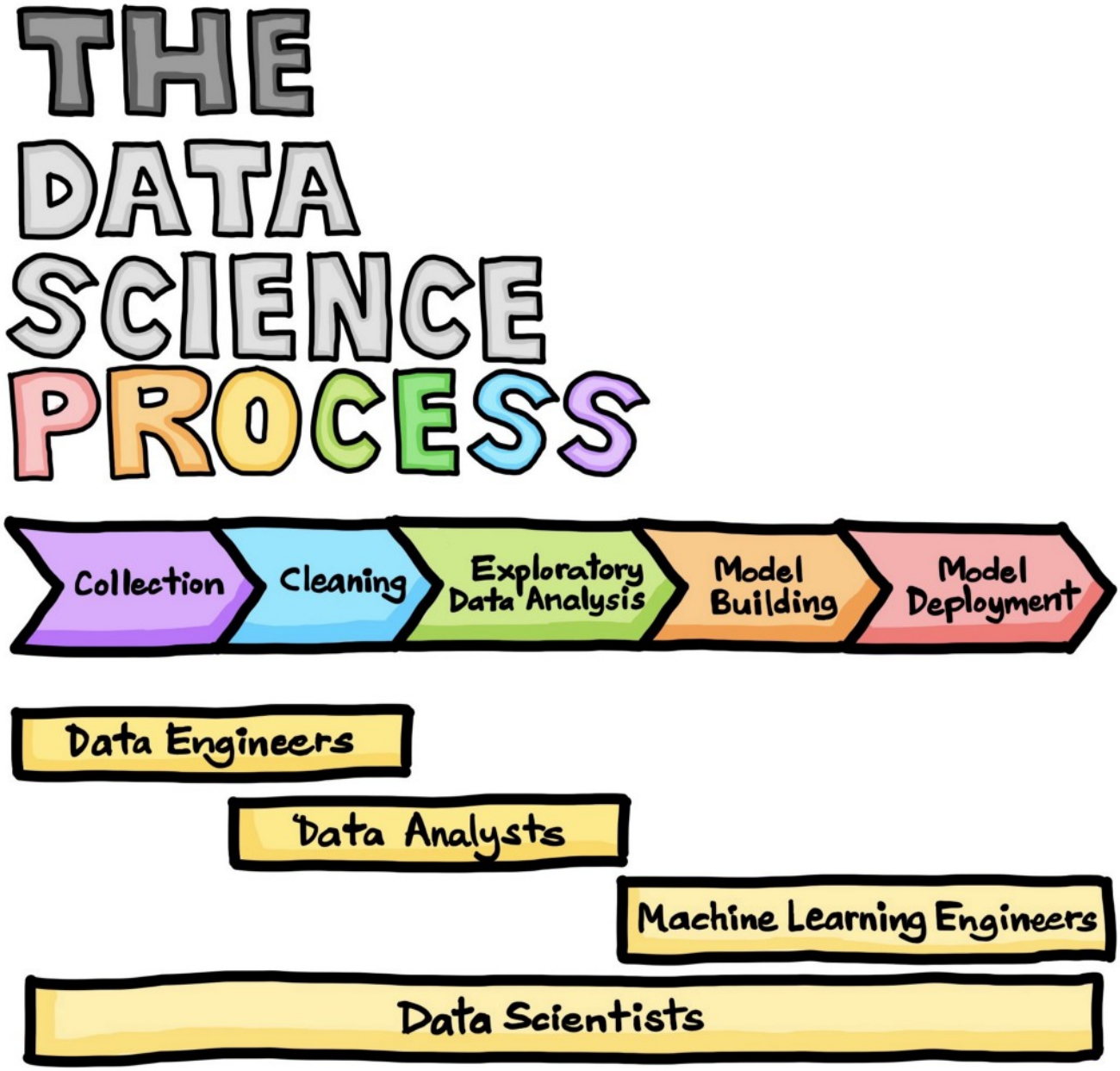


Data science life cycle. (Drawn by Chanin Nantasenamat in collaboration with Ken Jee)

DATA SCIENCE

The Data Science Process

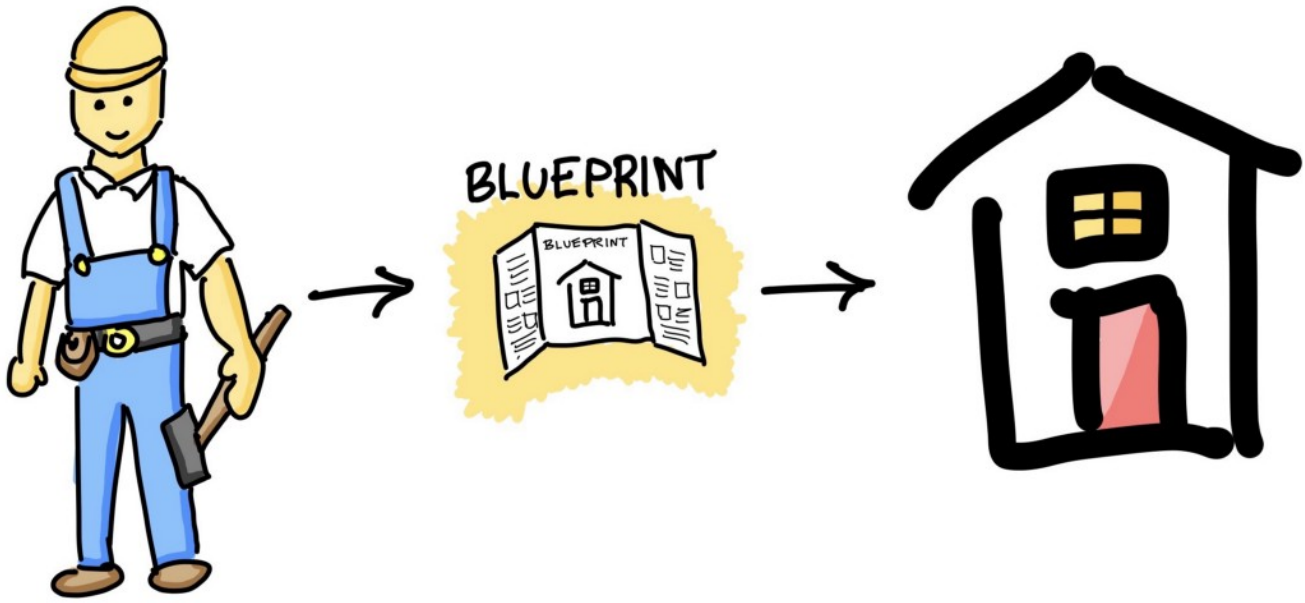
A Visual Guide to Standard Procedures in Data Science

Chanin Nantasenamat

Jul 28 · 7 min read ★

Let's suppose that you've been given a data problem to solve and you're expected to produce unique insights from the data given to you. So the question is, what do you exactly do to transform a data problem through to completion and generate data-driven insights? And most importantly of all, *Where do you start?*

Let's use some analogy here, in the construction of a house or building the guiding piece of information used is the blueprint. So what sorts of information are contained within these blueprints? Information pertaining to the building infrastructure, the layout and exact dimensions of each room, the location of water pipes and electrical wires, etc.

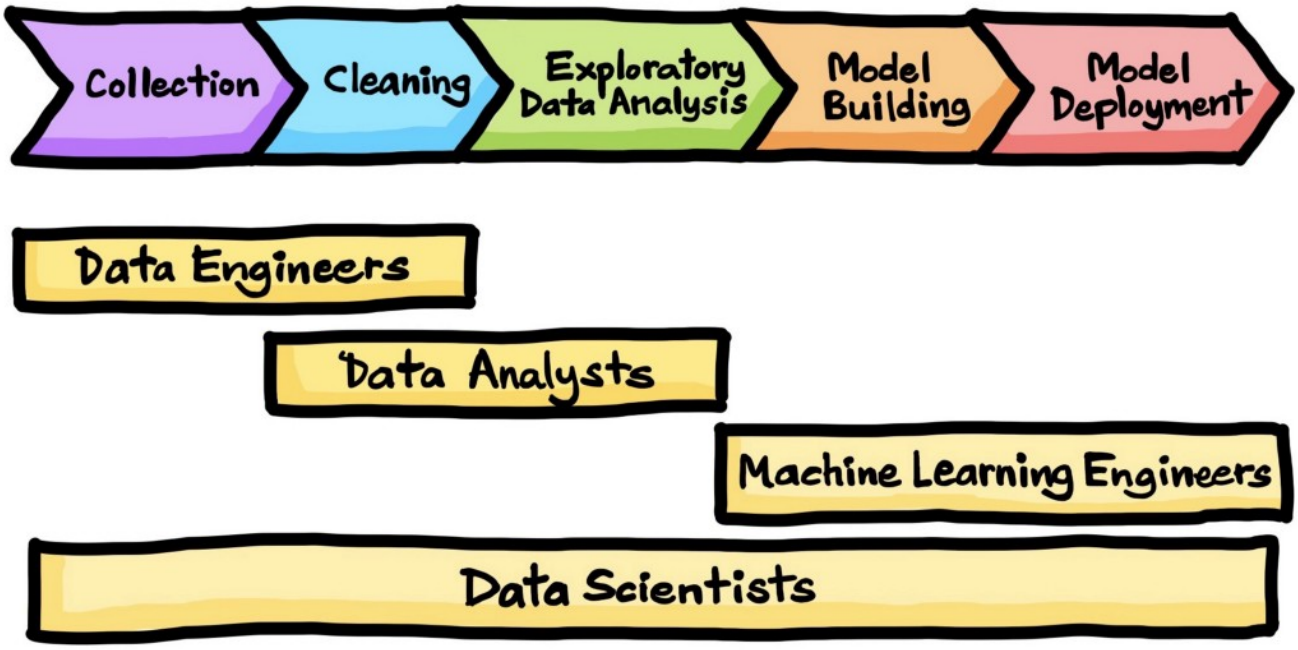


Continuing from where we left off earlier, so where do we start when given a data problem? That is where the *Data Science Process* comes in. As will be discussed in the forthcoming sections of this article, the data science process provides a systematic approach for tackling a data problem. By following through on these recommended guidelines, you will be able to make use of a tried-and-true workflow in approaching data science projects. So without further ado, let's get started!

Data Science Life Cycle

The *data science life cycle* is essentially comprised of data collection, data cleaning, exploratory data analysis, model building and model deployment. For more information, please check out the excellent video by Ken Jee on the *Different Data*

Science Roles Explained (by a Data Scientist). A summary infographic of this life cycle is shown below:



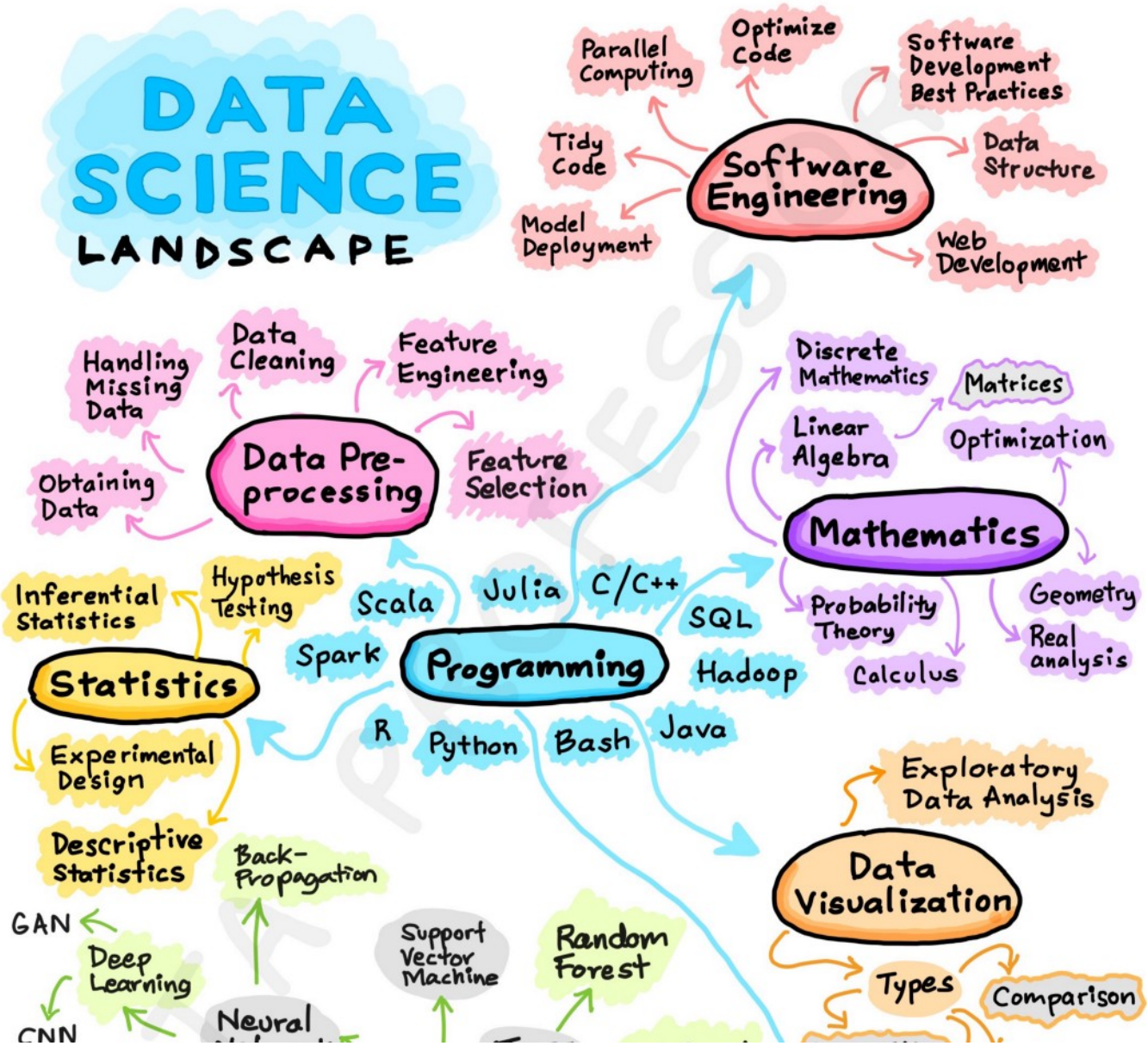
Data science life cycle. (Drawn by Chanin Nantasenamat in collaboration with Ken Jee)

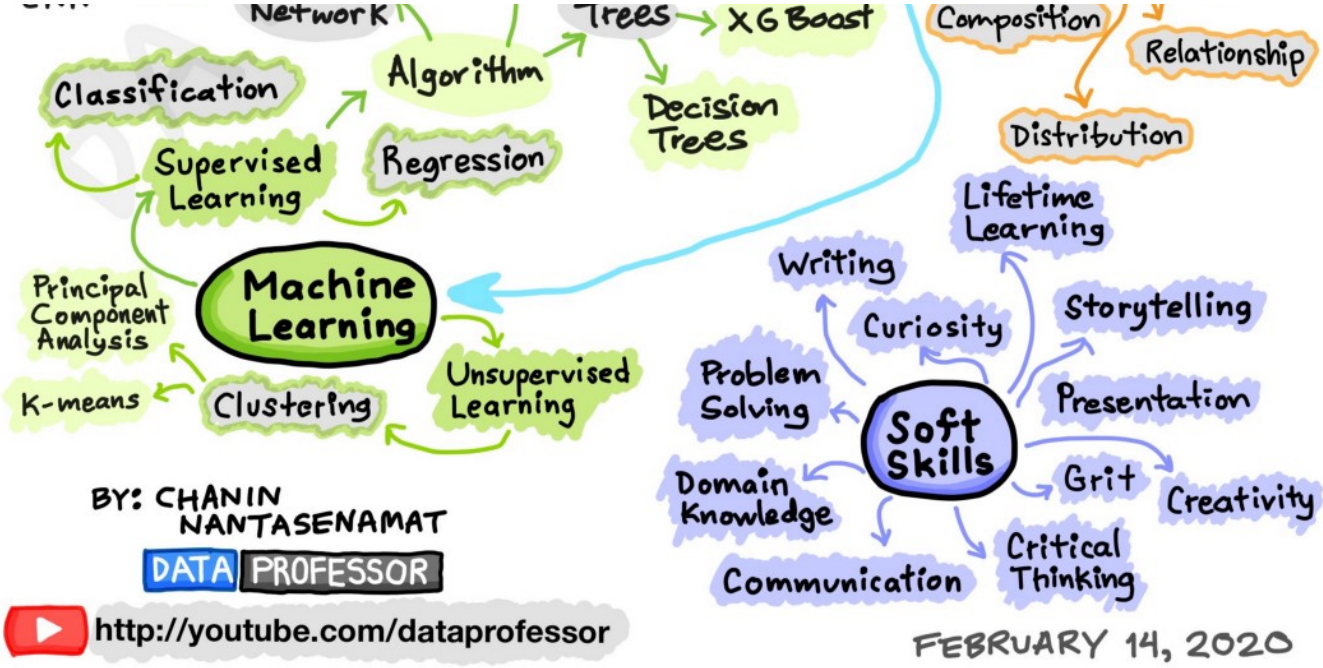
Such a process or workflow of drawing insights from data is best described by CRISP-DM and OSEMN. It should be noted that both are comprised of essentially the same core concepts while each framework was released at different time. Particularly, CRISP-DM was released at a time (1996) when data mining has started to gain traction

and was missing a standard protocol for carrying out data mining tasks in a robust manner. Fourteen years later (2010), the OSEMN framework was introduced and it summarizes the key tasks of a data scientist.

Personally, having started my own journey into the world of data in 2004 and the field was known back then as *Data Mining*. Much of the emphasis at the time was placed in translating data to knowledge where another common term that is also used to refer to data mining is *Knowledge Discovery in Data*.

Over the years, the field has matured and evolved to encompass other skillsets that led to the eventual coining of the term *Data Science* that goes beyond merely building models but also encompasses other skillsets both technical and soft skills. Previously, I have drawn an infographic that summarizes these 8 essential skillsets of data science as shown below. Also check out the accompanying YouTube video on *How to Become a Data Scientist (Learning Path and Skill Sets Needed)*.





8 Skillsets of Data Science. (Drawn by Chanin Nantasenamat)

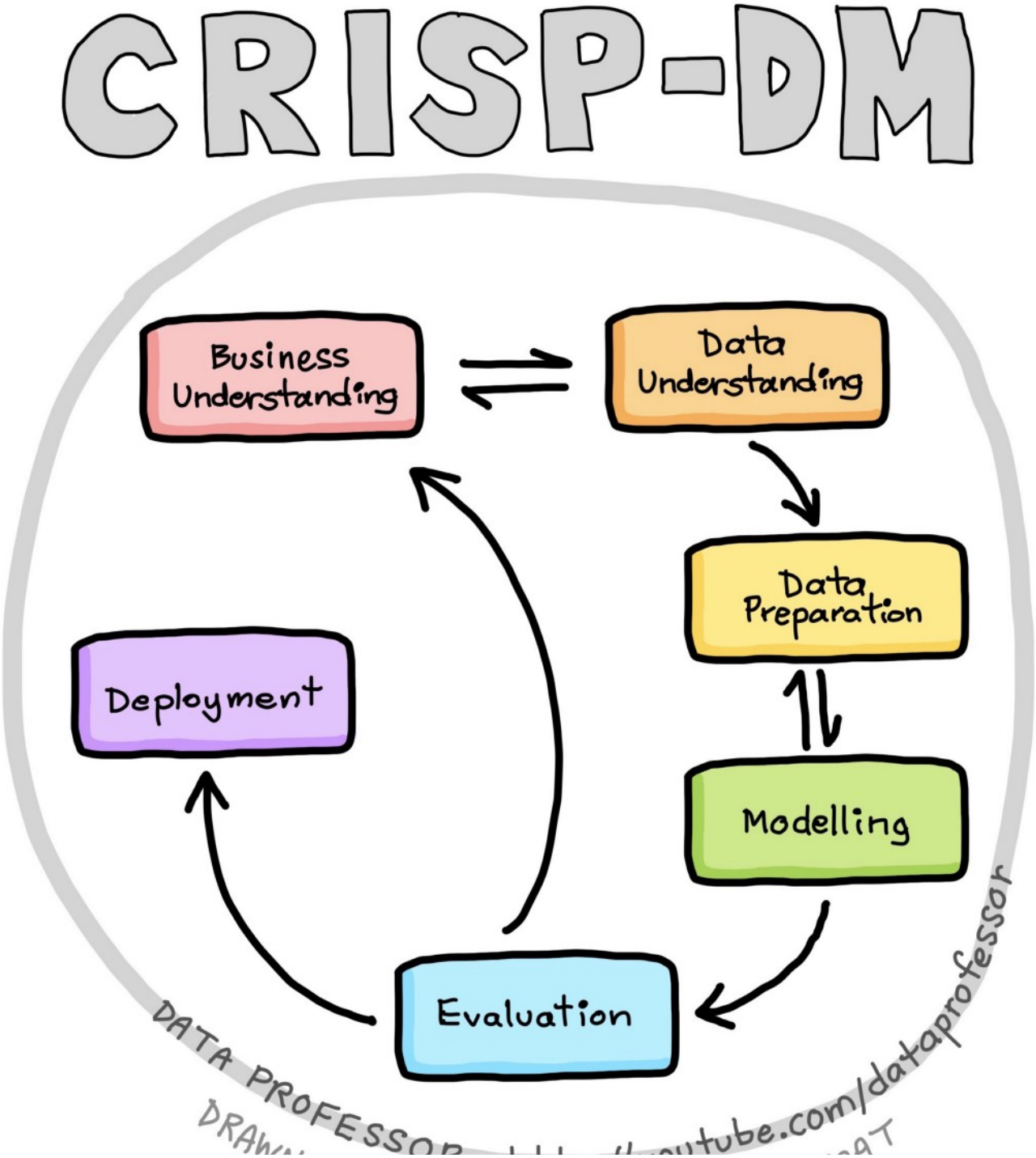
CRISP-DM

The acronym CRISP-DM stands for Cross Industry Standard Process for Data Mining and CRISP-DM was introduced in 1996 in efforts to standardize the process of data mining (also referred to as knowledge discovery in data) such that it can serve as a

standard and reliable workflow that can be adopted and applied in various industry. Such standard process would serve as a *"best practice"* that boasts several benefits.

Aside from providing a reliable and consistent of process by which to follow in carrying out data mining projects but it would also instill confidence to customers and stakeholders who are looking to adopt data mining in their organizations.

It should be noted that back in 1996, data mining had just started to gain mainstream attention and was at the early phases and the formulation of a standard process would help to lay the solid foundation and groundwork for early adopters. A more in-depth historical look of CRISP-DM is provided in the article by Wirth and Hipp (2000).



Standard process for performing data mining according to the CRISP-DM framework. (Drawn by Chanin Nantasenamat)

The **CRISP-DM framework** is comprised of 6 major steps:

1. ***Business understanding*** — This entails the understanding of a project's objectives and requirements from the business viewpoint. Such business perspectives are used to figure out what business problems to solve via the use of data mining.

2. ***Data understanding*** — This phase allows us to become familiarize with the data and this involves performing exploratory data analysis. Such initial data exploration may allow us to figure out which subsets of data to use for further modeling as well as aid in the generation of hypothesis to explore.

3. ***Data preparation*** — This can be considered to be the most time-consuming phase of the data mining process as it involves rigorous data cleaning and pre-processing as well as the handling of missing data.

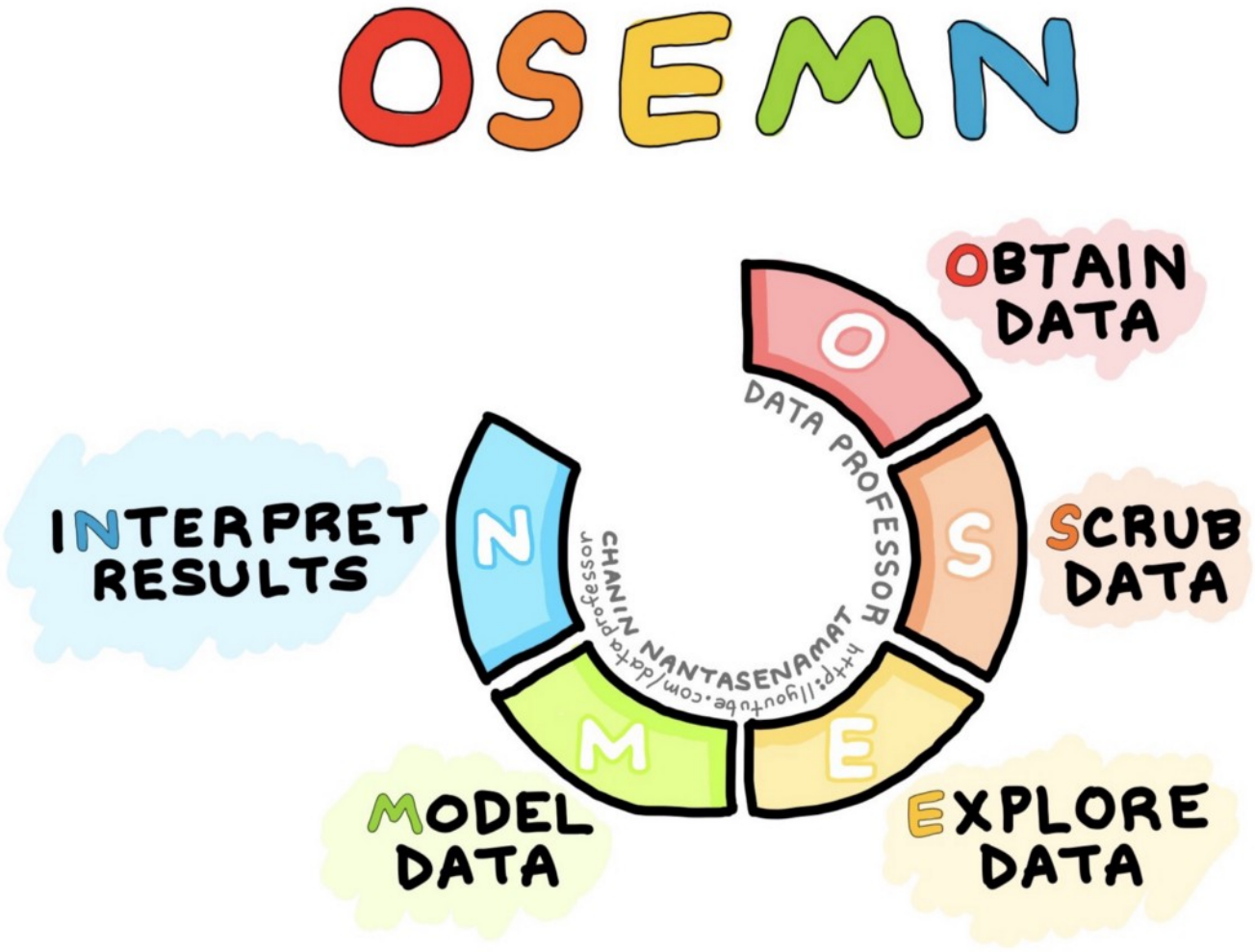
4. ***Modelling*** — The pre-processed data are used for model building in which learning algorithms are used to perform multivariate analysis.

5. ***Evaluation*** — In performing the 4 aforementioned steps, it is important to evaluate the accrued results and review the process performed thusfar to determine whether the originally set business objectives are met or not. If deemed appropriate, some steps may need to be performed again. Rinse and repeat. Once it is deemed that the results and process are satisfactory then we are ready to move to deployment. Additionally, in this evaluation phase, some findings may ignite new project ideas for which to explore.

6. ***Deployment*** — Once the model is of satisfactory quality, the model is then deployed, which may range from being a simple report, an API that can be accessed via programmatic calls, a web application, etc.

OSEMN

In a 2010 post ***"A Taxonomy of Data Science"*** on dataists blog, Hilary Mason and Chris Wiggins introduced the OSEMN framework that essentially constitutes a taxonomy of the general workflow that data scientists typically perform as shown in the diagram below. Shortly after in 2012, Davenport and Patil published their landmark article *"Data Scientist: The Sexiest Job of the 21st Century"* in the Harvard Business Review that has attracted even more attention to the burgeoning field of data science.



Data science process described in 5 steps by Mason and Higgins (2000) known as the OSEMN framework. (Drawn by Chanin Nantasenamat)

The **OSEMN framework** is comprised of 5 major steps and can be summarized as follows:

1. ***Obtain Data*** — Data forms the requisite of the data science process and data can come from pre-existing ones or from newly acquired data (from surveys), from newly queried data (from databases or APIs), downloaded from the internet (e.g. from repositories available on the cloud such as GitHub) or extracted

2. ***Scrub Data*** — Scrubbing the data is essentially data cleaning and this phase is considered to be the most time-consuming as it involves handling missing data as well as pre-processing it to be as error-free and uniform as possible.

3. ***Explore Data*** — This is essentially exploratory data analysis and this phase allows us to gain an understanding of the data such that we can figure out the course of actions and areas that we can to explore in the modeling phase. This entails the use of descriptive statistics and data visualizations.

4. ***Model Data*** — Here, we make use of machine learning algorithms in efforts to make sense of data and gain useful insights that are essential for data-driven decision-making.

5. ***Interpret Results*** — This is perhaps one of the most important phase and yet the least technical as it pertains to actually making sense of the data by figuring out how to simplify and summarize results from all the models built. This entails drawing meaningful conclusion and rationalizing actionable insights that would essentially allow us to figure out what the next course of actions are. For example, what are the most important features that influences the class labels (**Y** variables).

Conclusion

In summary, we have gone covered the data science process by showing you the highly simplified data science life cycle along with the widely popular CRISP-DM and OSEMN frameworks. These frameworks provides a high-level guidance on handling a data science project from end to end where all encompasses the same core concepts of data compilation, pre-processing, exploration, modeling, evaluation, interpretation and deployment. It should be noted that the flow amongst these processes is not linear and that in practice the flow can be non-linear and can re-iterate until satisfactory condition is met.

About Me

I work full-time as an Associate Professor of Bioinformatics and Head of Data Mining and Biomedical Informatics at a Research University in Thailand. In my after work

hours, I'm a YouTuber (AKA the Data Professor) making online videos about data science. In all tutorial videos that I make, I also share Jupyter notebooks on GitHub (Data Professor GitHub page).

Data Professor

Data Science, Machine Learning, Bioinformatics, Research and Teaching are my passion. The Data Professor YouTube...

www.youtube.com

Connect with Me on Social Network

- ☑ YouTube: http://youtube.com/dataprofessor/
- ☑ Website: http://dataprofessor.org/ (Under construction)
- ☑ LinkedIn: https://www.linkedin.com/company/dataprofessor/
- ☑ Twitter: https://twitter.com/thedataprof
- ☑ FaceBook: http://facebook.com/dataprofessor/
- ☑ GitHub: https://github.com/dataprofessor/
- ☑ Instagram: https://www.instagram.com/data.professor/

Task Cheatsheet for Almost Every Machine Learning Project

A checklist of tasks for building End-to-End ML projects

Harshit Tyagi
Jul 4 · 5 min read ★

Task Cheatsheet for Machine Learning Projects

To-do Checklist + Add a view Search New

#	Steps	Desc
1	Define the problem from a high-level view	Problem Statement
2	Get the data	Problem Statement
3	Initial Exploration of Data	Data Analysis
4	Exploratory Data Analysis to uncover hidden patterns and data transformation for ETL pipelines	Data Analysis Data Engineering
5	Analyse results of different models and shortlist the ones with good performance measure	Model Engineering Data Analysis
6	Fine-tune the shortlisted models check for ensemble methods	Model Engineering
7	Document code and communicate your solution	Data Analysis
8	Deploy and Monitor	Data Engineering

As I am working on creating a range of portfolio-worthy projects for all of you, I thought of documenting practices that I've either learned from someone or developed while working.

In this blog, I've captured the checklist of tasks that I keep referring to while working on an end-to-end ML project.

> *Why do I even need a checklist?*

Since you are required to deal with numerous elements in a project (wrangling, preparation, questions, models, fine-tuning, etc.), it's easy to lose track of things.

It guides you through the next steps and pushes you to check if every task has been executed successfully or not.

Sometimes, we struggle to find the starting point, the checklist helps you elicit the right information(data) from the right sources in order to establish relationships and uncover correlational insights.

It's a best practice to have every part of the project undergo a paradigm of checks.

As **Atul Gawande** says in his book — The Checklist Manifesto,

> the volume and complexity of what we know has exceeded our individual ability to deliver its benefits correctly, safely, or reliably.

So, let me walk you through this crisp and concise list of action items that will reduce your workload and enhance your output…

Machine Learning Project Checklist

These are 8–10 steps that you have to perform in almost every ML project. A few of the steps can be executed interchangeably in order.

1. Define the problem from a high-level view

This is to understand and articulate the business logic of the problem. It should tell you:

- the nature of the problem(supervised/unsupervised, classification/regression),
- type of solutions you can develop
- what metrics you should use to measure performance?

- is machine learning the right approach to solve this problem?
- manual approach to solving the problem.
- the inherent assumptions of the problem

2. Identify the data sources and acquire the data

In most cases, this step can be executed before the first step if you have the data with you and you want to define the questions(problem) around it to make better use of the incoming data.

Based on the definition of your problem, you'd need to identify the sources of data which can be a database, a data repository, sensors, etc. For an application to be deployed in production, this step should be automated by developing data pipelines to keep the incoming data flowing into the system.

- list the sources and amount of data you need.
- check if space is going to be an issue.
- check if you're authorized to use the data for your purpose or not.
- acquire the data and convert it into a workable format.
- check the type of data(textual, categorical, numerical, time series, images)
- take aside a sample of it for final testing purposes.

3. Initial Exploratoration of Data

This is the step where you study all the features that impact your outcome/prediction/target. If you have a huge chunk of data, sample it down for this step to make the analysis more manageable.
Steps to follow:

- use jupyter notebooks as they provide an easy and intuitive interface to study the data.
- identify the target variable
- identify the types of features(categorical, numerical, textual, etc.)
- analyze the correlation between features.

- add a few data visualizations for easy interpretation of the impact of each feature on the target variable.
- document your findings.

4. Exploratory Data Analysis to Prepare the Data

It's time to execute the findings of the previous step by defining functions for data transformations, cleaning, feature selection/engineering, and scaling.

- Write functions to transform the data and automate the process for the forthcoming batches of data.
- Write functions to clean the data(imputing missing values and handling outliers)
- Write functions to select and engineer features — drop redundant features, format conversion of features, and other mathematical transformations.
- Feature scaling — standardize the features.

5. Develop a baseline model and then explore other models to shortlist the best ones

Create a very basic model which should serve as the baseline for all the other complex machine learning model. Checklist of steps:

- Train a few commonly used ML models like naive bayes, linear regression, SVM, etc using default parameters.
- Measure and compare the performance of each model with the baseline and with all the others.
- Employ *N*-fold cross-validation for each model and compute the mean and standard deviation of the performance metrics on the *N* folds.
- Study the features that have the most impact on the target.
- Analyze the types of errors the models make while predicting.
- Engineer the features in a different manner.
- Repeat the above steps a few times(trial and error) to be sure that we have used the right features in the right format.
- Shortlist the top models based on their performance measures.

6. Fine-tune your shortlisted models and check for ensemble methods

This needs to be one of the crucial steps where you would be moving closer to your final solution. Major steps should include:

- Hyperparameter tuning using cross-validation.
- Use automated tuning methods like random search or grid search to find out the best configuration for your best models.
- Test ensemble methods like voting classifiers etc.
- Test the models with as much data as possible.
- Once finalized, use the unseen test sample that we set aside, in the beginning, to check for overfitting or underfitting.

7. Document Code and Communicate your solution

The process of communication is manifold. You need to keep in mind all the existing and potential stakeholders. Therefore the major steps include:

- Document the code as well as your approach and journey throughout your project.
- Create a dashboard like voila or an insightful presentation with close to self-explanatory visualizations.
- Write a blog/report capturing how you analyzed the features, tested different transformations, etc. Capture your learning(failures and techniques that worked)
- Conclude with the main outcome and future scope(if any)

8. Deploy your model in production, Monitor!

If your project requires deployment to be tested on live data, you should create a web application or a REST API to be used across all platforms(web, android, iOS). Major steps(would vary depending on the project) include:

- Save your final trained model into an h5 or pickle file.
- Serve your model using web services, you can use Flask to develop these web services.
- Connect the input data sources and set up the ETL pipelines.

- Manage dependencies using pipenv, docker/Kubernetes(based on scaling requirements)
- You can use AWS, Azure, or Google Cloud Platform to deploy your service.
- Monitor the performance on live data or simply for people to use your model with their data.

Note: The checklist can be adapted depending on the complexity of your project.

Data Science with Harshit

With this channel, I am planning to roll out a couple of series covering the entire data science space. Here is why you should be subscribing to the channel:

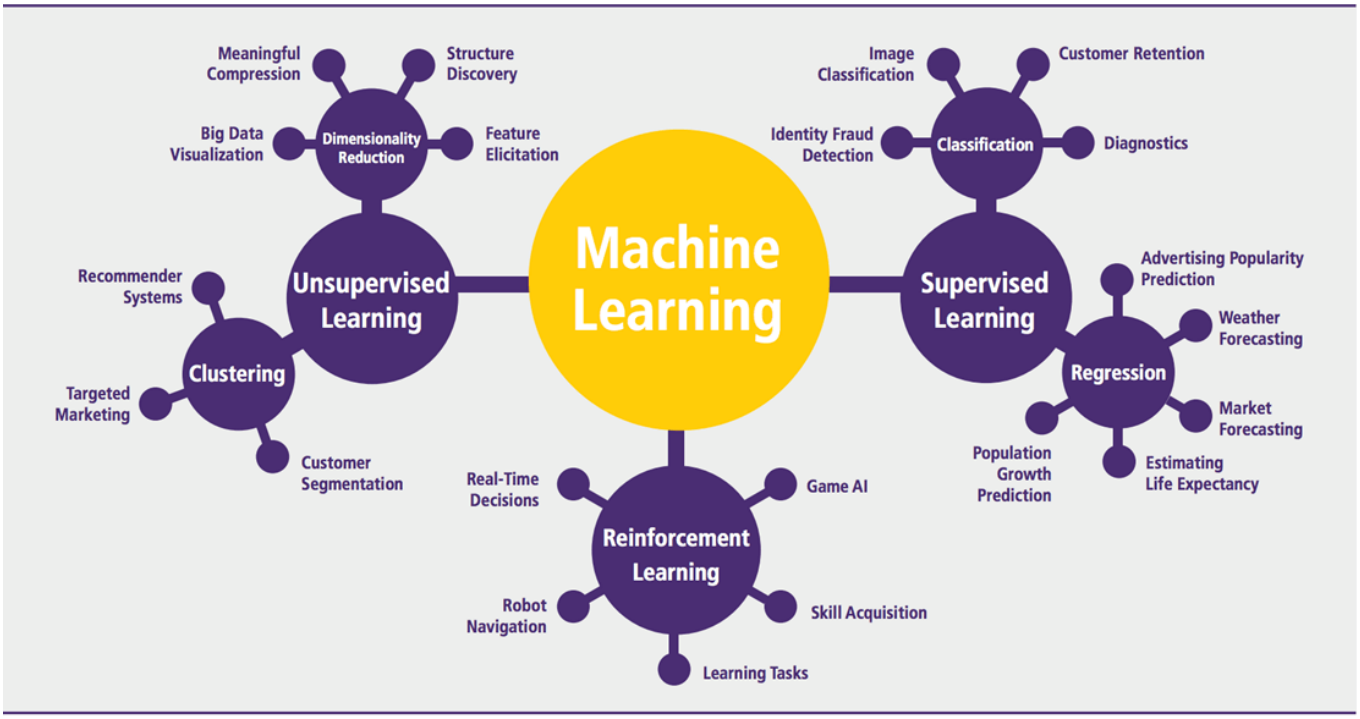
- These series would cover all the required/demanded quality tutorials on each of the topics and subtopics like Python fundamentals for Data Science.
- Explained Mathematics and derivations of why we do what we do in ML and Deep Learning.

- Podcasts with Data Scientists and Engineers at Google, Microsoft, Amazon, etc, and CEOs of big data-driven companies.
- Projects and instructions to implement the topics learned so far. Learn about new certifications, Bootcamp, and resources to crack those certifications like this **TensorFlow Developer Certificate Exam by Google.**

Analytics Vidhya

Which Machine Learning Algorithm Should You Use By Problem Type?

Sukanya Bag Follow
Oct 7 · 5 min read



When I was beginning my way in data science, I **often faced the problem** of **choosing the most appropriate algorithm** for my specific problem. If you're like me, when you open some article about machine learning algorithms, you see dozens of detailed descriptions. The **paradox** is that they **don't ease the choice.**

Well, to not let you feel out of the track, I would suggest you to have a good understanding of the implementation and mathematical intuition behind several supervised and unsupervised Machine Learning Algorithms like -

- **Linear regression**

- **Logistic regression**
- **Decision tree**
- **Naive Bayes**
- **Support vector machine**
- **Random forest**
- **AdaBoost**
- **Gradient-boosting trees**
- **Simple neural network**
- **Hierarchical clustering**
- **Gaussian mixture model**
- **Convolutional neural network**
- **Recurrent neural network**
- **Recommender system**

Remember, the list of Machine Learning Algorithms I mentioned are the ones that are mandatory to have a good knowledge of , while you are a beginner in Machine/Deep Learning !

Now that we have some intuition about types of machine learning tasks, let's explore the most popular algorithms with their applications in real life, based on their problem statements !

Try to work on each of these problem statements after getting to the end of this blog ! I can assure you would learn a lot, a hell lot!

Problem Statement 1 -

To Predict the Housing Prices

Machine Learning Algorithm(s) to solve the problem —

- **Advanced regression techniques like random forest and gradient boosting**

Problem Statement 2 -

Explore customer demographic data to identify patterns

Machine Learning Algorithm(s) to solve the problem —

- **Clustering (elbow method)**

Problem Statement 3 -

Predicitng Loan Repayment

Machine Learning Algorithm(s) to solve the problem —

- **Classification Algorithms for imbalanced dataset**

Problem Statement 4 -

Predict if a skin lesion is benign or malignant based on its characteristics (size, shape, color, etc)

Machine Learning Algorithm(s) to solve the problem —

- **Convolutional Neural Network (U-Net being the best for segmentation stuffs)**

Problem Statement 5 -

Predict client churn

Machine Learning Algorithm(s) to solve the problem —

- **Linear discriminant analysis** (LDA) or **Quadratic discriminant analysis** (QDA)

(particularly popular because it is both a classifier and a dimensionality reduction technique)

Problem Statement 6 -

Provide a decision framework for hiring new employees

Machine Learning Algorithm(s) to solve the problem —

- **Decision Tree** is a pro gamer here

Problem Statement 7 -

Understand and predict product attributes that make a product most likely to be purchased

Machine Learning Algorithm(s) to solve the problem **—**

- **Logistic Regression**
- **Decision Tree**

Problem Statement 8 -

Analyze sentiment to assess product perception in the market.

Machine Learning Algorithm(s) to solve the problem **—**

- **Naive Bayes — Support Vector Machines** (NBSVM)

Problem Statement 9 -

Create classification system to filter out spam emails

Machine Learning Algorithm(s) to solve the problem **—**

- **Classification Algorithms —**

Naive Bayes, SVM , Multilayer Perceptron Neural Networks (MLPNNs) and **Radial Base Function Neural Networks (RBFNN) suggested.**

Problem Statement 10 -

Predict how likely someone is to click on an online ad

Machine Learning Algorithm(s) to solve the problem **—**

- **Logistic Regression**
- **Support Vector Machines**

Problem Statement 11 -

Detect fraudulent activity in credit-card transactions.

Machine Learning Algorithm(s) to solve the problem —

- **Adaboost**
- **Isolation Forest**
- **Random Forest**

Problem Statement 12 -

Predict the price of cars based on their characteristics

Machine Learning Algorithm(s) to solve the problem —

- **Gradient-boosting trees are best at this.**

Problem Statement 13 -

Predict the probability that a patient joins a healthcare program

Machine Learning Algorithm(s) to solve the problem —

- **Simple neural networks**

Problem Statement 14 -

Predict whether registered users will be willing or not to pay a particular price for a product.

Machine Learning Algorithm(s) to solve the problem —

- **Neural Networks**

Problem Statement 15 -

Segment customers into groups by distinct charateristics (eg, age group)

Machine Learning Algorithm(s) to solve the problem —

- **K-means clustering**

Problem Statement 16 -

Feature extraction from speech data for use in speech recognition systems

Machine Learning Algorithm(s) to solve the problem —

- **Gaussian mixture model**

Problem Statement 17 -

Object tracking of multiple objects, where the number of mixture components and their means predict object locations at each frame in a video sequence.

Machine Learning Algorithm(s) to solve the problem —

- **Gaussian mixture model**

Problem Statement 18 -

Organizing the genes and samples from a set of microarray experiments so as to reveal biologically interesting patterns.

Machine Learning Algorithm(s) to solve the problem —

- **Hierarchical clustering algorithms**

Problem Statement 19 -

Recommend what movies consumers should view based on preferences of other customers with similar attributes.

Machine Learning Algorithm(s) to solve the problem —

- **Recommender system**

Problem Statement 20 -

Recommend news articles a reader might want to read based on the article she or he is reading.

Machine Learning Algorithm(s) to solve the problem —

- **Recommender system**

Problem Statement 21 -

Recommend news articles a reader might want to read based on the article she or he is reading.

Machine Learning Algorithm(s) to solve the problem —

- **Recommender system**

Problem Statement 22 -

Optimize the driving behavior of self-driving cars

Machine Learning Algorithm(s) to solve the problem —

- **Reinforcement Learning**

Problem Statement 23 -

Diagnose health diseases from medical scans.

Machine Learning Algorithm(s) to solve the problem —

- **Convolutional Neural Networks**

Problem Statement 24 -

Balance the load of electricity grids in varying demand cycles

Machine Learning Algorithm(s) to solve the problem —

- **Reinforcement Learning**

Problem Statement 25 -

When you are working with time-series data or sequences (eg, audio recordings or text)

Machine Learning Algorithm(s) to solve the problem —

- **Recurrent neural network**
- LSTM

Problem Statement 26 -

Provide language translation

Machine Learning Algorithm(s) to solve the problem —

- **Recurrent neural network**

Problem Statement 27 -

Generate captions for images

Machine Learning Algorithm(s) to solve the problem —

- **Recurrent neural network**

Problem Statement 28 -

Power chatbots that can address more nuanced customer needs and inquiries

Machine Learning Algorithm(s) to solve the problem —

- **Recurrent neural network**

I hope that I could explain to you common perceptions of the most used machine learning algorithms and give intuition on how to choose one for your specific problem.

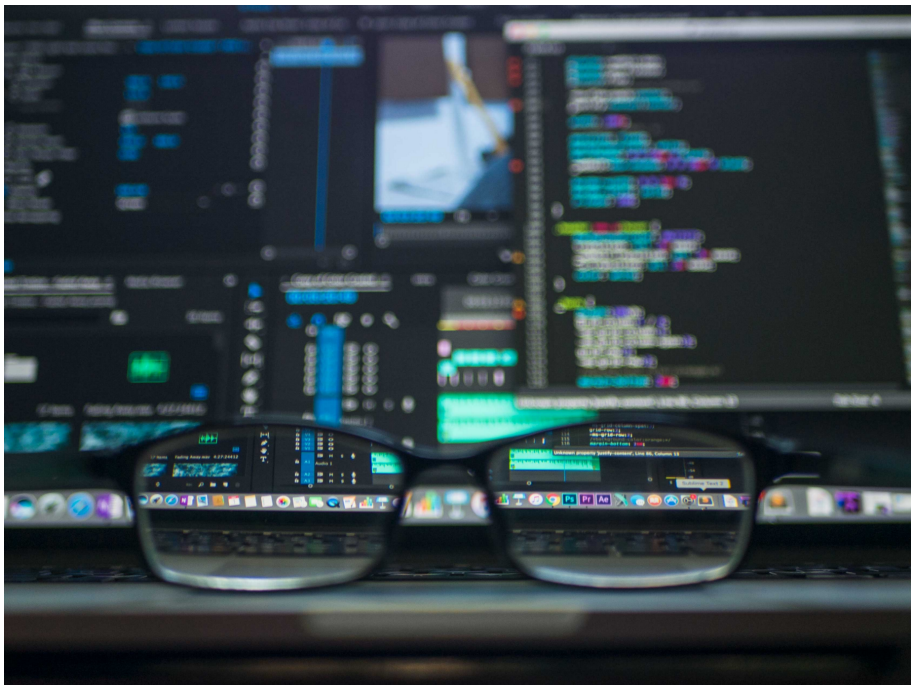
Happy Machine Learning ! :)

Until next time..!

All Machine Learning Models Explained in 6 Minutes

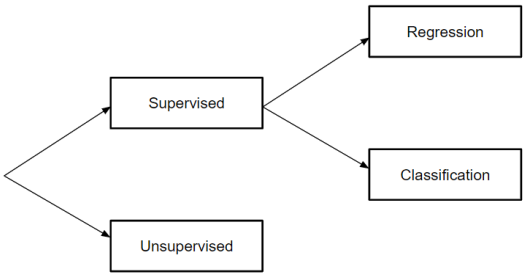
Intuitive explanations of the most popular machine learning models.

Terence S Follow
Jan 6 · 7 min read ★



If you like this, you should check out my free data science resource with new material every week!

In my previous article, I explained what **regression** was and showed how it could be used in application. This week, I'm going to go over the majority of common machine learning models used in practice, so that I can spend more time building and improving models rather than explaining the theory behind it. Let's dive into it.



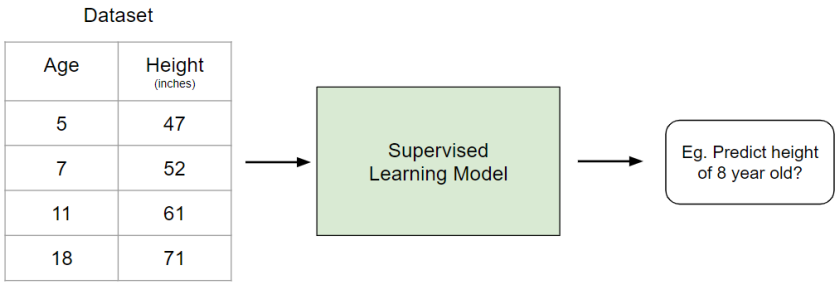
Fundamental Segmentation of Machine Learning Models

All machine learning models are categorized as either **supervised** or **unsupervised**. If the model is a supervised model, it's then sub-categorized as either a **regression** or **classification** model. We'll go over what these terms mean and the corresponding models that fall into each category below.

Supervised Learning

Supervised learning involves learning a function that maps an input to an output based on example input-output pairs [1].

For example, if I had a dataset with two variables, age (input) and height (output), I could implement a supervised learning model to predict the height of a person based on their age.



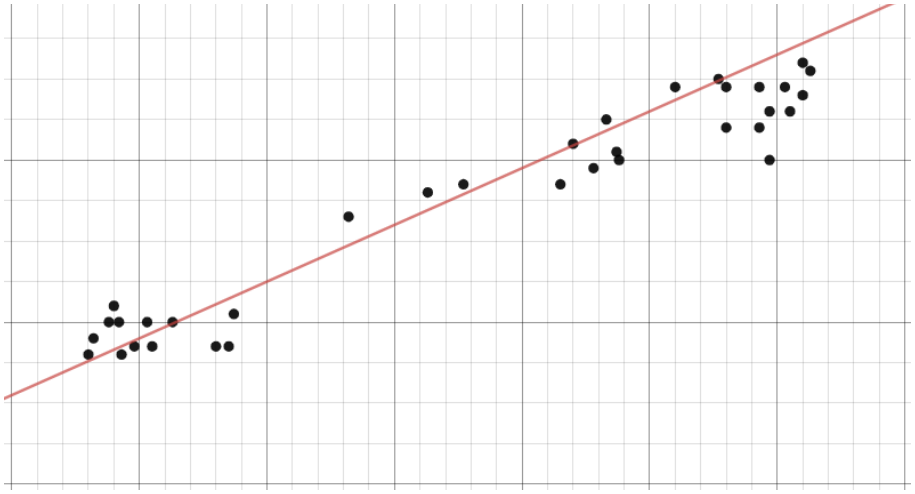
Example of Supervised Learning

To re-iterate, within supervised learning, there are two sub-categories: regression and classification.

Regression

In **regression** models, the output is continuous. Below are some of the most common types of regression models.

Linear Regression



Example of Linear Regression

The idea of linear regression is simply finding a line that best fits the data. Extensions of linear regression include multiple linear regression (eg. finding a plane of best fit) and polynomial regression (eg. finding a curve of best fit). You can learn more about linear regression in my previous article.

Decision Tree

Image taken from Kaggle

Decision trees are a popular model, used in operations research, strategic planning, and machine learning. Each square above is called a **node**, and the more nodes you have, the more accurate your decision tree will be (generally). The last nodes of the decision tree, where a decision is made, are called the **leaves** of the tree. Decision trees are intuitive and easy to build but fall short when it comes to accuracy.

Random Forest

Random forests are an ensemble learning technique that builds off of decision trees. Random forests involve creating multiple decision trees

using bootstrapped datasets of the original data and randomly selecting a subset of variables at each step of the decision tree. The model then selects the mode of all of the predictions of each decision tree. What's the point of this? By relying on a "majority wins" model, it reduces the risk of error from an individual tree.

For example, if we created one decision tree, the third one, it would predict 0. But if we relied on the mode of all 4 decision trees, the predicted value would be 1. This is the power of random forests.

StatQuest does an amazing job walking through this in greater detail. See here.

Neural Network

Visual Representation of a Neural Network

A Neural Network is essentially a network of mathematical equations. It takes one or more input variables, and by going through a network of equations, results in one or more output variables. You can also say that a neural network takes in a vector of inputs and returns a vector of outputs, but I won't get into matrices in this article.

The blue circles represent the **input layer,** the black circles represent the **hidden layers,** and the green circles represent the **output layer.** Each node in the hidden layers represents both a linear function and an activation function that the nodes in the previous layer go through, ultimately leading to an output in the green circles.

- If you would like to learn more about it, check out my beginner-friendly explanation on neural networks.

Classification

In classification models, the output is discrete. Below are some of the most common types of classification models.

Logistic Regression

Logistic regression is similar to linear regression but is used to model the probability of a finite number of outcomes, typically two. There are a number of reasons why logistic regression is used over linear regression when modeling probabilities of outcomes (see here). In essence, a logistic equation is created in such a way that the output values can only be between 0 and 1 (see below).

Support Vector Machine

A **Support Vector Machine** is a supervised classification technique that can actually get pretty complicated but is pretty intuitive at the most fundamental level.

Let's assume that there are two classes of data. A support vector machine will find a **hyperplane** or a boundary between the two classes of data that maximizes the margin between the two classes (see below). There are many planes that can separate the two classes, but only one plane can maximize the margin or distance between the classes.

If you want to get into greater detail, Savan wrote a great article on Support Vector Machines here.

Naive Bayes

Naive Bayes is another popular classifier used in Data Science. The idea behind it is driven by Bayes Theorem:

In plain English, this equation is used to answer the following question. "What is the probability of y (my output variable) given X? And because of the naive assumption that variables are independent given the class, you can say that:

As well, by removing the denominator, we can then say that P(y|X) is proportional to the right-hand side.

Therefore, the goal is to find the class y with the maximum proportional probability.

Check out my article "A Mathematical Explanation of Naive Bayes" if you want a more in-depth explanation!

Decision Tree, Random Forest, Neural Network

These models follow the same logic as previously explained. The only difference is that that output is discrete rather than continuous.

Unsupervised Learning

Unlike supervised learning, **unsupervised learning** is used to draw inferences and find patterns from input data without references to labeled outcomes. Two main methods used in unsupervised learning include clustering and dimensionality reduction.

Clustering

Taken from GeeksforGeeks

Clustering is an unsupervised technique that involves the grouping, or **clustering**, of data points. It's frequently used for customer segmentation, fraud detection, and document classification.

Common clustering techniques include **k-means** clustering, **hierarchical** clustering, **mean shift** clustering, and **density-based** clustering. While each technique has a different method in finding clusters, they all aim to achieve the same thing.

Dimensionality Reduction

Dimensionality reduction is the process of reducing the number of random variables under consideration by obtaining a set of principal variables [2]. In simpler terms, its the process of reducing the dimension of your feature set (in even simpler terms, reducing the number of features). Most dimensionality reduction techniques can be categorized as either **feature elimination** or **feature extraction.**

A popular method of dimensionality reduction is called **principal component analysis.**

Principal Component Analysis (PCA)

In the simplest sense, **PCA** involves project higher dimensional data (eg. 3 dimensions) to a smaller space (eg. 2 dimensions). This results in a lower dimension of data, (2 dimensions instead of 3 dimensions) while keeping all original variables in the model.

There is quite a bit of math involved with this. If you want to learn more about it…

Check out this awesome article on PCA here.

If you'd rather watch a video, StatQuest explains PCA in 5 minutes here.

Conclusion

Obviously, there is a ton of complexity if you dive into any particular model, but this should give you a fundamental understanding of how each machine learning algorithm works!

> *For more articles like this one, check out https://blog.datatron.com/*

References

[1] Stuart J. Russell, Peter Norvig, Artificial Intelligence: A Modern Approach (2010), Prentice Hall

[2] Roweis, S. T., Saul, L. K., Nonlinear Dimensionality Reduction by Locally Linear Embedding (2000), *Science*

Thanks for Reading!

If you like my work and want to support me…

1. The BEST way to support me is by following me on **Medium** here.
2. Follow me on **LinkedIn** here.
3. Sign up on my **email list** here.
4. Want to collaborate? Check out my website, **shintwin.com**