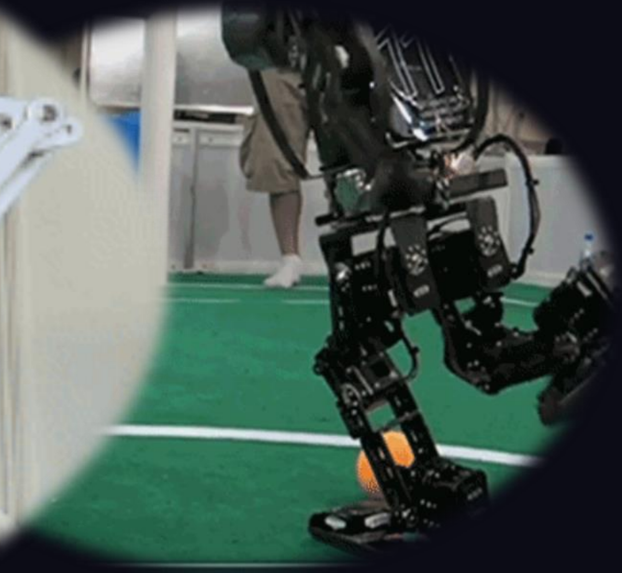
Topic - 01
Introduction to Machine Learning: Concept & Fundamentals
Part: 01

+88 0172 6867 984

tanvir@socian.ai

What is Artificial Intelligence?



In Reality

Any artificial device or mechanism that is capable of self learning is in reality an Artificial Intelligent being and the technology that empowers that Artificial being is Artificial Intelligence

What is Machine Learning?

Machine Learning is the science (and art) of programming computers so they can learn from data.

Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.

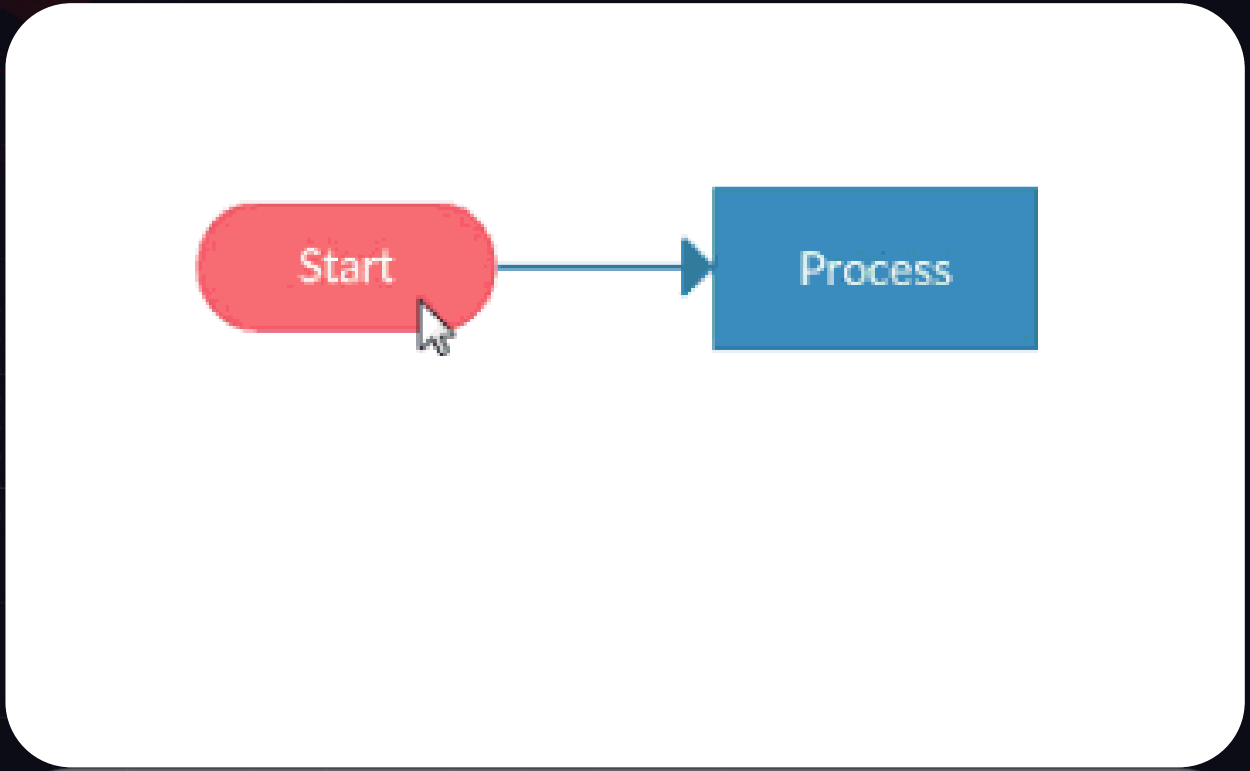
Examples:
Virtual Assistant (Siri, Alexa, Cortana)
Traffic Prediction (Google)
Face Recognition (Facebook)
Emails (Spam and not Spam)

Definition (Scientific)

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

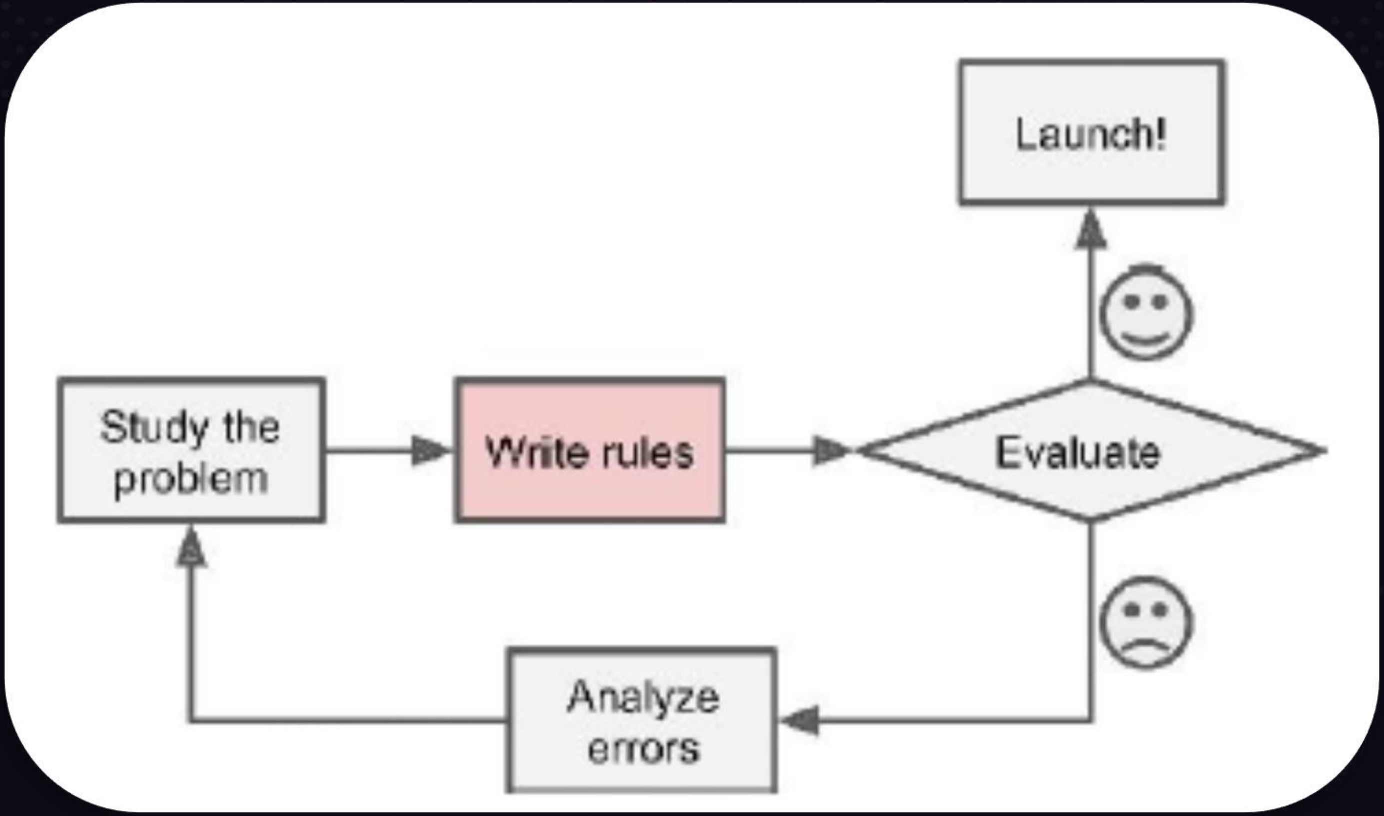
Machine Learning is all about Probability.

Probability of something happening with lots of conditions!!!



Machine Learning Example – Spam Filter

We get lots of Spam emails from time to time from various sources. If we had to write a Spam Filter program with traditional conditional method, the program would look like the following:



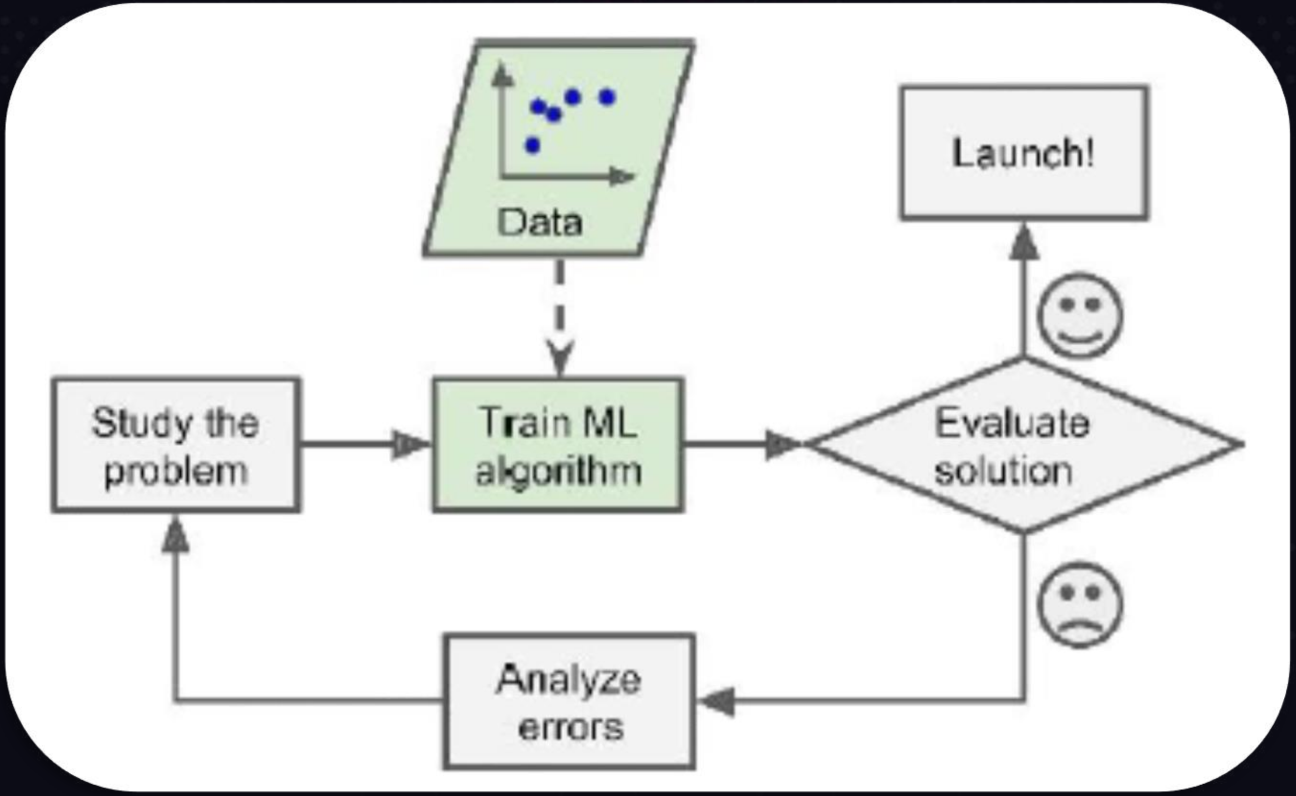
Spam Filter – Conditional Approach

- First we will create a list of words or phrases that are commonly available in a Spam Email.
  For example: Offer, Bonus, Free Money, 4U, Lottery, Credit Card, Amazing, Gift, Won etc.
- Then we will create a login when these words are found, we will call an email SPAM, otherwise HAM (Not SPAM)
- Make lots of modifications in conditions and keep repeating the above 2 processes.

But the problem with the Conditional Approach is that, it will contain lots of rules and you will have to change rules each and every time a new kind of SPAM email comes with different word variations. **For example**: Instead of **4U**, using **For You**

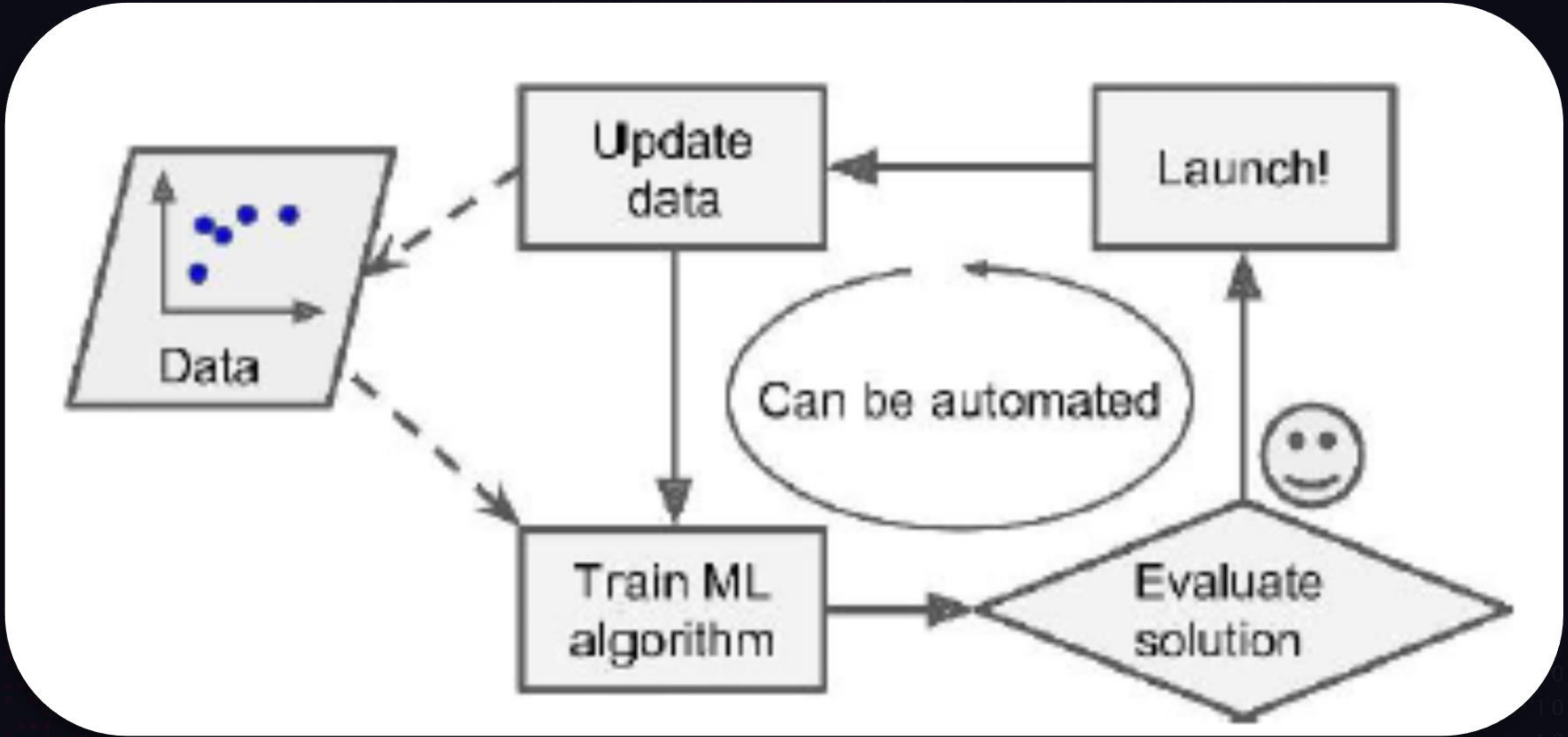
Spam Filter – Machine Learning Approach

Spam Filter based on Machine Learning can learn from all the marked emails (SPAM & HAM) and when a new email comes, it can automatically detect which email is SPAM and which email is HAM.



Spam Filter – Machine Learning Approach Cont.

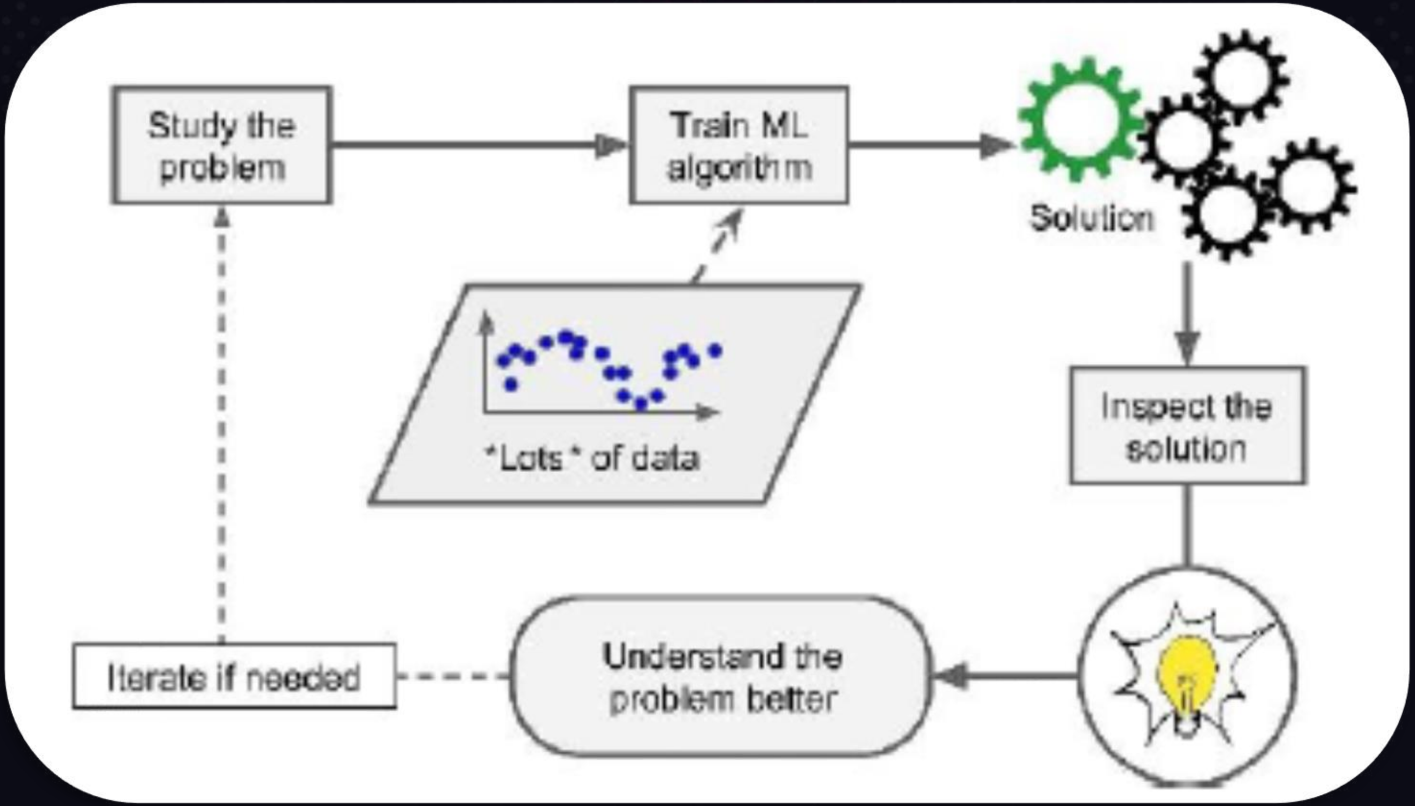
And even though some email type was not marked SPAM earlier, if they are later on Marked SPAM from several users, it can also learn from there! You don't have to write new rules for new SPAMs!!! ☺



SOCIAN

Machine Learning can Help Humans Learn

As Machine Learning can Flag emails for SPAM, it can help us to also find the words or patterns for which SPAM is happening! Applying Machine learning in data can help finding NEW Patterns!



Difficult Problems with Machine Learning

In case of problems, where there is no available algorithm or are too difficult for traditional approach to overcome, Machine Learning can be applied to solve those problems!

Example 1: Speech Recognition, where the whole thing is measured by Voice Spectrogram (ছয় and নয় – have some similarities but the Pitch gets lot intensified in case of ছয়) and different people speak in different way in different situations

Example 2: Generating Automated Answers from Questions. Even though human understands which question is what, for Machine to understand that is very difficult – আমার ব্যালেন্স কত, আমার বালান্স, বালান্স কত, balance koto, blance koto, balance kto, belance koto, valance khoto, vlance koto – lot many variations can be obtained for the same problem

SOCIAN

What Machine Learning is Good for

- Solving problems where you need to write lots of rules – less code, less time and better performance
- Solving complex problems where there is no good solution available
- Getting adapted to new kind of data – it can be known data, it can be unknown data
- Getting a good idea on problems and measure the data that are being used
- Getting insights about problems
- Solving problems which were impossible before

SOCIAN

Classification – based on Data

Based on Data Supervision, Machine Learning can be divided into:

Supervised Learning

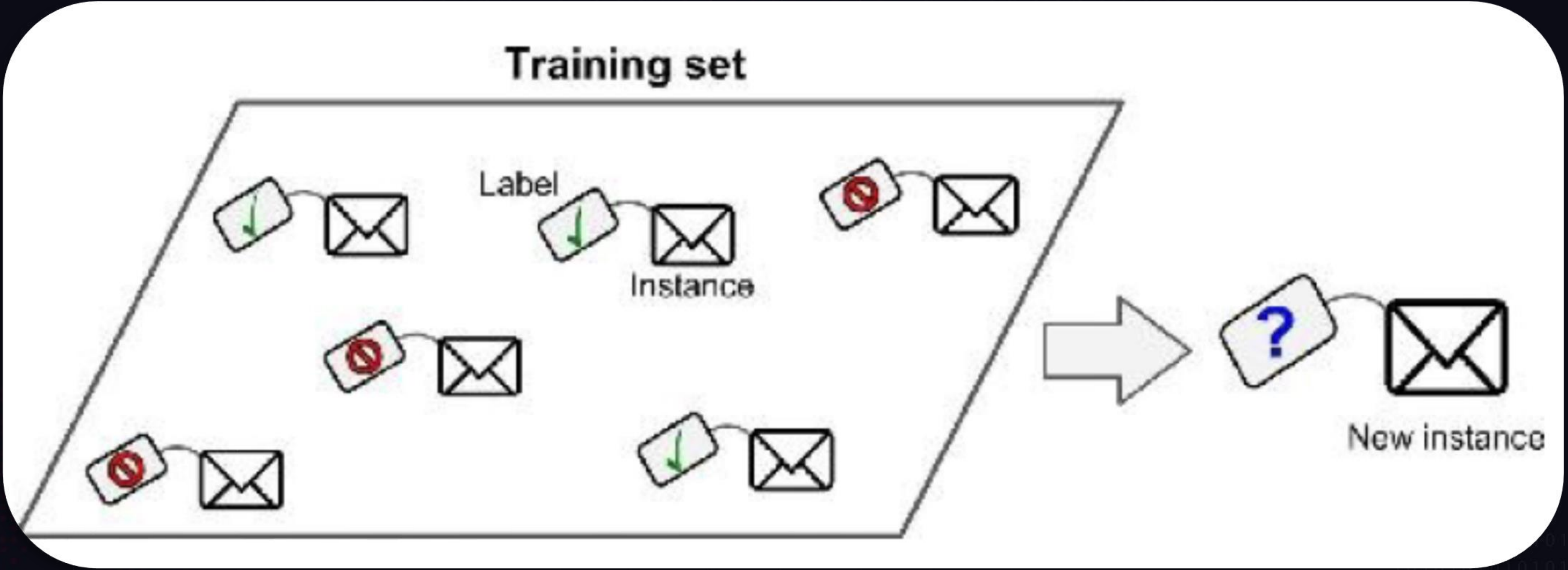
Unsupervised Learning

Semi-supervised Learning

Reinforcement Learning

Supervised Learnings

In case of supervised learning, the solution of the problem is also given to the algorithm in the time of Training. Organizing data with their desired value is called “Labelling” and the data as “Labelled Data”



Supervised Learnings - Examples

Let’s say we want to measure the Price of houses in an area based on: Number of rooms, House rent, Age, Neighborhood and few other matrices. In order to apply the Supervised Learning process, we will give lots of House samples (data) with all the matrices as well as the Price as input and then Train the System (Training Data).

After successful training, we can test our Accuracy with new Data samples (Testing Data/Validation Set).

Based on the data and everything, we can measure how good our Supervised Model is performing.

Among all the Machine Learning Approaches based on Data, Supervised Learning is widely used.

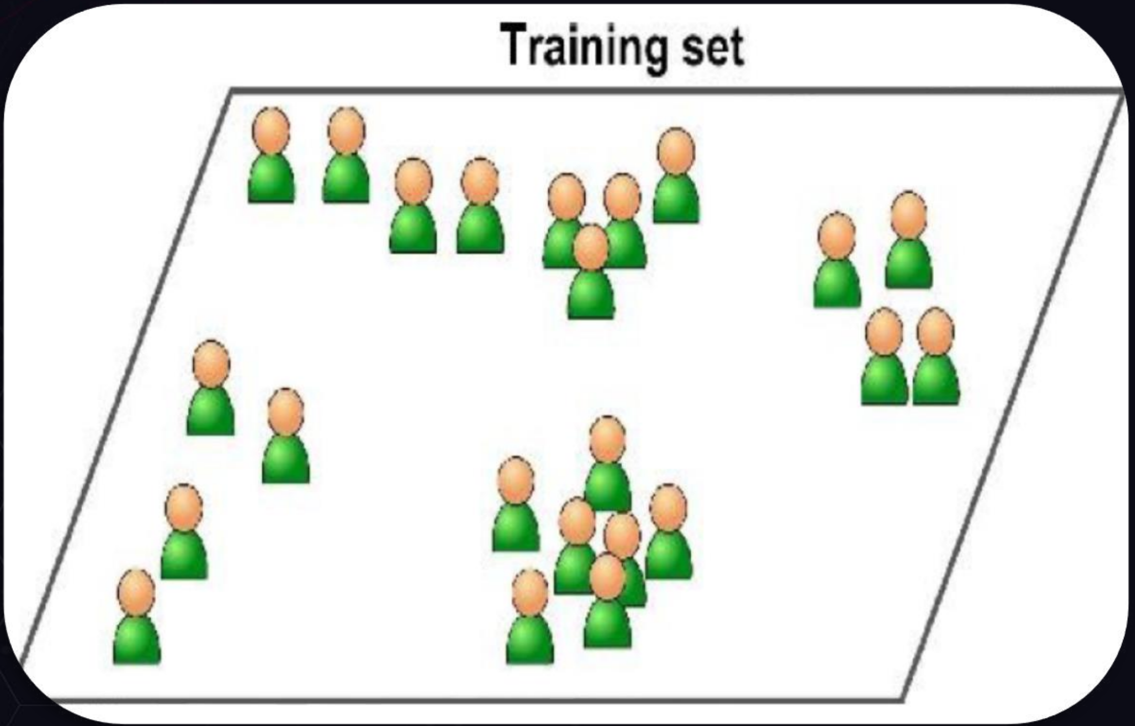
Widely Used Supervised Learning Algorithms

- k-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVMs)
- Decision Trees
- Random Forests
- Neural networks

Unsupervised Learnings

In case of unsupervised learning, the solution of the problem is **NOT** given to the algorithm in the time of Training. As a result, no pre-defined labelled data is given into the System. The system finds pattern from BIG Data available and tries to solve the problem.

Human Example: You have joined into a library but you don't know anything about Library and book structures. Now, your supervisor has given you the task of sorting the books and putting them properly in the shelf. Now, you need to learn the right way of doing that.



Widely Used Unsupervised Learning Algorithms

- Clustering based:
  - ✓ k-Means
  - ✓ Hierarchical Cluster Analysis (HCA)
  - ✓ Expectation Maximization
- Visualization and dimensionality reduction
  - ✓ Principal Component Analysis (PCA)
  - ✓ Kernel PCA
  - ✓ Locally-Linear Embedding (LLE)
  - ✓ t-distributed Stochastic Neighbor Embedding (t-SNE)
- Association rule learning
  - ✓ Apriori
  - ✓ Eclat

Self Task

- Watch Movie:
  - ✓ HER (by 19.01.2019)
- Watch TV Series
  - ✓ Silicon Valley (No time frame)

Topic - 01
Introduction to Machine Learning: Concept & Fundamentals
Part: 02

+88 0172 6867 984

tanvir@socian.ai

SOCIAN

Attributes vs Features

In Machine Learning an ***ATTRIBUTE*** is a data type (e.g., “Price”), while a ***FEATURE*** has several meanings depending on the context, but generally means an attribute plus its value (e.g., “Price = 85,50,000”).

Many people use the words attribute and feature to represent the same thing.

Dimensionality Reduction

In many cases, we have to reduce our data without losing information in order to make the training faster. Easiest way of it is merging several related data as one.

For example, a House Price is connected with the Age of the House. So, we merge those 2 together and get a new House-Price feature. This process is called **FEATURE EXTRACTION**.

We will always try to achieve Dimensionality Reduction before starting any Machine Learning process (For example, any supervised process).

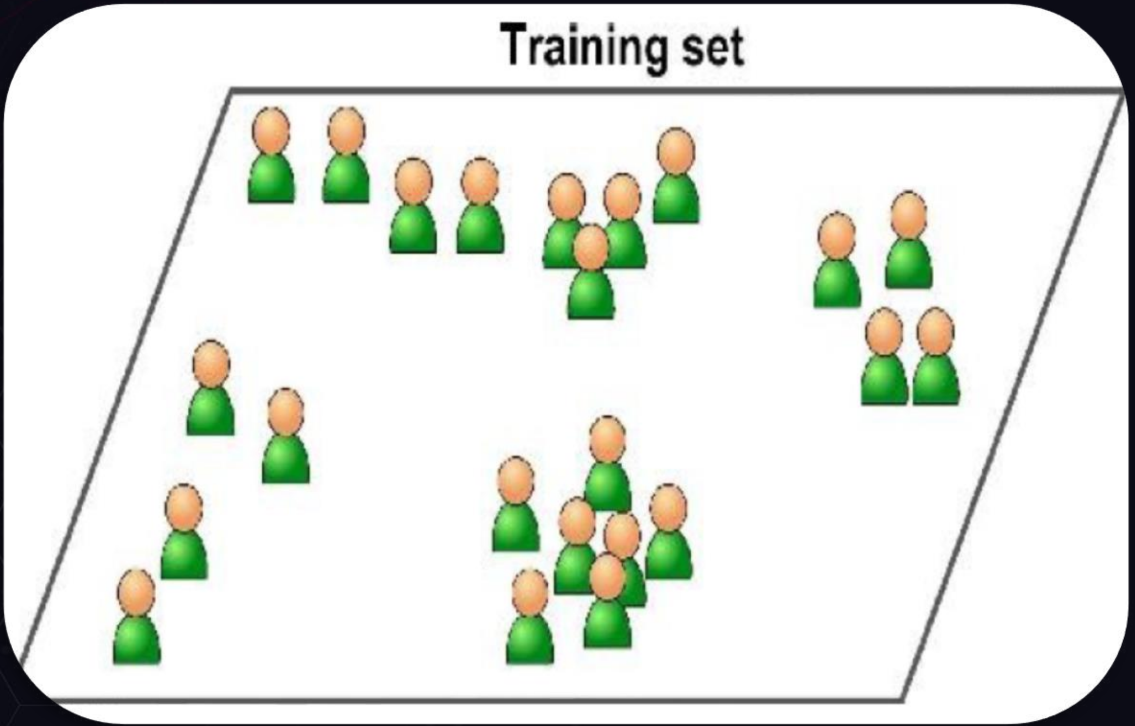
It will run much faster, the data will take up less disk and memory space, and in some cases it may also perform better.

Unsupervised Learnings - Examples

Let's say you want to categorize all of your Blog visitors into different groups of similar attributes. This process is called ***Clustering***. We don't give any input to the algorithm.

It tries to find the connection among the data and give output.

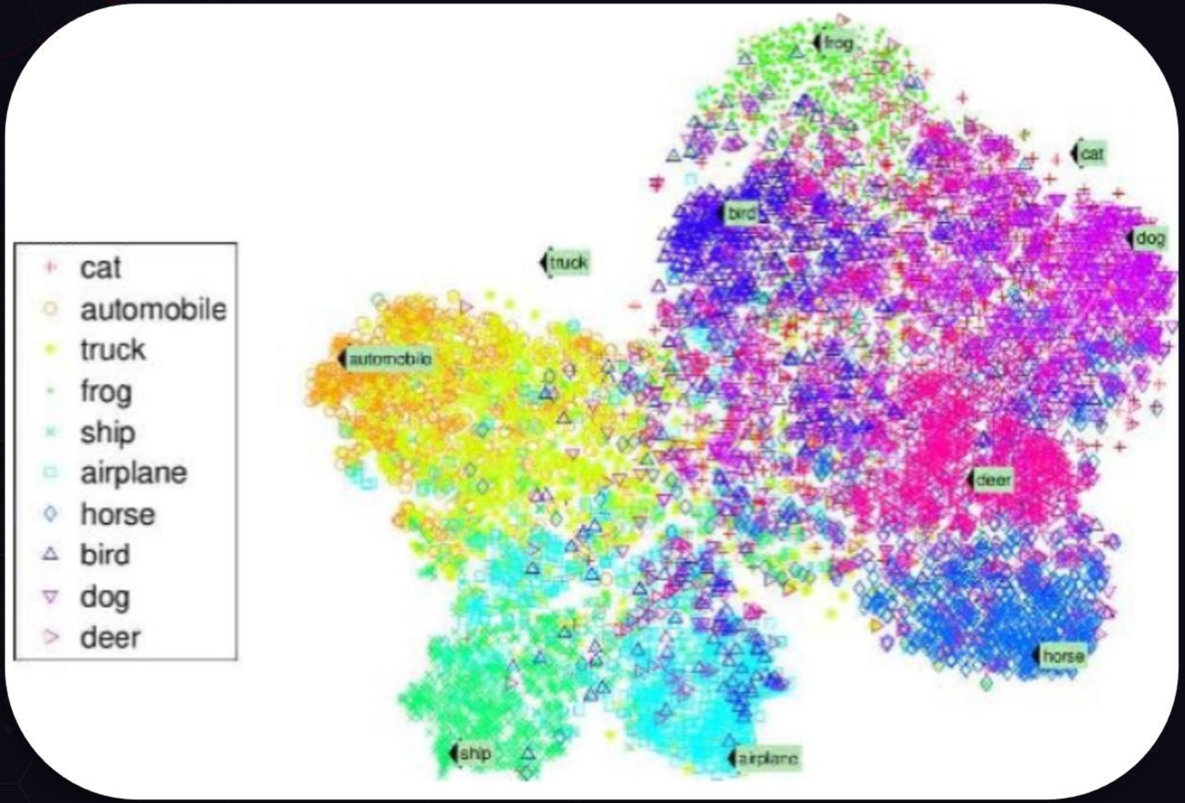
For example: You may find out that 30% of your blog visitors are female who love TRAVEL blogs in night, whilst 80% of your blog visitors are male who love SPORTS blog.



Unsupervised Learnings - Examples Cont.

A good example of Unsupervised Learning is Visualization Algorithm. You feed them UNLABELLED complex data and they will generate pattern which will help you to distinguish them. That way, both 2D and 3D representation can be obtained which can easily be plotted.

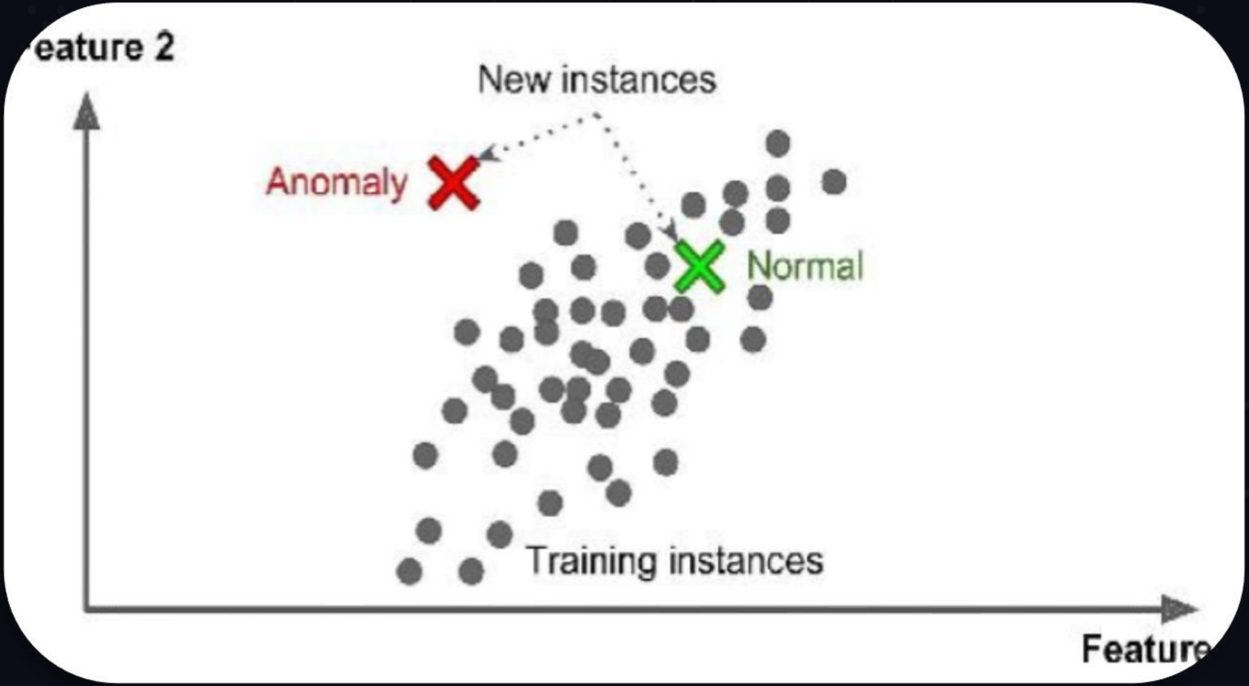
These algorithms keep as much structure
as possible so you can understand how the data is organized and identify unexpected patterns



Unsupervised Learnings - Examples Cont..

Anomaly Detection is another good task of unsupervised learning.

For example: Finding credit card fraud or catching manufacturing defects. First, the system is trained with all the normal instances. When a new instance comes, it can tell whether it is a normal instance or an anomaly.



SOCIAN

Association Rule Learning

A common unsupervised task is ***Association Rule Learning***. There we go thoroughly into large amount of data and find relationship between ATTRIBUTES.

For example, you own a Super Shop. Running an Association Rule Learning into your Sales Log may help you find out that people who buy Barbecue Sauce and Potato Chips also buy Steak most of the time.

So, re-organizing your Shop and keeping these items together will help you increase your sales many times.

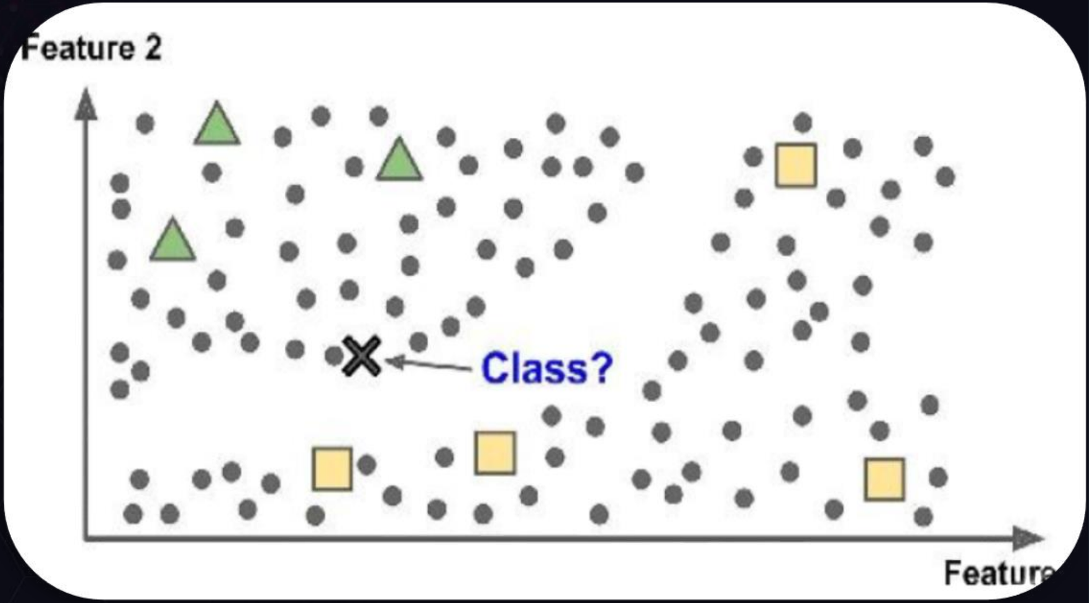
SOCIAN

Semi-supervised Learnings

There are some algorithms out there which can deal with training data which are Partially labelled (only little of the data is labeled, rest are unlabeled). This process is called Semi-supervised learning.

For example, in Google photos, if you upload a lot of photos, it will first identify how many people are available in all the photos and giving their Thumbnail to identify. This is an unsupervised process.

Now, if you just TAG with NAME, who is who, then it can identify all the people in all the photos. Now, you can easily search people with their name.



Deep Belief Networks (DBNs)

Most Semi-supervised learning algorithms are combinations of Unsupervised Algorithms and Supervised Algorithms.

For example, Deep Belief Networks (DBNs) are based on unsupervised components called Restricted Boltzmann Machines (RBMs) stacked on top of one another.

RBMs are trained sequentially in an unsupervised manner, and then the whole system is fine-tuned using supervised learning techniques.

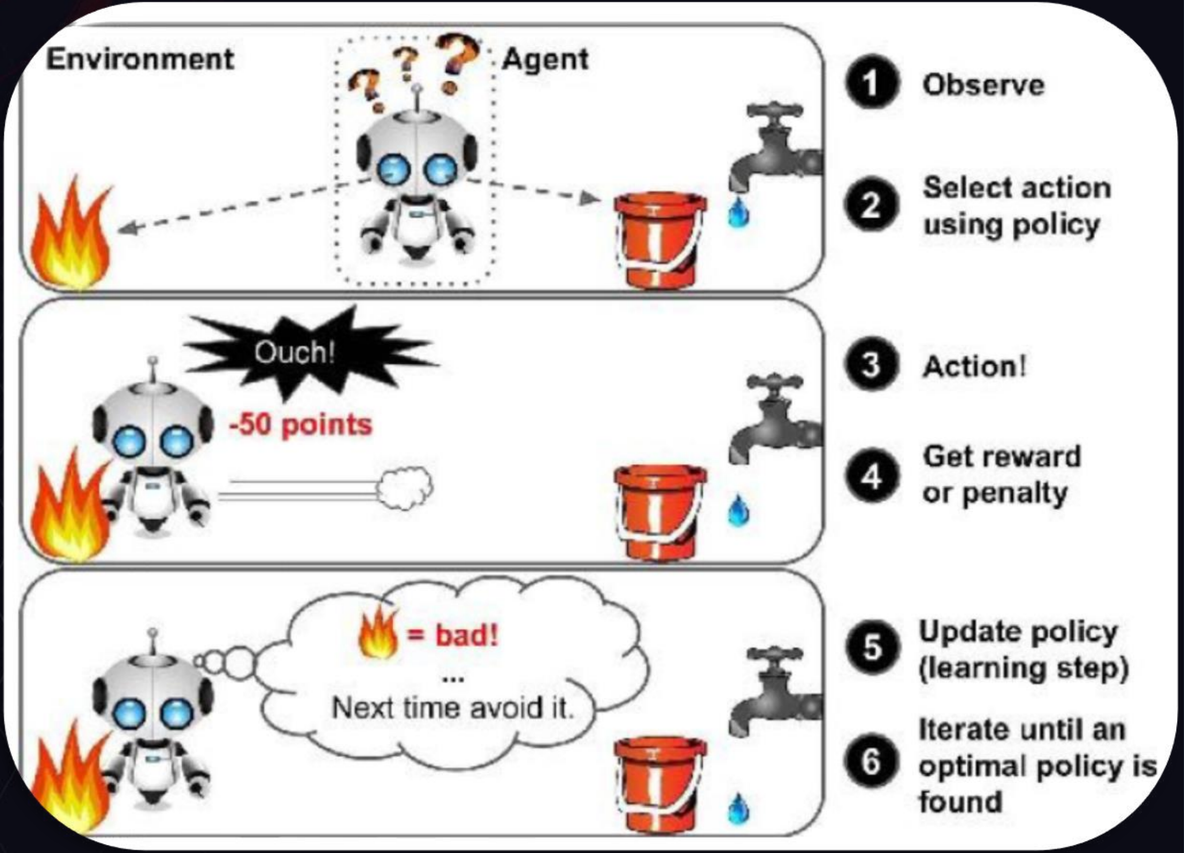
Reinforcement Learning

This is a totally different kind of Machine Learning process. In this case, an ***AGENT*** is appointed.

AGENT can –

- Observe the environment
- Select and perform actions
- Get rewards in return (or penalties in the form of negative rewards).

It must then learn by itself what is the best strategy, called a ***POLICY***, to get the most reward over time. A POLICY defines what action the agent should choose when it is in a given situation.


Environment
Agent
Ouch!
-50 points
= bad!
...
Next time avoid it.
1 Observe
2 Select action using policy
3 Action!
4 Get reward or penalty
5 Update policy (learning step)
6 Iterate until an optimal policy is found

Reinforce Learning – Example

Many robots implement Reinforcement Learning algorithms to learn how to walk.

DeepMind’s AlphaGo program is also a good example of Reinforcement Learning. It made the headlines in March 2016 when it beat the world champion Lee Sedol at the game of Go. ***GO*** is considered the hardest mind games in the whole world with Infinite possibilities.

It learned its winning policy by –
Analyzing millions of games, and then playing many games Against Itself.

Note that learning was turned off during the games against the champion; AlphaGo was just applying the policy it had learned.

Classification – based on Incremental Learning

Based on whether or not the system can learn ***INCREMENTALLY*** from a stream of incoming data, Machine Learning is divided into:

Batch Learning

Online Learning

Batch Learning

In batch learning, the system is incapable of learning incrementally. It must be trained using all the available data. This will generally take a lot of time and computing resources, so it is typically done offline.

First the system is trained, and then it is launched into production and runs without learning anymore; it just applies what it has learned. This is called OFFLINE LEARNING.

If you want a batch learning system to know about new data (such as a new type of spam), you need to train a new version of the system from scratch on the full dataset (not just the new data, but also the old data), then stop the old system and replace it with the new one.

Topic - 01
Introduction to Machine Learning: Concept & Fundamentals
Part: 03

+88 0172 6867 984

tanvir@socian.ai

SOCIAN

Advantages & Disadvantages of Batch Learning

Advantages:

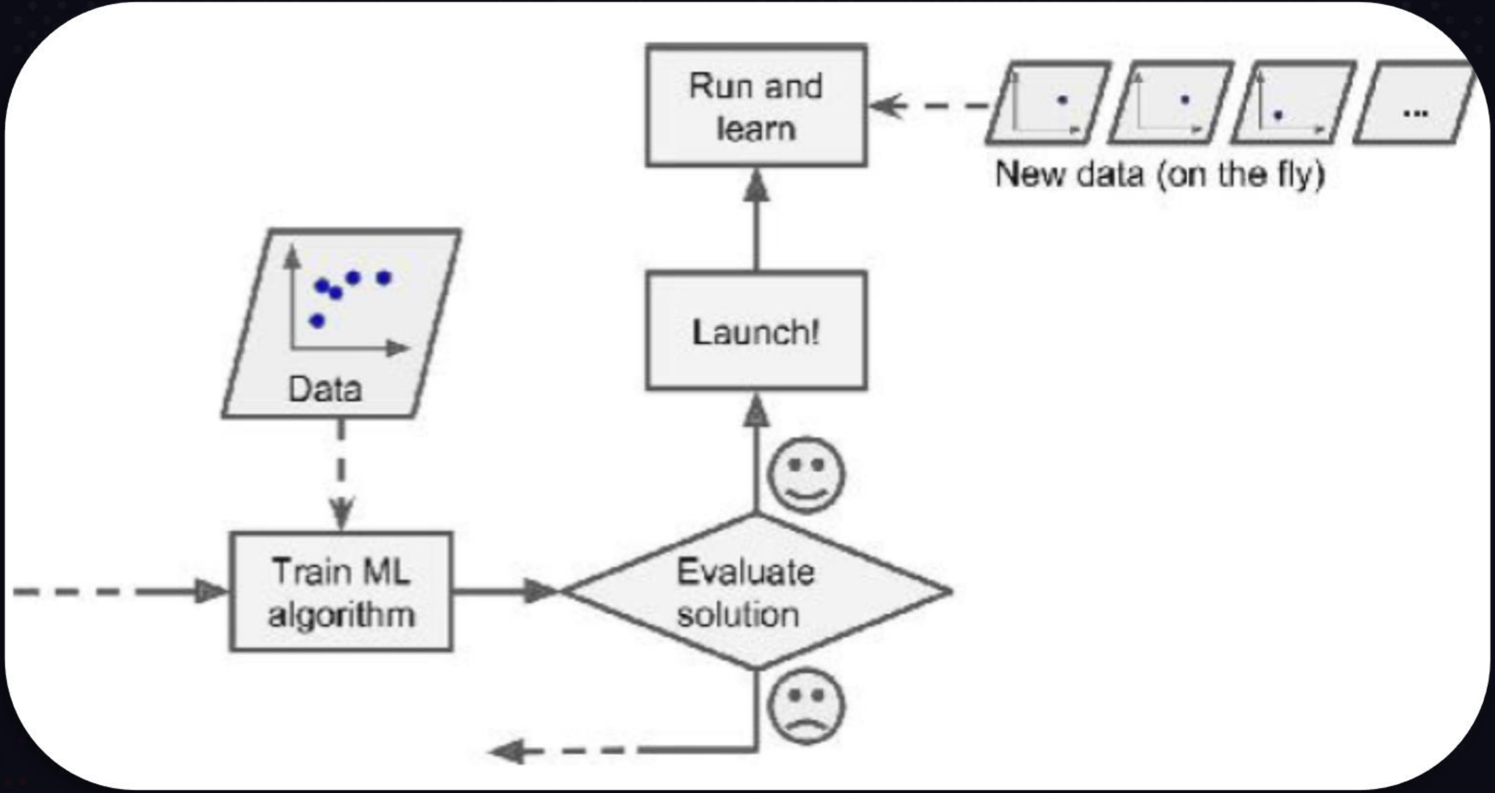
- If your data doesn't have continuous update, then Batch Learning is the best solution
- Easy to maintain and the old model can be replaced with the new one very easily (with just replacing the old file).
- Works good for a generic problem. For example: Speech Recognition

Disadvantages:

- This solution is simple and often works fine, but training using the full set of data can take many hours. Doesn't work well if your data needs continuous update (24 hours). For example: Stock Price Prediction.
- Training on the full set of data requires a lot of computing resources (CPU, memory space, disk space, disk I/O, network I/O)
- If the amount of data is huge, it may even be impossible to use a batch learning algorithm (If the training takes several days to finish)
- If your system needs to be able to learn autonomously and it has limited resources (e.g., a smartphone app), then carrying around large amounts of training data and taking up a lot of resources to train for hours every day is quite impossible.

Online Learning

In online learning, you train the system incrementally by feeding it data instances sequentially, either individually or by small groups called mini-batches. Each learning step is fast and cheap, so the system can learn about new data on the fly, as it arrives.



Online Learning Cont.

Online learning is great for systems that receive data as a continuous flow (e.g., stock prices) and need to adapt to change rapidly or autonomously.

It is also a good option if you have limited computing resources: once an online learning system has learned about new data instances, it does not need them anymore, so you can discard them (unless you want to be able to roll back to a previous state and "replay" the data).

This can save a huge amount of space.

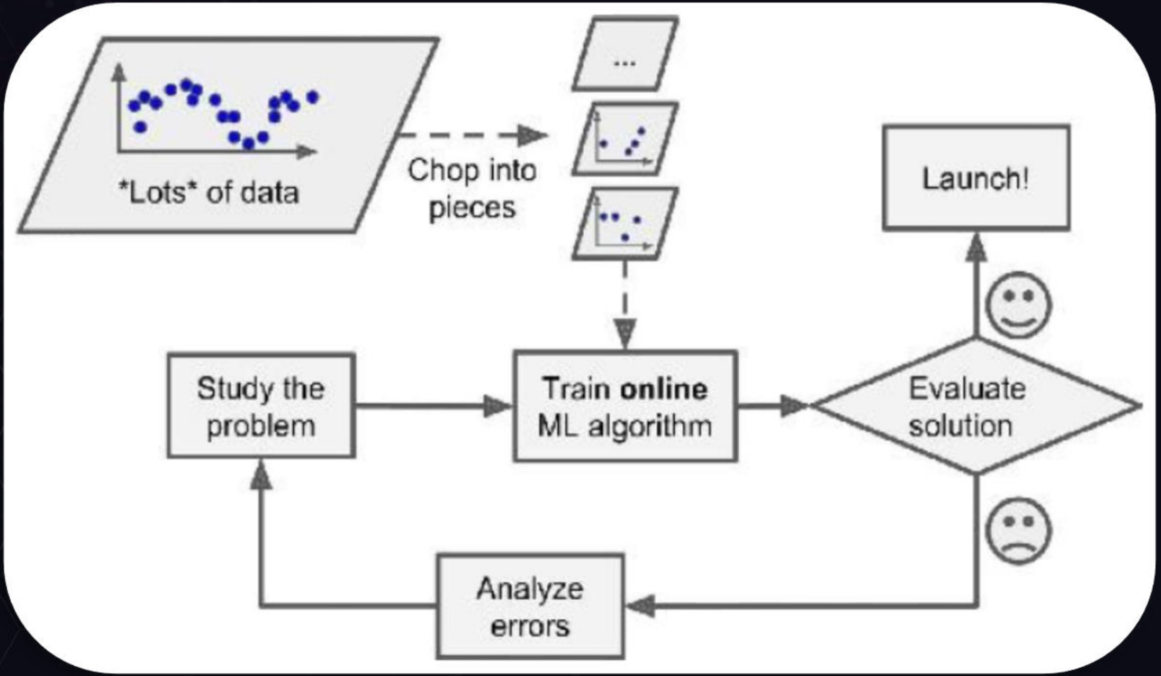
Online learning algorithms can also be used to train systems on huge datasets that cannot fit in one machine's main memory (this is called ***OUT-OF-CORE*** learning). The algorithm loads part of the data, runs a training step on that data, and repeats the process until it has run on all of the data.

Online Learning Cont..

One important parameter of online learning is how fast they can adapt to changing data: this is called the ***LEARNING RATE***.

If you set a high learning rate, then your system will rapidly adapt to new data, but it will also tend to quickly forget the old data (you don't want a spam filter to flag only the latest kinds of spam it was shown).

Conversely, if you set a low learning rate, the system will have more inertia; that is, it will learn more slowly, but it will also be less sensitive to noise in the new data or to sequences of nonrepresentative data.



Online Learning is Actually OFFLINE

Even though the name of the Process is ONLINE LEARNING, the process is usually done ***OFFLINE*** (i.e., not on the live system).

So online learning can be a confusing name.

We can alternatively call it ***INCREMENTAL LEARNING.***

Advantages & Disadvantages of Online Learning

Advantages:

- Easy for maintaining solutions that need continuous update
- Don’t need a lot of computing power and can learn from the New Data
- Best for Live Solutions
- Works easily on low resource devices like Smartphones or rover on the Mars which don’t have a lot of processing power

Disadvantages:

- If bad data is fed to the system, the system’s performance will gradually decline.
- If we are talking about a live system, clients will notice the downfall of the system that occurs due to the bad data.
  For example, bad data could come from a malfunctioning sensor on a robot, or from someone spamming a search engine to try to rank high in search results. To reduce this risk, you need to monitor your system closely and promptly switch learning off (and possibly revert to a previously working state) if you detect a drop in performance.
- You may also have to monitor the input data and react to abnormal data (e.g., using an anomaly detection algorithm).

Classification – based on Generalization

Based on how the Generalization happens, Machine Learning can be divided into:

Instance-based Learning

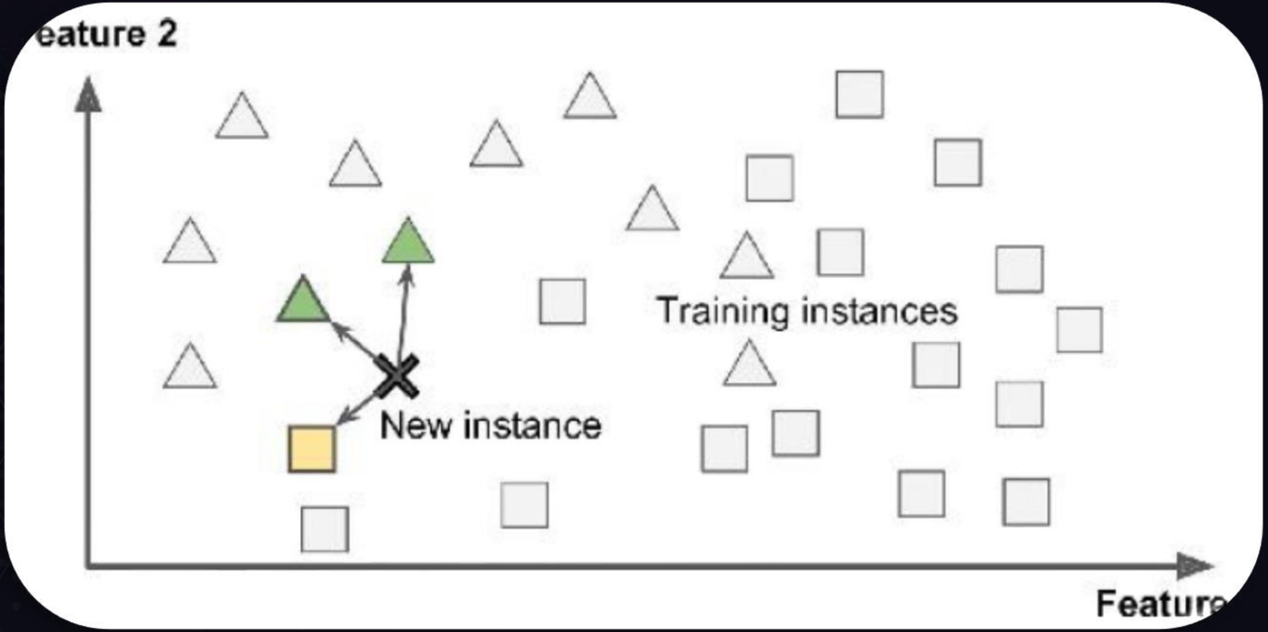
Model-based Learning

Instance-based Learning

In Instance-based learning, the system learns by heart. For example: Let’s say you have built a Spam Filter system which identifies emails that are identical to previously Tagged Spam Emails.

We can update this system by not just only Tagging emails that are identical to Spam emails but also has some similarity with the Spam Email. A simple approach of this can be measuring same words in both the emails. The system will identify email as Spam which already has many words common with a previously known Spam email.

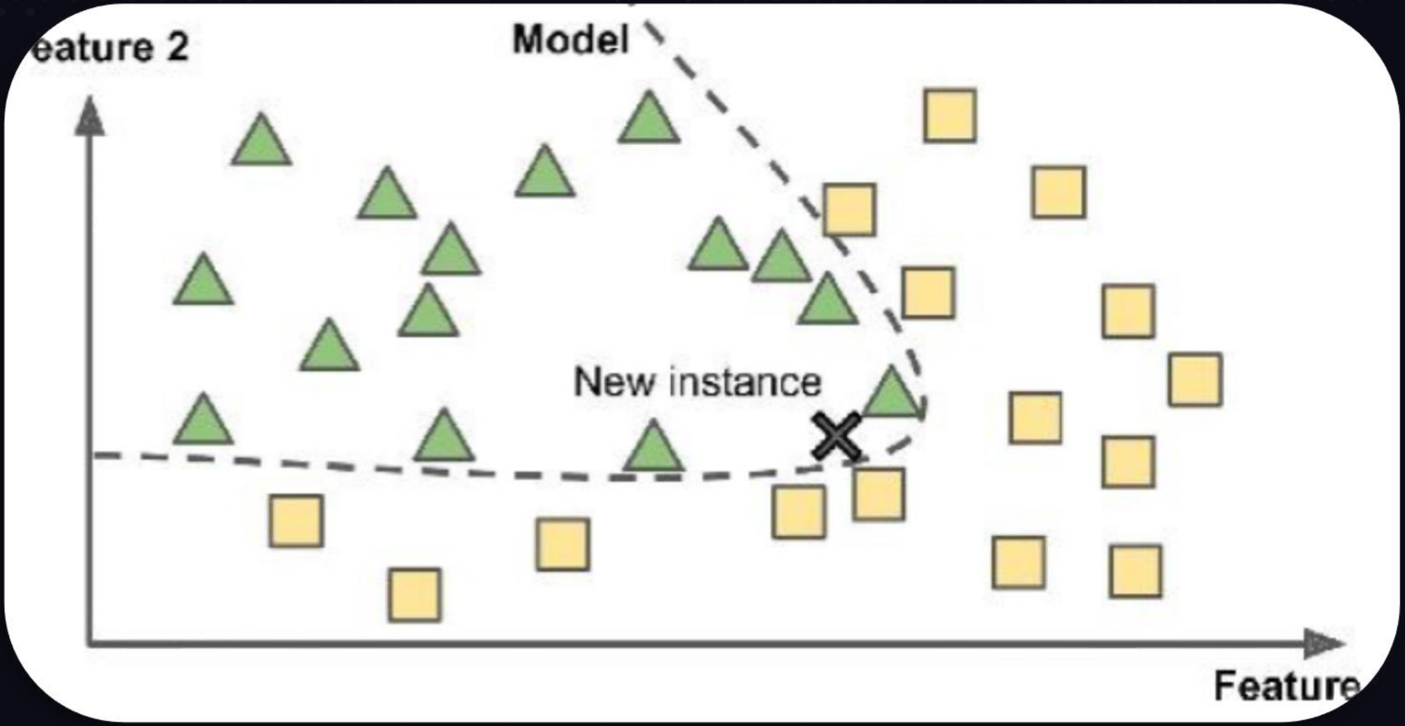
This is called Instance-based learning. The system learns the examples by heart, then generalizes to new cases using a similarity measure.



Model-based Learning

A common way to generalize a machine learning system is based on making a Model from a set of examples. Then save this model somewhere and use this model to predict the Accuracy of the system by giving new data into the system.

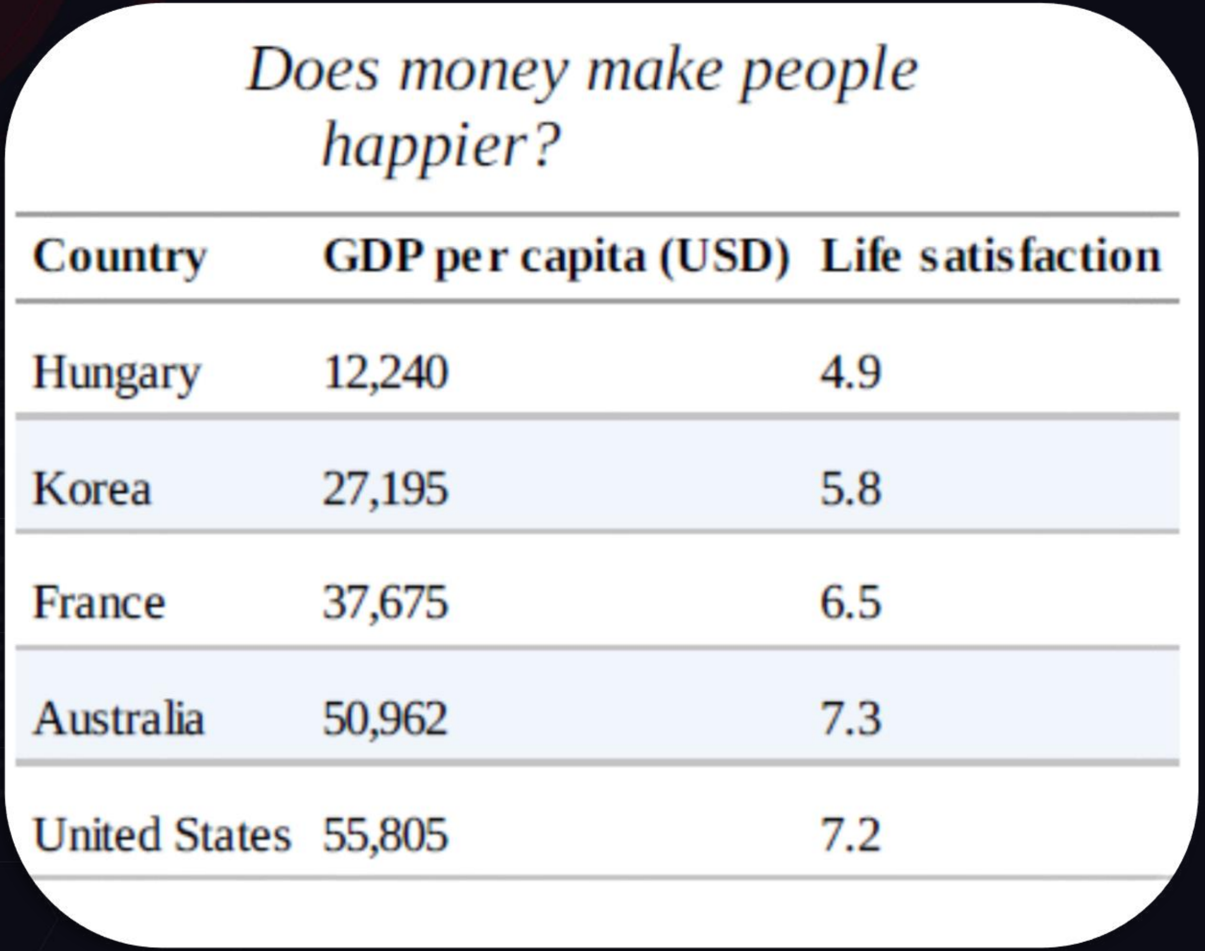
This is called Model-based learning.



Model-based Learning – Example

Suppose you want to know if people get happy with the money! If money makes people happy. So you download the Better Life Index data from the OECD's website as well as stats about GDP Per Capita from the IMF's website.

Then you join the tables and sort by GDP per capita.



Does money make people happier?

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2

What is Corpus

A Corpus is a collection of Well-organized Data that has been developed or generated for solving a particular problem.

For example -

- IMDB 5000 Movie Dataset Corpus for Sentiment Analysis
- MNIST Corpus for English Digit Recognition
- ImageNet for Image Processing

Challenges of Machine Learning

The Main Challenges that can happen in Machine Learning are the following:

- Bad Data
- Bad Algorithm

BAD DATA EXAMPLES

+88 0172 6867 984

tanvir@socian.ai

SOCIAN

Insufficient Quantity of Training Data

For a small baby to learn what an apple is, you just point the Apple in front of the Baby and say the word “Apple” several times (possibly repeating this procedure a few times). Now the child is able to recognize apples in all sorts of colors and shapes.

Genius!!!

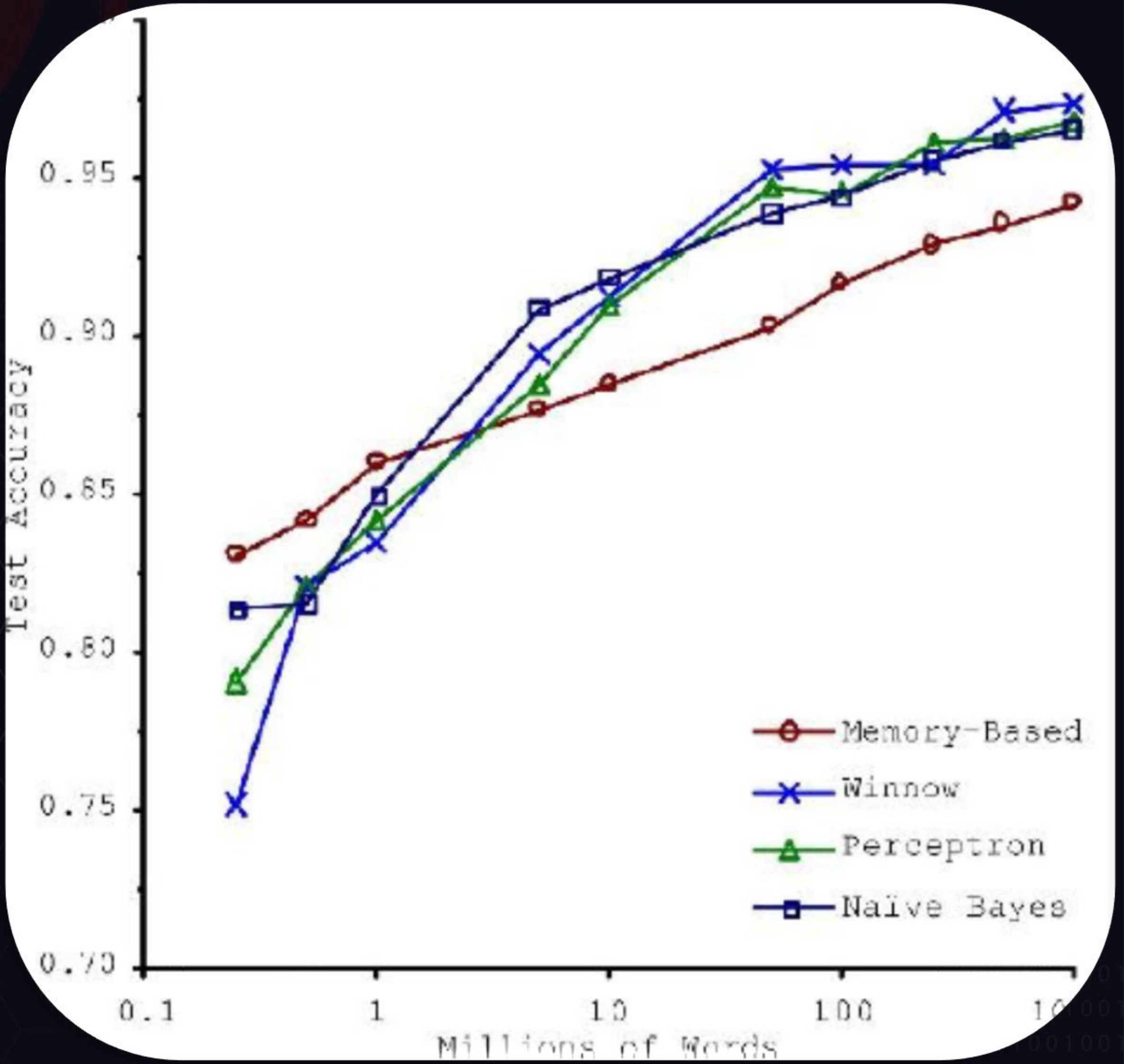
Machine Learning is not quite there yet; it takes a lot of data for most Machine Learning algorithms to work properly. Even for very simple problems you typically need thousands of examples, and for complex problems such as image or speech recognition you may need millions of examples ***(unless you can reuse parts of an existing model)***.

SOCIAN

Data is KING

In a famous paper published in 2001, Microsoft researchers Michele Banko and Eric Brill showed that very different Machine Learning algorithms, including fairly simple ones, performed almost identically well on a complex problem of natural language disambiguation once they were given enough data.

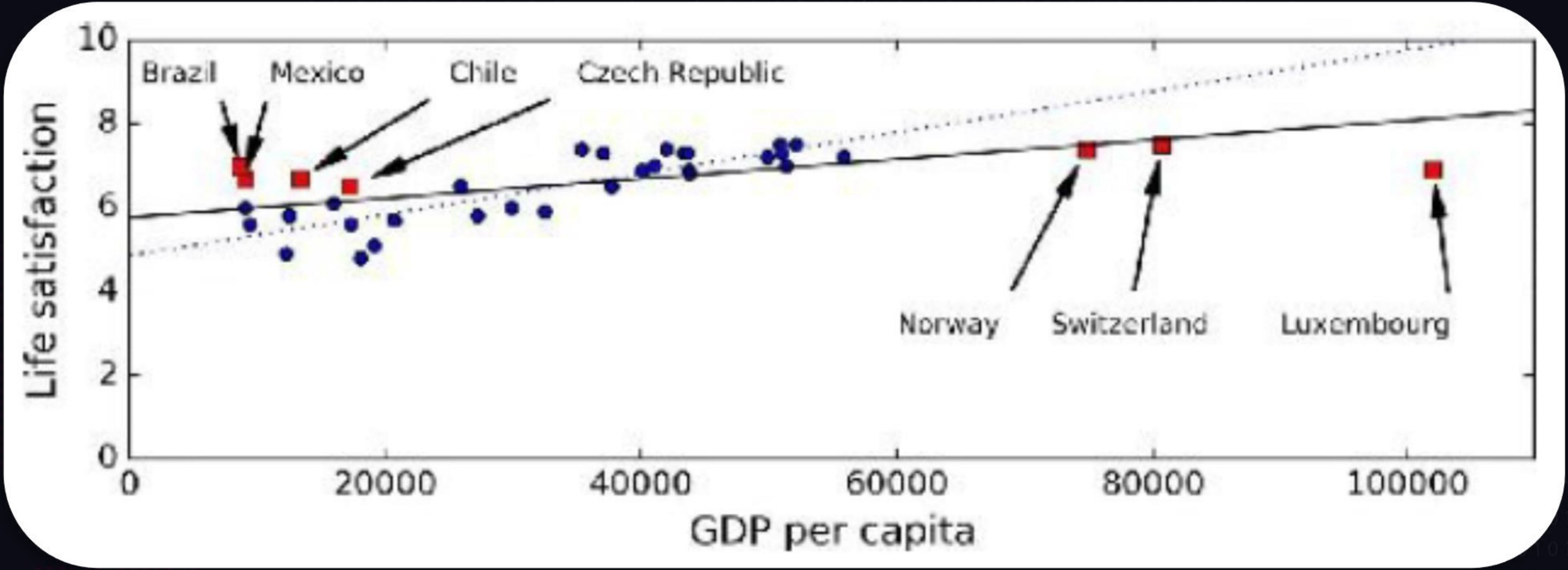
These results suggest that we may want to reconsider the trade-off between spending time and money on algorithm development versus spending it on corpus development.



Non-representative Training Data

It is very important that your Training Data can be properly represented by your Test Data that you want to generalize.

For example: If we look into the Missing Values for the use case where Money makes people Happy and we train a linear model on this data, we get the solid line, while the old model is represented by the dotted line:



Non-representative Training Data Cont.

On the previous figure, we can see that by adding a few missing countries significantly altered the model. From the figure, it makes it clear that such a simple linear model is probably never going to work well.

It seems that very rich countries are not happier than moderately rich countries (in fact they seem unhappier), and conversely some poor countries seem happier than many rich countries.

It is crucial to use a training set that is representative of the cases you want to generalize to. This is often harder than it sounds: if the sample is too small, you will have sampling noise (i.e., nonrepresentative data as a result of chance), but even very large samples can be nonrepresentative if the sampling method is flawed. This is called ***SAMPLING BIAS***.

Sampling Bias

Sample bias occurs when the distribution of one’s training data doesn’t reflect the actual environment that the machine learning model will be running in.

For example - If you’re trying to build a self-driving car, and you want it to drive at all times of day — at day and at night — and you’re only building training data based on daylight video, then that’s a bias that’s in your data.

The humans helping to build the training data for the algorithm can be completely correct, and have no bias. And yet the data is still biased because we didn’t include any nighttime examples. It’s bias to the day time.

It’s our job as data scientists to make sure the sample we’re building on matches the environment it’s going to be deployed in

Poor-Quality Data

If your training data is full of errors, outliers, and noise (e.g., due to poor-quality measurements), it will make it harder for the system to detect the underlying patterns, so your system is less likely to perform well. It is often well worth the effort to spend time cleaning up your training data.

The truth is, most data scientists spend a significant part of their time doing just that.

For example: You have obtained Voice Data collection with their transcription from a same user. Now, you want to built a Text to Speech System out of this data using AI. But unfortunately, 20% of these data contains noise.

The system will perform lot better if we can identify the audios that contain the noise and not use them in our Training.

Not Being Able to Identify Relevant Features

As we have seen over the time (The House Price Prediction example), we had lots of features in our available data. But all of these features may not be relevant for our System to learn. There is a saying: “Garbage in, Garbage out!”

Even the existing data may be missing some Relevant Features.

A critical part of the success of a Machine Learning project is coming up with a good set of features to train on. This process, called FEATURE ENGINEERING. It involves (We can get rid of this step in Deep Learning):

Feature Selection: Selecting the most useful features to train on among existing features.
Feature Extraction: Combining existing features to produce a more useful one (as we saw earlier, dimensionality reduction algorithms can help). Creating new features by gathering new data.

Topic - 01
Introduction to Machine Learning: Concept & Fundamentals
Part: 04

+88 0172 6867 984

tanvir@socian.ai

SOCIAN

BAD ALGORITHM EXAMPLES

+88 0172 6867 984

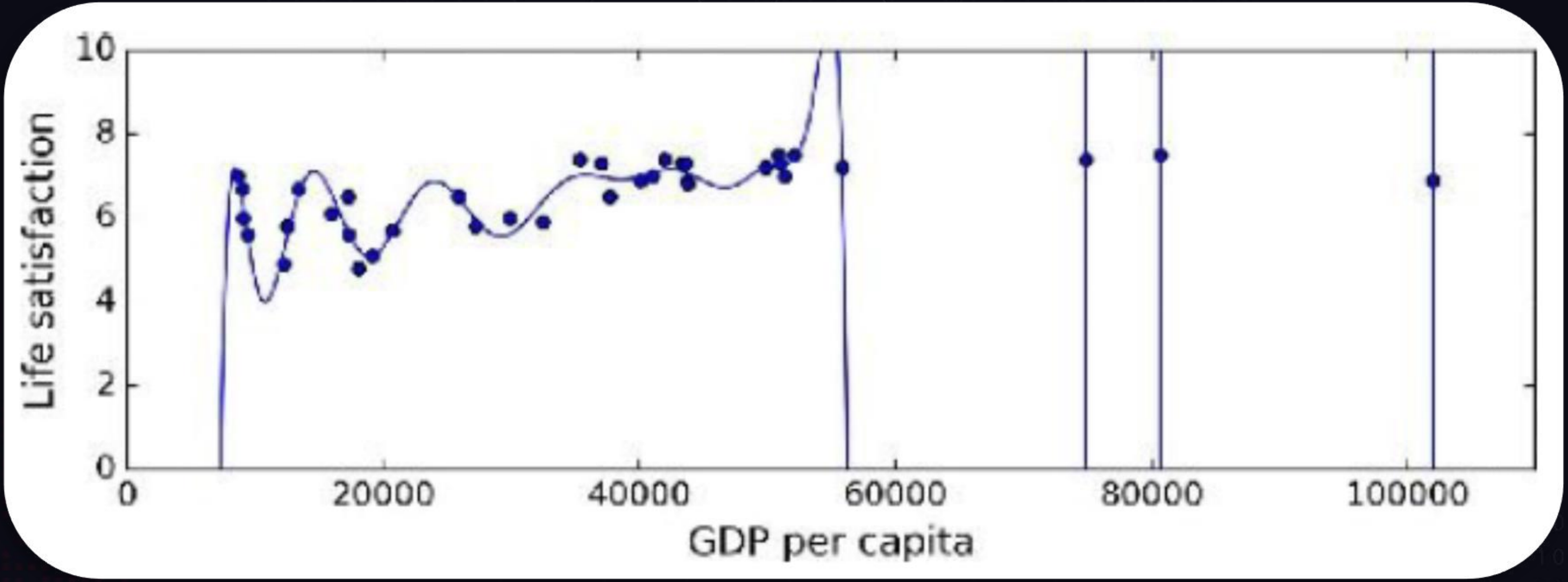
tanvir@socian.ai

SOCIAN

Overfitting the Training Data

Say a foreign tourist has come to Bangladesh and the taxi driver rips the tourist off. You might be tempted to say that all taxi drivers in Bangladesh are thieves. We humans do that quite often.

Unfortunately machines can fall into the same trap if we are not careful. In Machine Learning this is called overfitting: it means that the model performs well on the training data, but it does not generalize well.



Overfitting the Training Data Cont.

In fact, Overfitting is a common incident in Machine Learning.

For example, say you feed your Life Satisfaction with Money model has many more attributes, including Country’s Name. In that case, a complex model may detect patterns like the fact that all countries in the training data with a ***W*** in their name have a life satisfaction greater than 7: New Zealand (7.3), Norway (7.4), Sweden (7.2), and Switzerland (7.5).

How confident are you that the W-satisfaction rule generalizes to Rwanda or Zimbabwe? Obviously this pattern occurred in the training data by pure chance, but the model has no way to tell whether a pattern is real or simply the result of noise in the data.

How to Get Rid of Overfitting

Overfitting happens when the model is too complex relative to the amount and noisiness of the training data.

The possible solutions are:

- To simplify the model by selecting one with fewer parameters (e.g., a linear model rather than a high-degree polynomial model)
- By reducing the number of attributes in the training data by constraining the model
- To gather more training data
- To reduce the noise in the training data (e.g., fix data errors and remove outliers)

Overfitting & Regularization

Let’s talk about another Real Life scenario where the Overfitting happens.

Let’s say you are doing Speech to Text (STT) with the help of Deep Learning.

If you keep training your data with certain features and run the process iteratively, it will also start finding patterns in the Training Data that is not needed.

For example: Let’s say there were 20 Users’ Voice Data in the Training Dataset. If we overfit the model, the Accuracy will increase but the problem that will happen is that the System will only be able to successfully do Speech to Text on those 20 Users. Here, the system is overfit with the Voice of the 20 users. It won’t work well for new Speakers.

To get rid of this, we look into Accuracy in both Training Set and Test Set and where the values are similar, we stop the Training at that stage so that the System doesn’t Overfit anymore.

Constraining a model to make it simpler and reduce the risk of overfitting is called ***REGULARIZATION***

Regularization Hyperparameter

The amount of regularization to apply during learning can be controlled by a Hyperparameter.

A hyperparameter is a parameter of a learning algorithm (not of the model). As such, it is not affected by the learning algorithm itself; it must be set prior to training and remains constant during training.

If you set the regularization hyperparameter to a very large value, you will get an almost flat model (a slope close to zero); the learning algorithm will almost certainly not overfit the training data, but it will be less likely to find a good solution.

Tuning hyperparameters is an important part of building a Machine Learning system

Underfitting the Training Data

Underfitting is the opposite of overfitting. It occurs when your model is too simple to learn the underlying structure of the data.

For example, a linear model of life satisfaction is prone to underfit; reality is just more complex than the model, so its predictions are bound to be inaccurate, even on the training examples.

The main options to fix this problem are:

- Selecting a more powerful model, with more parameters
- Feeding better features to the learning algorithm (Feature Engineering)
- Reducing the constraints on the model (e.g., reducing the Regularization Hyperparameter)

Testing & Validating

The only way to know how well a model will generalize to new cases is to actually try it out on new cases. One way to do that is to put our model in production and monitor how well it performs. This works well, but if our model is horribly bad, your users will complain — not the best idea.

A better option is to split our data into two sets: the Training set and the Test set. As these names imply, we train your model using the training set, and we test it using the test set. The error rate on new cases is called the ***GENERALIZATION ERROR*** (or ***OUT-OF-SAMPLE*** error),

By evaluating our model on the test set, we get an estimation of this error. This value tells us how well your model will perform on instances it has never seen before. If the ***TRAINING ERROR*** is low (i.e., your model makes few mistakes on the training set) but the generalization error is high, it means that your model is overfitting the training data.

Testing & Validating Cont.

It is common to use 80% of the data for training and hold out 20% for testing. But sometimes we also use 70%-30% or separate the data based on our needs.

Now suppose you are hesitating between two models (say a linear model and a polynomial model): which one to use? One option is to train both Models and compare how well each of them generalize using the same test set.

Now suppose that the linear model generalizes better, but you want to apply some regularization to avoid overfitting. The question is: how do you choose the value of the regularization hyperparameter?

One option is to train 100 different models using 100 different values for this hyperparameter. Let's say you find the best hyperparameter value that produces a model with the lowest generalization error - 5%. So you launch this model into production, but unfortunately it does not perform as well as expected and produces 15% errors. What went wrong?

The problem is that you measured the generalization error multiple times on the test set, and you adapted the model and hyperparameters to produce the best model for that set. This means that the model is unlikely to perform as well on new data.

Testing & Validating Cont..

A common solution to this problem is to have a second holdout set called the ***VALIDATION SET***. You train multiple models with various hyperparameters using the training set, you select the model and hyperparameters that perform best on the validation set, and when you're happy with your model you run a single final test against the test set to get an estimate of the generalization error.

To avoid “wasting” too much training data in validation sets, a common technique is to use cross-validation: the training set is split into complementary subsets, and each model is trained against a different combination of these subsets and validated against the remaining parts.

Once the model type and hyperparameters have been selected, a final model is trained using these hyperparameters on the full training set, and the generalized error is measured on the test set.

SOCIAN

Questions to Look Into

1. How would you define Machine Learning?
2. Can you name four types of problems where it shines?
3. What is a labeled training set?
4. What are the two most common supervised tasks?
5. Can you name four common unsupervised tasks?
6. What type of Machine Learning algorithm would you use to allow a robot to walk in various
unknown terrains?
7. What type of algorithm would you use to segment your customers into multiple groups?
8. Would you frame the problem of spam detection as a supervised learning problem or an
unsupervised learning problem?
9. What is an online learning system?
10. What is out-of-core learning?

Questions to Look Into Cont.

11. What type of learning algorithm relies on a similarity measure to make predictions?
12. What is the difference between a model parameter and a learning algorithm’s hyperparameter?
13. What do model-based learning algorithms search for? What is the most common strategy they use to
succeed? How do they make predictions?
14. Can you name four of the main challenges in Machine Learning?
15. If your model performs great on the training data but generalizes poorly to new instances, what is
happening? Can you name three possible solutions?
16. What is a test set and why would you want to use it?
17. What is the purpose of a validation set?
18. What can go wrong if you tune hyperparameters using the test set?
19. What is cross-validation and why would you prefer it to a validation set?