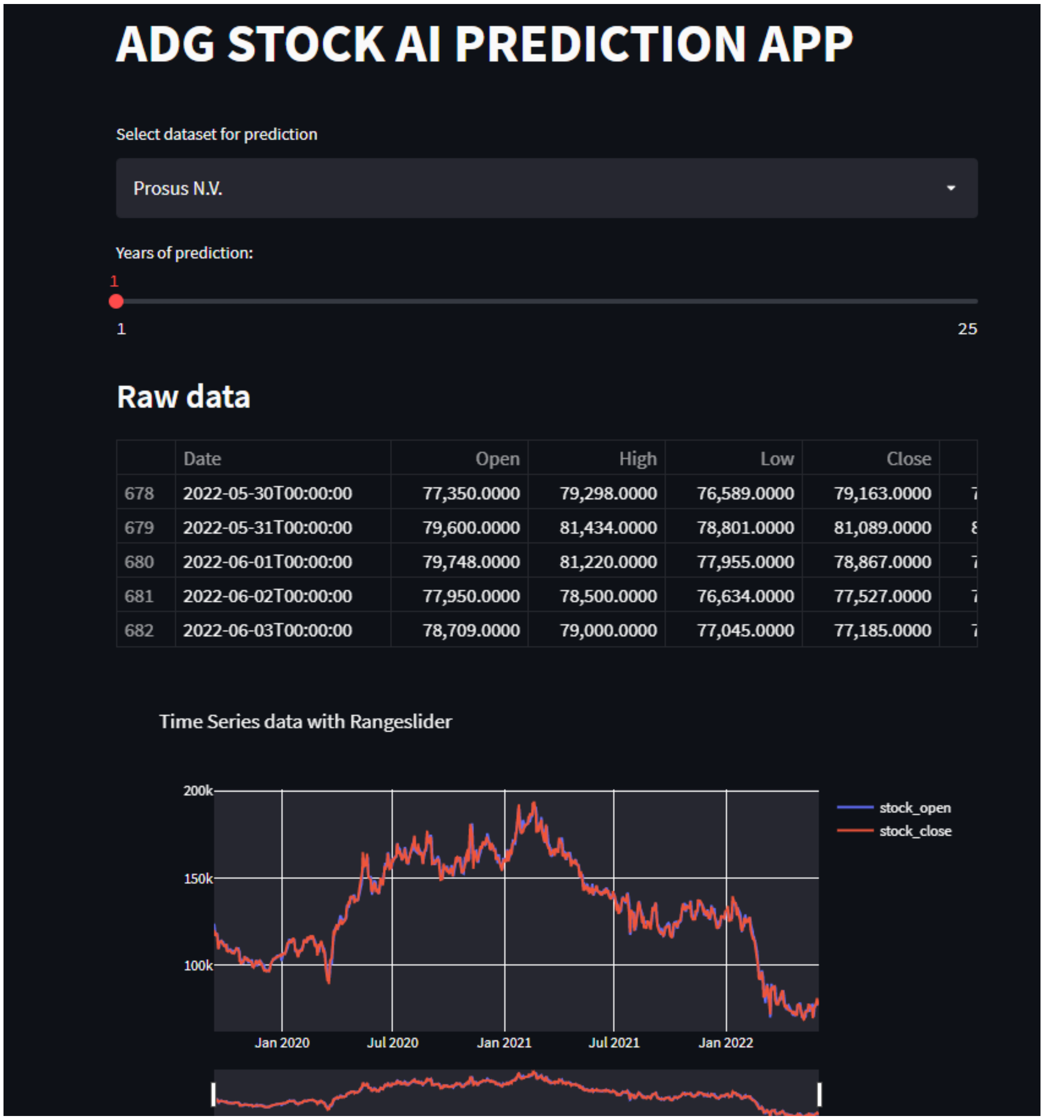


Figure 2.2: Example of my application dashboard displaying JSE stocks for a given date interval and forecasting stocks for a given number of years https://stonks.adgstudios.co.za

2.2 Machine Learning and Forecasting

2.2.1 Introduction to Machine Learning and Forecasting for building digital twins

Machine learning and forecasting are important tools in the digital twin toolbox. It is the most important component of the digital twin; without these tools and knowledge, the digital twin would not exist. Artificial Intelligence includes the subcategory of Machine Learning. Machine Learning is used to find trends/equations created from Machine Learning models that best describe our data and will be used to make predictions based on input and output data. Let me describe the above in simple mathematical equations.

Let's take our old-school way of a simple function/formula to determine some sort of output. Let us say

$$\{p_n|n = p_1, p_2, p_2, \dots\} \tag{2.1}$$

where p_n is our parameters.

General Equations we are taught in life

$$f(p_n) = p_n + rules \tag{2.2}$$

$$f(\{p_n|n = p_1, p_2, p_2, \dots\}) = \{p_n|n = p_1, p_2, p_2, \dots\} + rules \tag{2.3}$$

In Layman's Equation Terms
Input + rules = output

(Input is put through the rules returning some form of output)

In normal programming

$$\{p_n|n = p_1, p_2, p_2, \dots\} + rules =? \tag{2.4}$$

In Layman's Equation Terms
Input + rules = output

Our question mark represents (output, but depends on input and/or rules)

In Machine Learning

Let us define some machine learning equation let's say if we are using the KNN algorithm (more about that later in this paper)

$$f(p_n) = f_{KNN}(p_n) \tag{2.5}$$

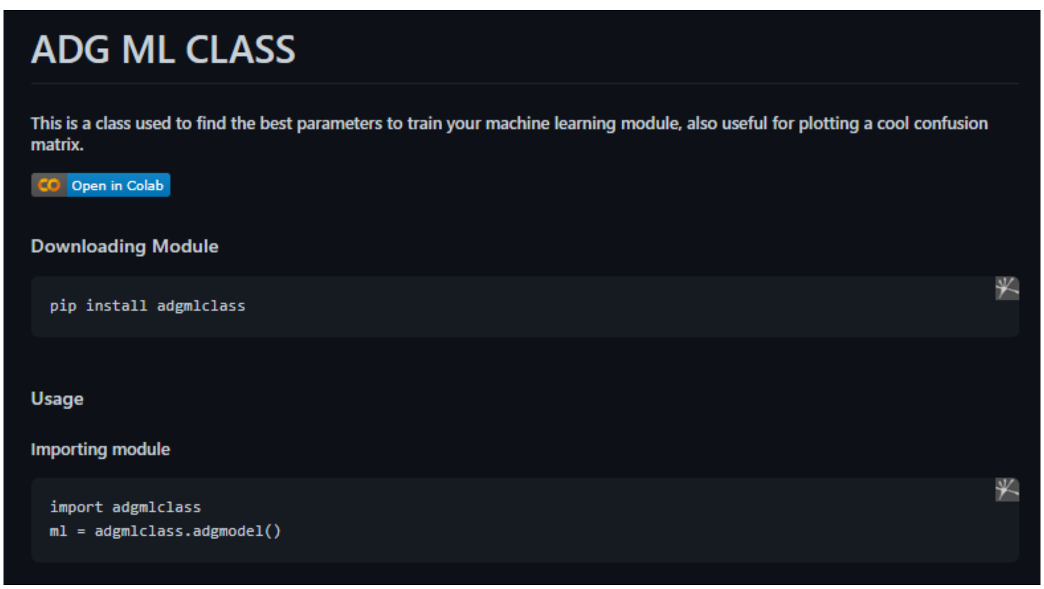
$$f(\{p_n | n = p_1, p_2, p_2, \ldots\}) = f_{KNN}(\{p_n | n = p_1, p_2, p_2, \ldots\}) \tag{2.6}$$

In Layman's Equation Terms

Input + ? = output

In this case the ? means the rules (program makes rules based on input and expected output)

I have built a Python Module to find the best parameters to trains your machine learning module. It is called adgmlclass which is on PyPi and code is open source on my GitHub. You could also set the rules to a generic machine learning algorithm, modify the algorithm with fine-tuned Machine Learning parameters.



ADG ML CLASS

This is a class used to find the best parameters to train your machine learning module, also useful for plotting a cool confusion matrix.

Open in Colab

Downloading Module

```
pip install adgmlclass
```

Usage

Importing module

```
import adgmlclass
ml = adgmlclass.adgmodel()
```

Figure 2.3: ADGMLCLASS : https://github.com/adgsenpai/adgmlclass

2.2.2 What is Machine Learning?

Machine learning, also known as automatic learning, is a branch of science and a subcategory of artificial intelligence. It entails allowing algorithms to find "patterns" in data sets, i.e., recurring patterns. This information can take the form of numbers, words, images, statistics, and so on. Machine Learning can use anything that can be stored digitally as data. Algorithms learn and improve their performance in performing a specific task by detecting patterns in this data.

2.2.3 Which Mathematical Concepts Are Used in Machine Learning and Data Science?

Statistics, Linear Algebra, Probability, and Calculus are the four key concepts that drive machine learning. While statistical concepts are at the heart of all models, calculus aids in the learning and optimization of those models.

2.2.4 Importance of Machine Learning

Machine learning is important because it allows businesses to see trends/predictions in customer behavior and business operational patterns while also assisting in the development of new products. Machine learning is at the heart of many of today's most successful businesses, including Facebook, Google, and Uber. In this paper, the importance of Machine Learning is that we are using it to predict and forecast data based on our input data which is the parameters to determine an output value on an output parameter.

2.2.5 Machine Learning Types, Algorithms, and Usage

Types of Machine Learning:

Supervised Learning
Supervised learning is a type of machine learning in which machine learning from known datasets (set of training examples), and then predict the output. A supervised learning agent needs to find out the function that matches a given sample set. Supervised learning further can be classified into two categories of algorithms:

1. Classifications

2. Regression

Unsupervised Learning
Unsupervised learning is associated with learning without supervision or training. In unsupervised learning, the algorithms are trained with data which is neither labeled nor classified. In unsupervised learning, the agent needs to learn from patterns without corresponding output values. Unsupervised learning can be classified into two categories of algorithms:

1. Clustering
2. Association

Reinforcement Learning
Reinforcement learning is a type of learning in which an AI agent is trained by giving some commands, and on each action, an agent gets a reward as feedback. Using these feedbacks, agent improves its performance. Reward feedback can be positive or negative which means on each good action, agent receives a positive reward while for wrong action, it gets a negative reward. Reinforcement learning is of two types:

1. Positive Reinforcement learning
2. Negative Reinforcement learning

List of Common Machine Learning Algorithms

- **Linear Regression – (will explain the algorithm in paper)**
- Logistic Regression
- Decision Tree
- SVM
- Naive Bayes
- **KNN – (will explain the algorithm in paper)**
- K-Means
- Random Forest

- Dimensionality Reduction Algorithms
- Gradient Boosting algorithms
- Autoencoder
- Convolutional Neural Network
- Recurrent Neural Network
- Support Vector Machine
- Deep Learning

Deep learning is another complicated topic that I won't go into much detail about in this paper; instead, I'll give you some general information about it. Deep learning is a subset of a larger family of machine learning techniques based on representation learning and artificial neural networks. There are three types of learning: supervised, semi-supervised, and unsupervised. For image classification, object detection, image restoration, and image segmentation, deep learning has delivered superhuman accuracy—even handwritten digits can be recognized. Deep learning, which makes use of massive neural networks, is teaching machines to automate the tasks that human visual systems perform. Deep learning is a machine learning and artificial intelligence (AI) technique that mimics how humans acquire knowledge.

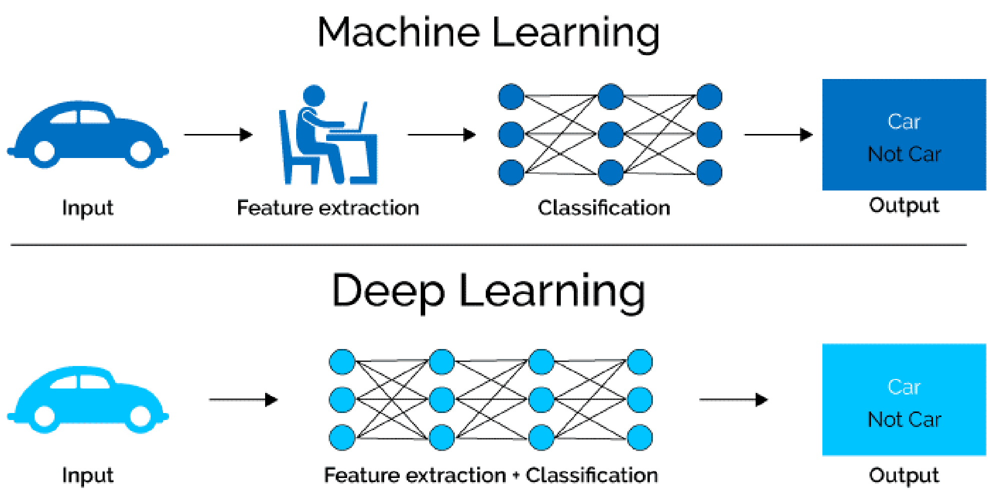


Figure 2.4: Deep Learning General Structure Diagram

2.2.6 Linear Regression

Introduction to Linear Regression

A linear approach to modeling the relationship between a scalar response and one or more explanatory variables is known as linear regression in statistics. Simple linear regression is used when there is only one explanatory variable; multiple linear regression is used when there is more than one. Linear regression is a supervised machine-learning regression algorithm.
A typical linear regression equation that you learned in high school is in the form of

$$y = b_0 + b_1 x \tag{2.7}$$

where y is the response variable, b_0 is the intercept, and b_1 is the coefficient.

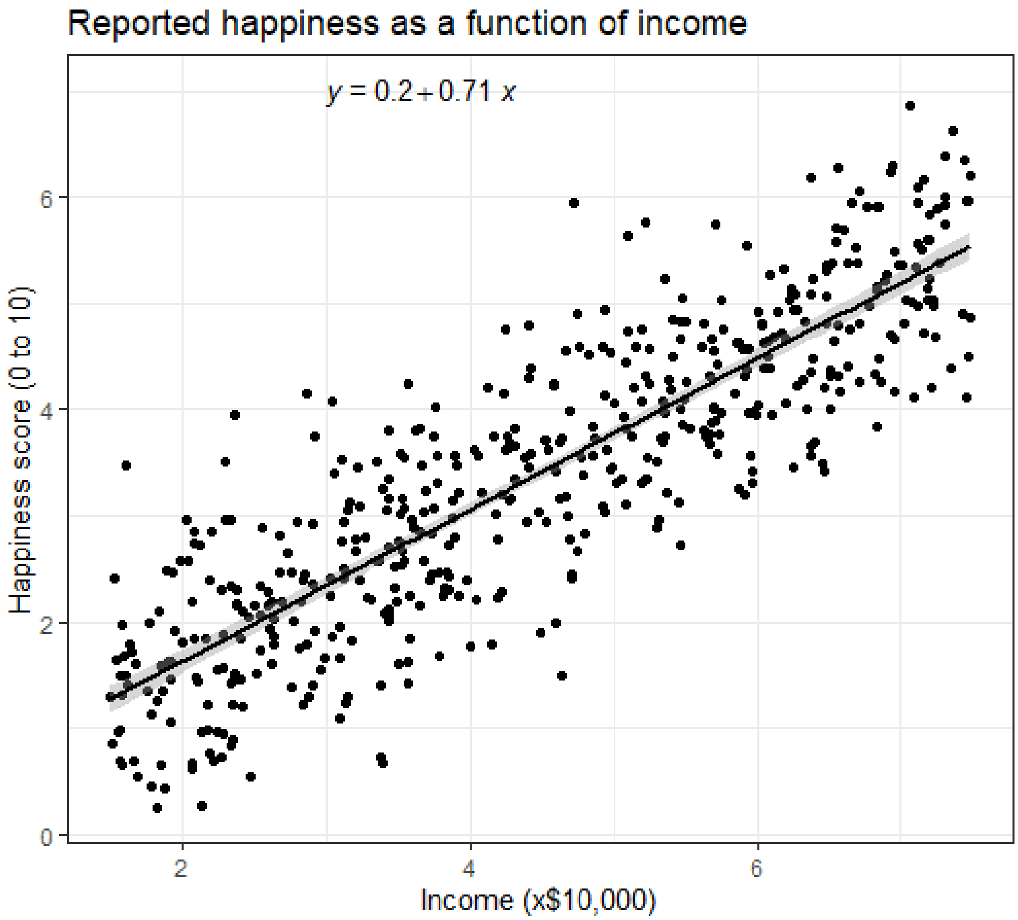


Figure 2.5: Linear Regression function of Reported Happiness by income values

In this example, employees are interviewed and are asked to fill in the form below. Rate your happiness in life from

$$y \in \{0..10\} \tag{2.8}$$

We capture our data and represent it in a digital format or record it on a page. The data scientist visualizes the data in a scatter graph. To make predictions on this data he calculates the regression equation which is

$$y = 0.2 + 0.71x \tag{2.9}$$

and plots the function on the scatter graph. If you look at the diagram above that is exactly what is being done to predict data.

Simple Prediction and Flaw with Linear Regression

Let us predict my happiness score based on this model.

where x is the income in 10k
and y is the happiness score.

$$y = 0.2 + 0.71(100) \tag{2.10}$$

Our predicted score is **71.2**. That is not possible as the scale can only be $y \in \{0..10\}$ You can see that the model above is invalid and inaccurate. The training size is too small. We could get a better model if we use a larger population. The point I'm trying to make with Linear Regression is that you will get false alerts and reading based on the input data. I would also say emotions with people are not accounted for when working with Machine Learning models. For example, some people who are multi-millionaires are depressed with life. Some people with low income are happy in life they appreciate the simple things in life. Machine Learning does not account for emotion there is another aspect in AI that can help predict emotions and that is called Natural Language Processing. It works with text and speech. Deep learning powers that technology.

Other types of Linear Regression

What I showed you in the introduction was a **simple linear regression** equation which is

$$y = b_0 + b_1 x \tag{2.11}$$

where y is the response variable, b_0 is the intercept, and b_1 is the coefficient.

A simple graph of that is

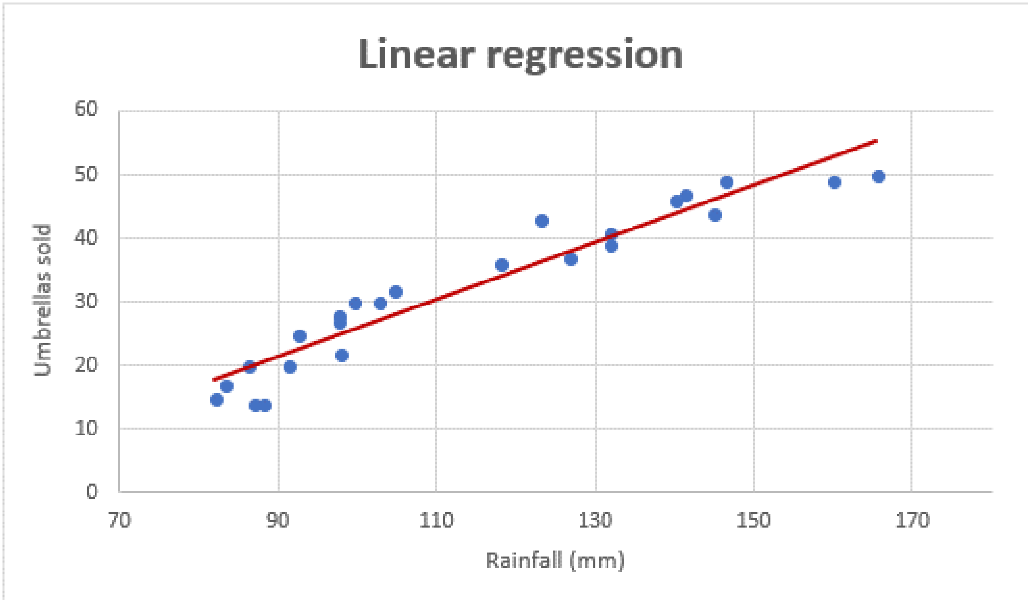


Figure 2.6: Simple Linear Regression

The second type of linear regression is called **Multiple Linear Regression**

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2^2 + \cdots + \beta_n x^n \tag{2.12}$$

where y is the response variable, β_0 is the intercept, and β_1 is the coefficient for x_1, β_2 is the coefficient for x_2^2, and so on.

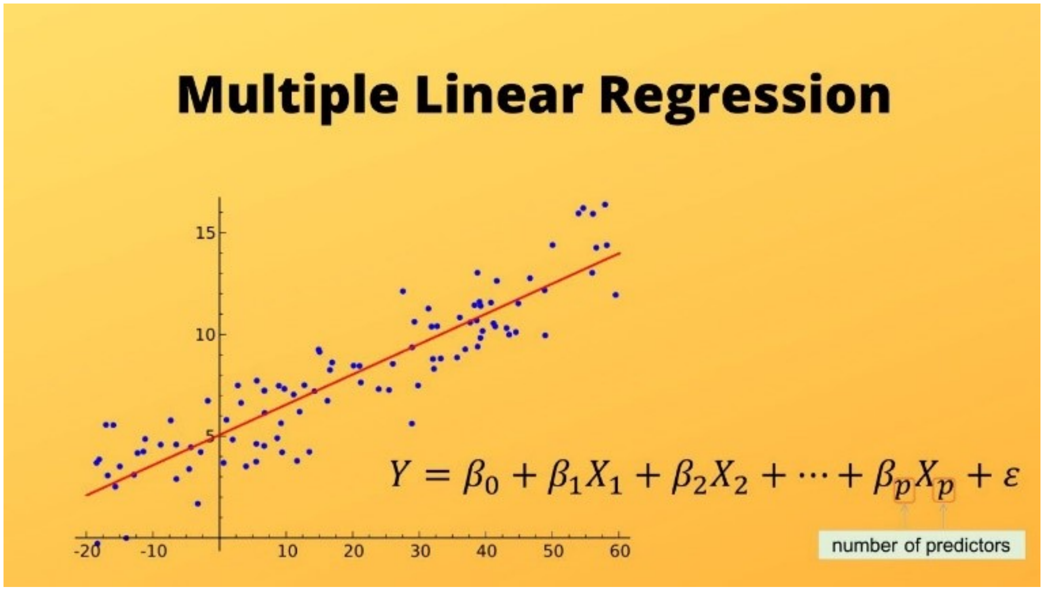


Figure 2.7: Multiple Linear Regression

A graph for multiple linear regression looks similar to the simple linear regression above.

The last type of linear regression is called **Polynomial Linear Regression**

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \cdots + \beta_n x_1^n \tag{2.13}$$

where y is the response variable, β_0 is the intercept, and β_1 is the coefficient for x_1, β_2 is the coefficient for x_1^2, and so on.

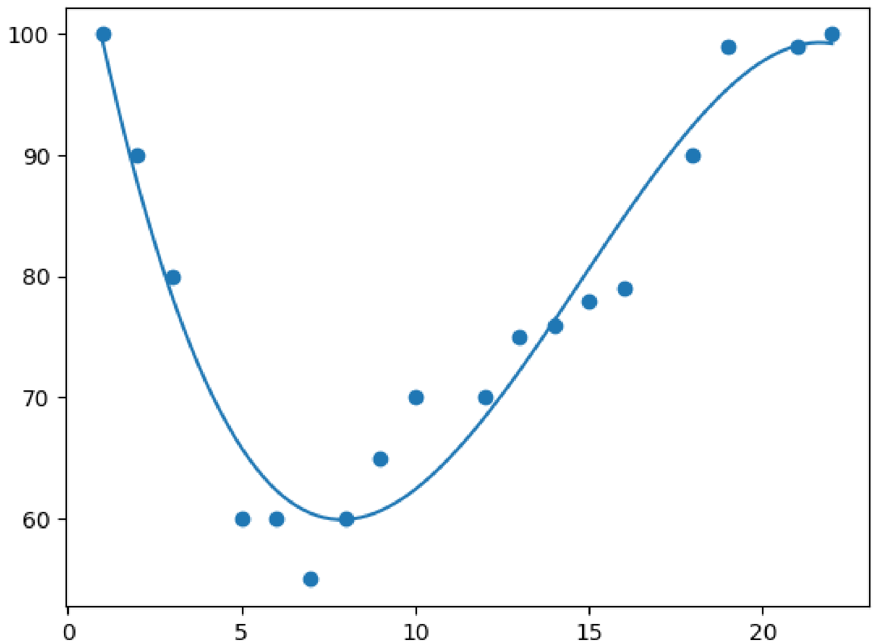


Figure 2.8: Multiple Linear Regression

2.2.7 K-Nearest Neighbors

Introduction to K-Nearest Neighbors

K-nearest neighbors (k-NN) is a supervised machine learning algorithm that can be used to solve both classification and regression tasks. Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.
Consider the below diagram:

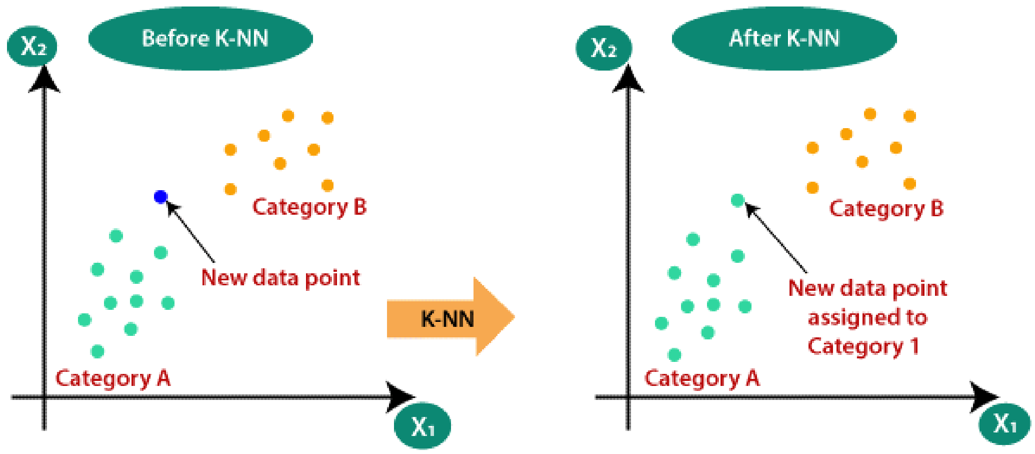


Figure 2.9: K-Nearest Neighbors

The K-NN algorithm is a supervised machine learning algorithm that can be used to solve both classification and regression tasks.

How does the k-nearest neighbors algorithm work?

1. Select the number K of the Neighbors
2. Calculate the Euclidean distance of K number of neighbors
3. Take the K nearest neighbors as per the calculated Euclidean distance.
4. Among these k neighbors, count the number of the data points in each category.
5. Assign the new data points to that category for which the number of the neighbour is maximum.

Suppose we have a new data point and we need to put it in the required category.

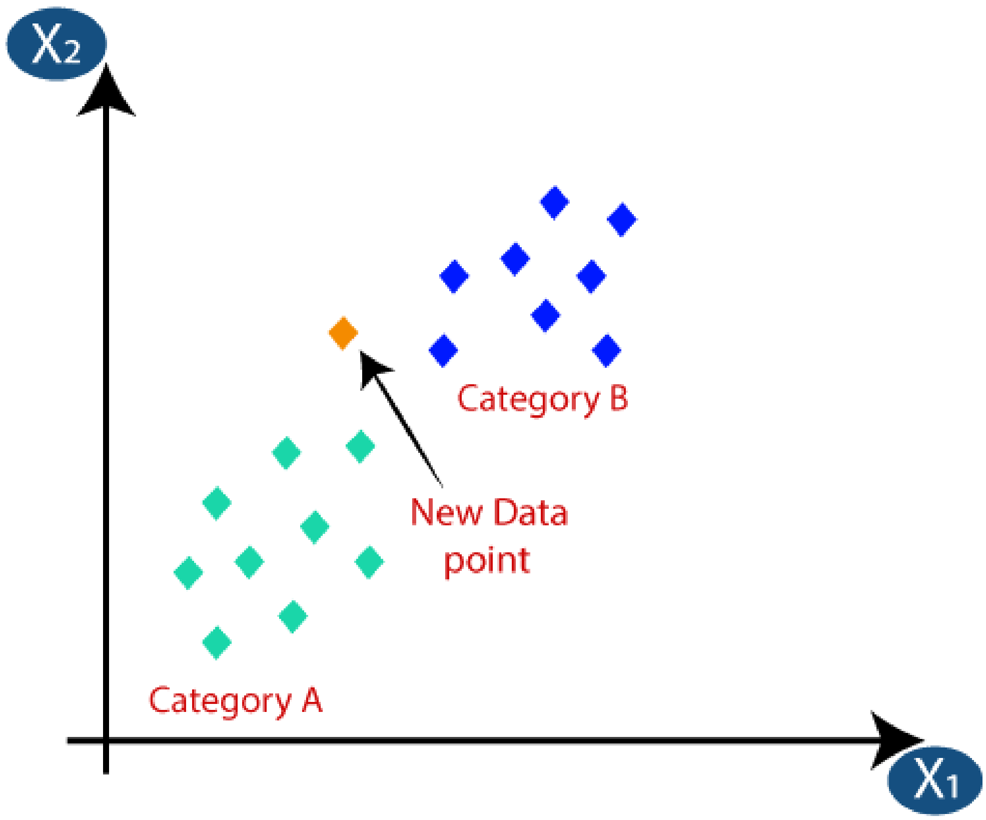


Figure 2.10: K-Nearest Neighbors Datapoint Graph

- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the Euclidean distance (which we learnt in high school) between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:

$$D_{Euclidean} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2.14}$$

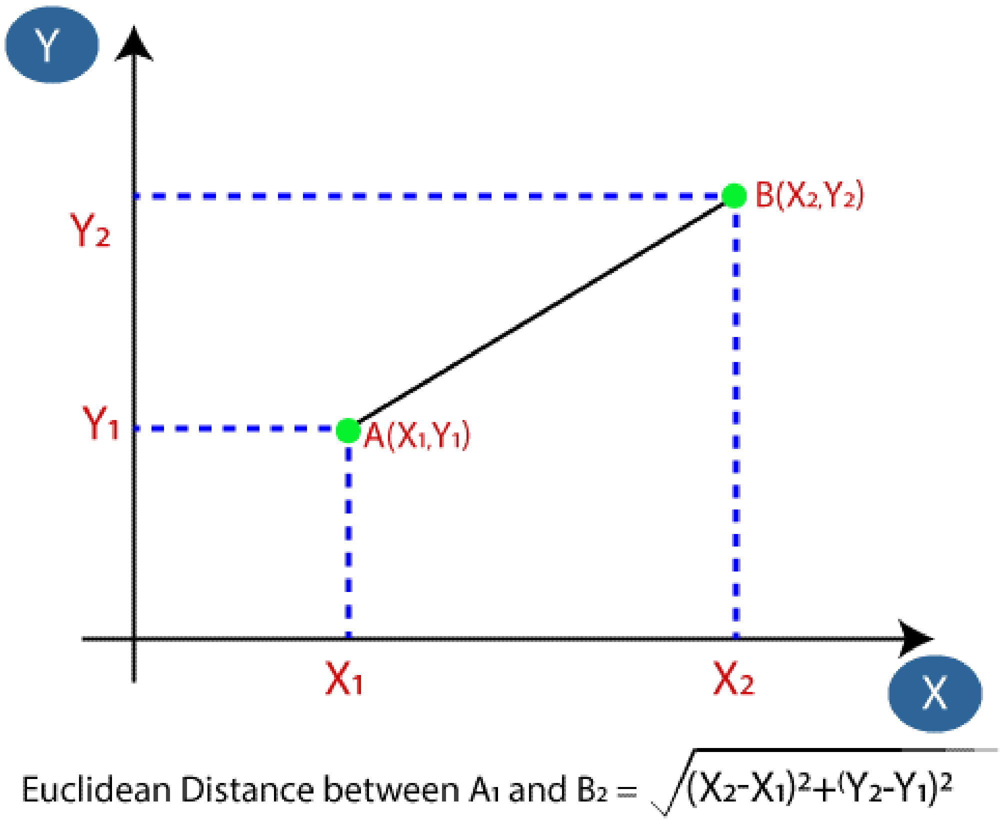


Figure 2.11: Calculating Euclidean distance between two data points

By calculating the Euclidean distance, we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:

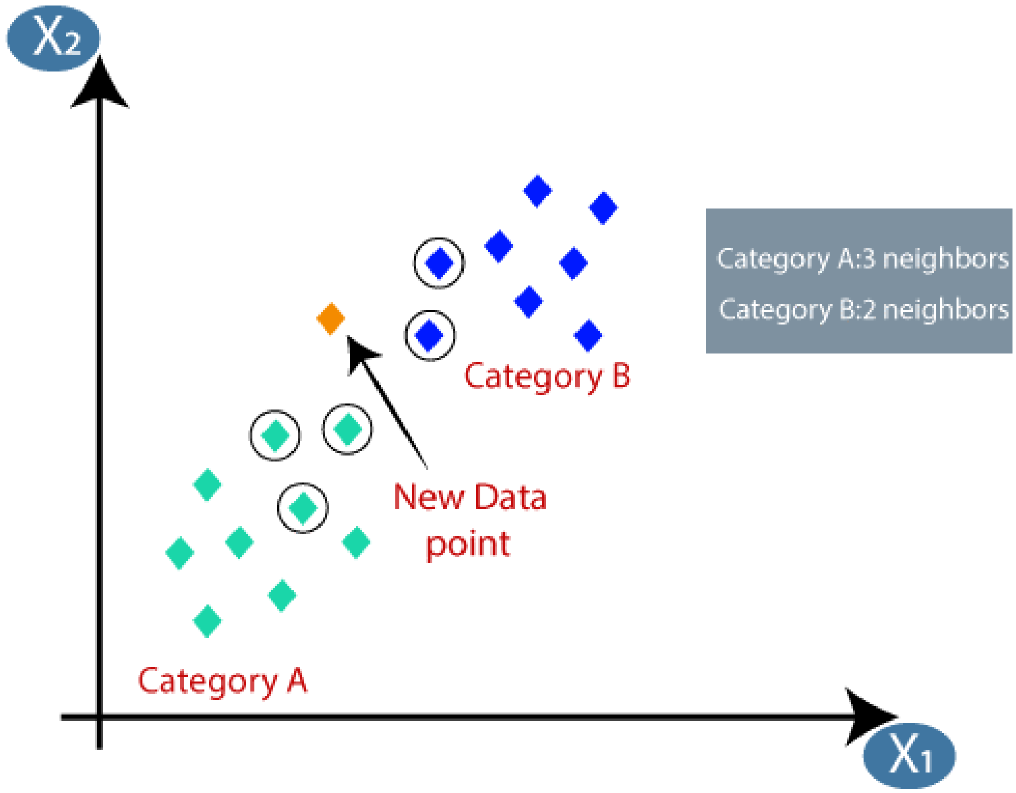


Figure 2.12: Nearest Neighbors

As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Advantages of KNN Algorithm

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

Distance functions

For KNN we can use different distance functions to achieve better accuracy or different K values.

$$D_{Euclidean} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2.15}$$

$$D_{Manhattan} = |x_1 - x_2| + |y_1 - y_2| \tag{2.16}$$

$$D_{Minkowski} = \left\{[((x_1 - x_2)^p)^q]^r\right\}^{1/r} \tag{2.17}$$

2.2.8 Measuring of Performance of Models

We can use functions in our machine learning libraries to measure performance of our model.

Scoring	Function	Comment
Classification		
'accuracy'	metrics.accuracy_score	
'balanced_accuracy'	metrics.balanced_accuracy_score	
'top_k_accuracy'	metrics.top_k_accuracy_score	
'average_precision'	metrics.average_precision_score	
'neg_brier_score'	metrics.brier_score_loss	
'f1'	metrics.f1_score	for binary targets
'f1_micro'	metrics.f1_score	micro-averaged
'f1_macro'	metrics.f1_score	macro-averaged
'f1_weighted'	metrics.f1_score	weighted average
'f1_samples'	metrics.f1_score	by multilabel sample
'neg_log_loss'	metrics.log_loss	requires `predict_proba` support
'precision' etc.	metrics.precision_score	suffixes apply as with 'f1'
'recall' etc.	metrics.recall_score	suffixes apply as with 'f1'
'jaccard' etc.	metrics.jaccard_score	suffixes apply as with 'f1'
'roc_auc'	metrics.roc_auc_score	
'roc_auc_ovr'	metrics.roc_auc_score	
'roc_auc_ovo'	metrics.roc_auc_score	
'roc_auc_ovr_weighted'	metrics.roc_auc_score	
'roc_auc_ovo_weighted'	metrics.roc_auc_score	
Clustering		
'adjusted_mutual_info_score'	metrics.adjusted_mutual_info_score	
'adjusted_rand_score'	metrics.adjusted_rand_score	
'completeness_score'	metrics.completeness_score	
'fowlkes_mallows_score'	metrics.fowlkes_mallows_score	
'homogeneity_score'	metrics.homogeneity_score	
'mutual_info_score'	metrics.mutual_info_score	
'normalized_mutual_info_score'	metrics.normalized_mutual_info_score	
'rand_score'	metrics.rand_score	
'v_measure_score'	metrics.v_measure_score	
Regression		
'explained_variance'	metrics.explained_variance_score	
'max_error'	metrics.max_error	
'neg_mean_absolute_error'	metrics.mean_absolute_error	
'neg_mean_squared_error'	metrics.mean_squared_error	

Figure 2.13: Performance measurement functions in Python Scikit-Learn Module

above is a photo of performance metrics functions for the module scikit-learn in Python if you research these function names you can view formulas on these tools.

Check out the following link for more information on performance metrics: `https://scikit-learn.org/stable/modules/model_evaluation.html`

Real life example of model evaluation

Below is performance metrics on a model I improved on a project of Galaxy Machine Learning.

```
print(confusion_matrix(y_test, predictions))
print(classification_report(y_test, predictions))
```

```
[[42  1]
 [ 3 44]]
              precision    recall  f1-score   support

           0       0.93      0.98      0.95        43
           1       0.98      0.94      0.96        47

    accuracy                           0.96        90
   macro avg       0.96      0.96      0.96        90
weighted avg       0.96      0.96      0.96        90
```

Figure 2.14: Performance metrics on a model

For the model above you can see it has extremely good accuracy of around 96% accuracy Anything above 90% I consider good enough there was times I even got 100% accuracy for a model.
Let's check out how our F1-Score is calculated.

$$F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.18}$$

alternatively, we can use the following formula:

$$F1\text{-}Score = \frac{2TP}{2TP + FP + FN} \tag{2.19}$$

where TP is the number of true positives, FP is the number of false positives, FN is the number of false negatives.

What does TP mean?
That means True Positives in our dataset. Which our correct data in our dataset which tested true in our data.
What does FP mean?
That means False Positives in our dataset. Which our incorrect data in our dataset which tested false in our data.
What does FN mean?

That means False Negatives in our dataset. Which our incorrect data in our dataset which tested true in our data.

Another tool we can use to showcase our TP, FP, FN is called a **confusion matrix**

Mathematically, a confusion matrix looks like this:

$$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix} \tag{2.20}$$

Graphically could look like this below

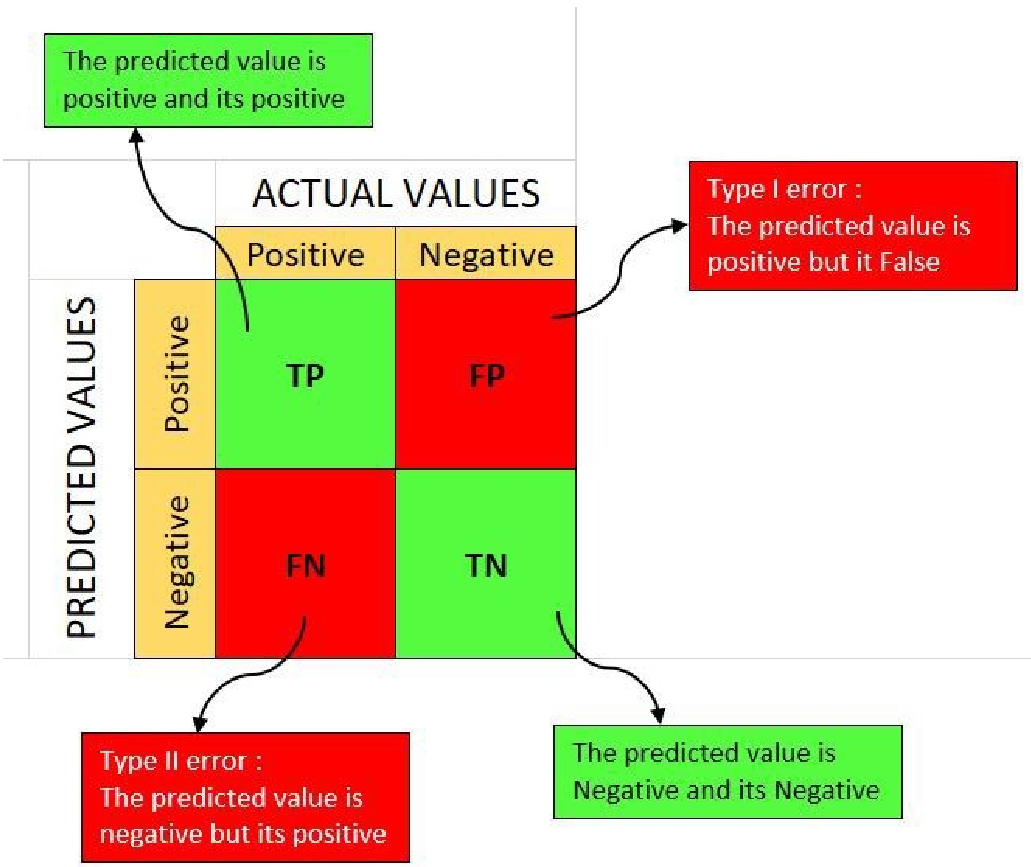


Figure 2.15: Confusion Matrix

Another graphical representation using the ADGMLCLASS library is shown below:

```
Test - Accuracy : 0.8607888631090487

Test - classification report :
              precision    recall  f1-score   support

           0       0.84      0.90      0.87       431
           1       0.89      0.83      0.86       431

    accuracy                           0.86       862
   macro avg       0.86      0.86      0.86       862
weighted avg       0.86      0.86      0.86       862
```

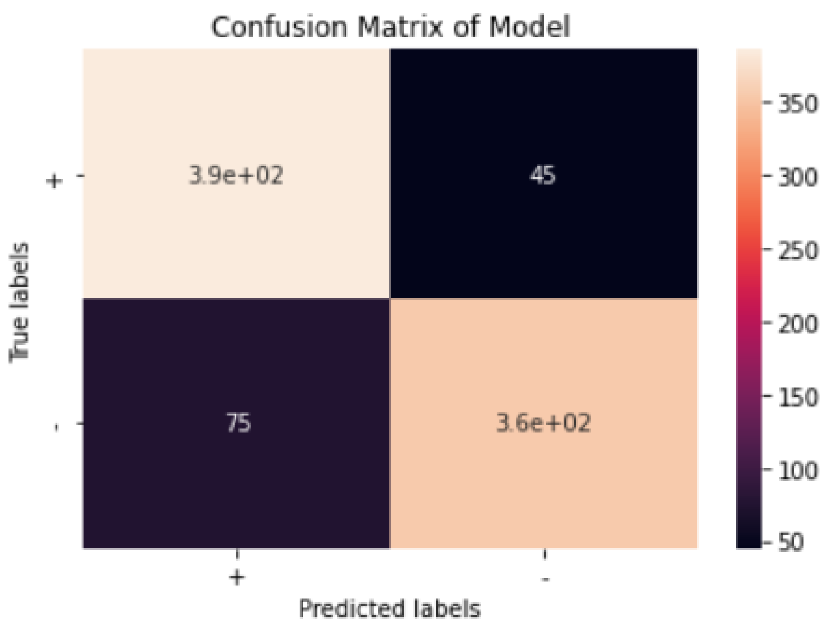
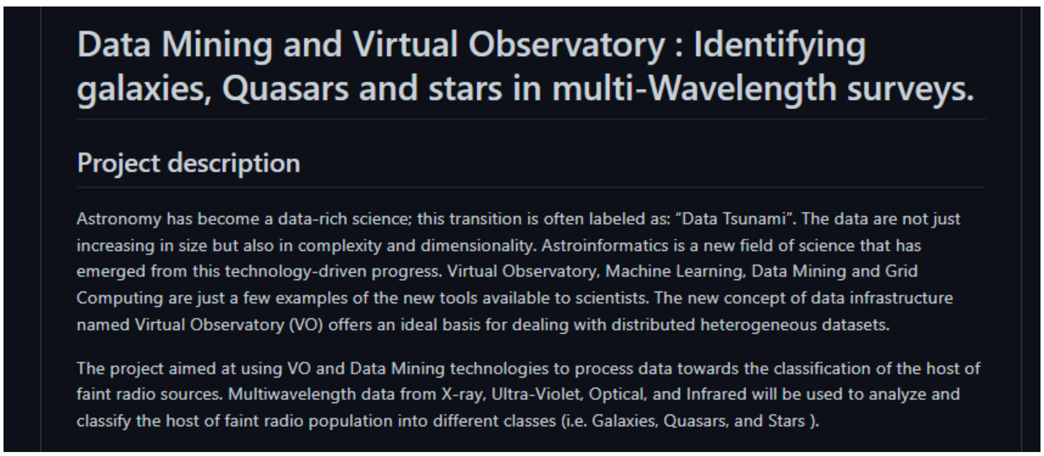


Figure 2.16: Real life Confusion Matrix of a model

2.2.9 Playing with Machine Learning Demos + Real World Examples I did

GalaxyML

You can play with GalaxyML. I helped a masters student optimize the ML Solution using adgmlclass and put the repo because this is a real-world example of ML in action.



Data Mining and Virtual Observatory : Identifying galaxies, Quasars and stars in multi-Wavelength surveys.

Project description

Astronomy has become a data-rich science; this transition is often labeled as: “Data Tsunami”. The data are not just increasing in size but also in complexity and dimensionality. Astroinformatics is a new field of science that has emerged from this technology-driven progress. Virtual Observatory, Machine Learning, Data Mining and Grid Computing are just a few examples of the new tools available to scientists. The new concept of data infrastructure named Virtual Observatory (VO) offers an ideal basis for dealing with distributed heterogeneous datasets.

The project aimed at using VO and Data Mining technologies to process data towards the classification of the host of faint radio sources. Multiwavelength data from X-ray, Ultra-Violet, Optical, and Infrared will be used to analyze and classify the host of faint radio population into different classes (i.e. Galaxies, Quasars, and Stars).

Figure 2.17: GalaxyML Repo

`https://github.com/ADGSTUDIOS/GalaxyML`

Neural Networks

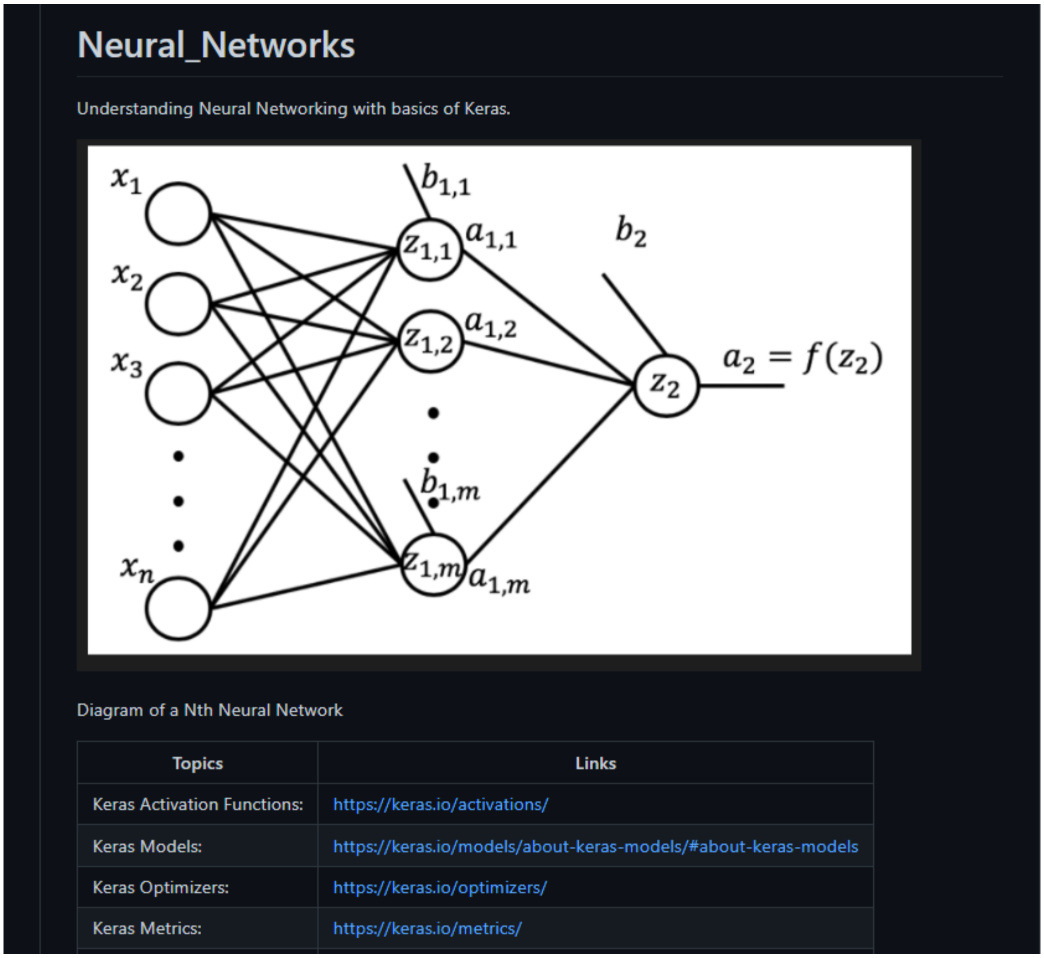


Figure 2.18: Neural Networks Repo

If you want to play with Neural Networks got great notebooks to play with @ `https://github.com/adgsenpai/Neural_Networks`

Machine Learning Application in ECommerce

Clothing Size Guide AI Application used for predicting clothing size given a users BUST(CM) and HIPS(CM)

Hannah Grace

SA SIZE	BUST(CM)	HIPS(CM)	WAIST(CM)	TARGET
28	79-81	88-91	61-63	XXS
30	84-86	94-96	66-68	XS
32	89-91	99-101	71-73	S
34	94-96	104-106	76-78	M
36	99-101	109-111	81-83	L
38	104-106	114-116	86-88	XL
40	109-111	119-121	91-93	2XL
42	114-116	124-126	96-98	3XL
44	119-121	129-131	101-103	4XL
46	123-126	134-136	106-108	5XL
48	129-132	139-141	111-113	6XL
50	135-138	144-146	116-118	7XL

Use the Size Guide AI Application to determine the size of clothing you can use

BUST(CM)

BUST(CM)

HIPS(CM)

HIPS(CM)

Figure 2.19: Hannah Grace Clothing Size AI Application

you can play with it @ https://www.hannahgrace.co.za/size-guide/ this app has an accuracy of 100%