Introduction to Machine Learning

Some images and notes were retrieved from Aurélien Géron - Hands-on Machine Learning with Scikit-Learn and TensorFlow (2019, O'Reilly)

Definition

- Machine Learning (ML) is the science of writing programs in computers so they can **learn from data**.

- Another definition

  "*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.*" (Tom Mitchell, 1997)

  - task T → Spam Classification
  - experience E → previous emails labelled as spam/ham
  - performance measure P → accuracy

Definition

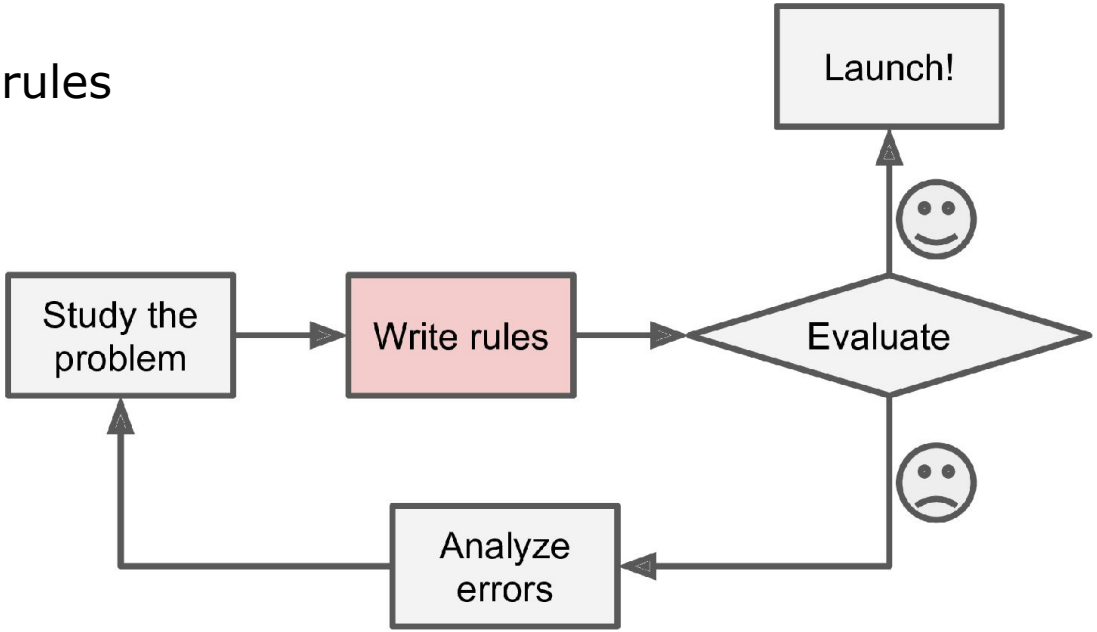
- Machine Learning is **NOT**
  - Collecting millions of tweets from Twitter
  - Tracing web traffic of users
  - Downloading data from Wikipedia

- Maybe these can be considered as a pre-step before running a Machine Learning algorithm.

Why do we need ML?

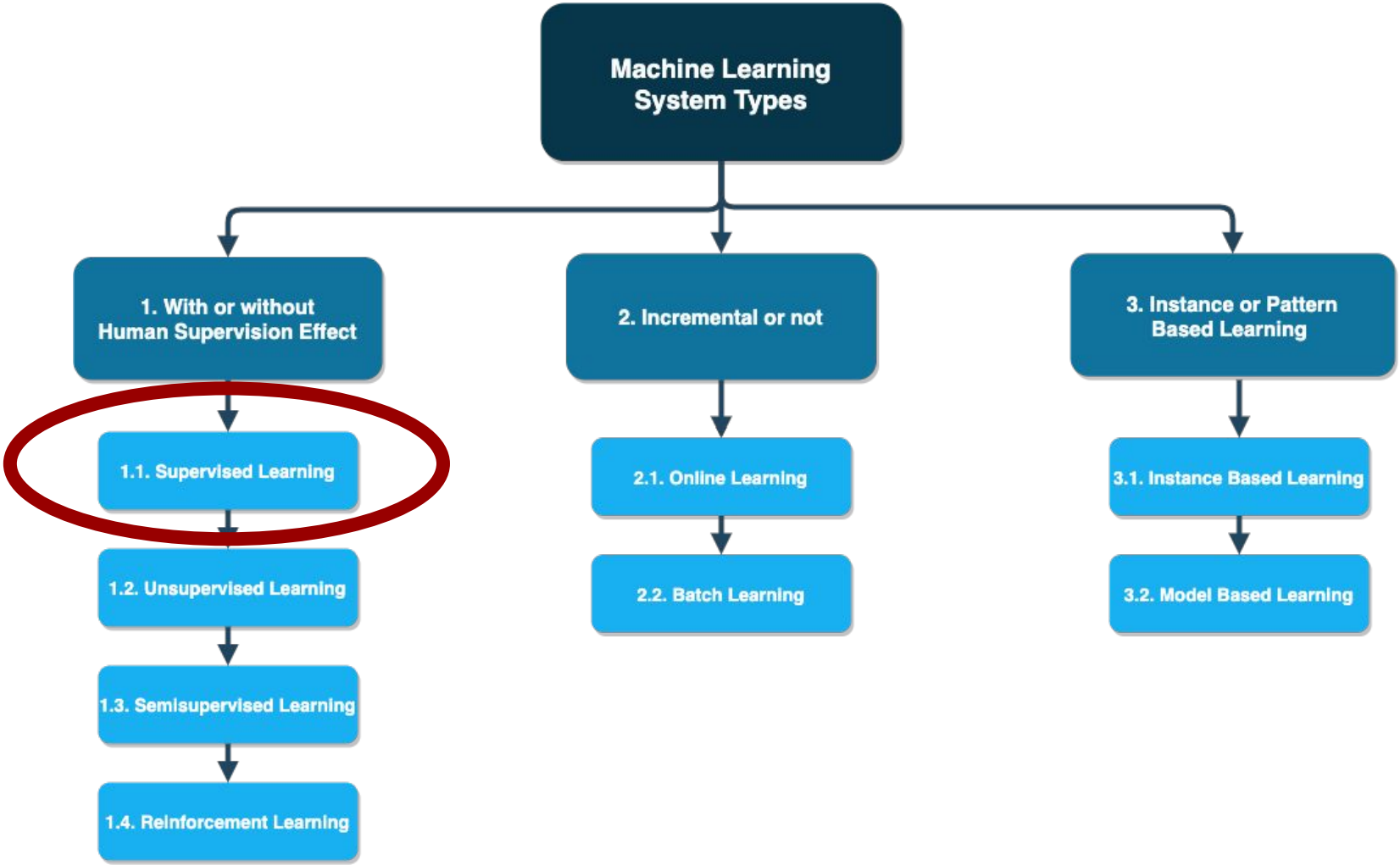
- We may write some constant rules (like rule1, rule2, rule3… etc) for a given task. **But**
  - Can we cover all of these rules?
  - Can these rules be updated over time?

- We use ML to extract these rules in an automated way.



Applications

- Image classification for products
- Document topic categorization
- Sentiment Analysis of text documents
- Breast cancer prediction
- Chatbots and Q/A systems
- Forecasting
- Fraud detection for credits
- Recommendation systems (Netflix, Amazon, Youtube, Spotify)
- And many other examples


Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Supervised Learning

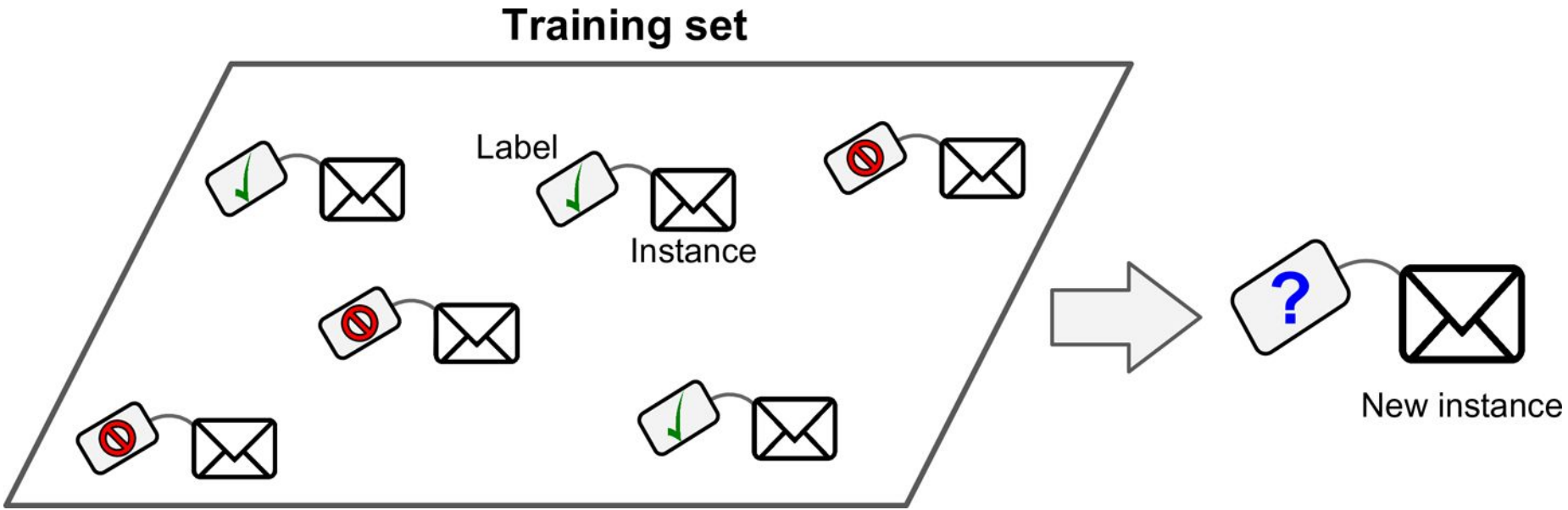
- We need to provide desired solution (called as **label** or **target**) for each training instance to compose a model.

    1. **Classification Task:** When the label is categorical value for a given set of *features* (attributes). Eg. Fraud Detection Problem

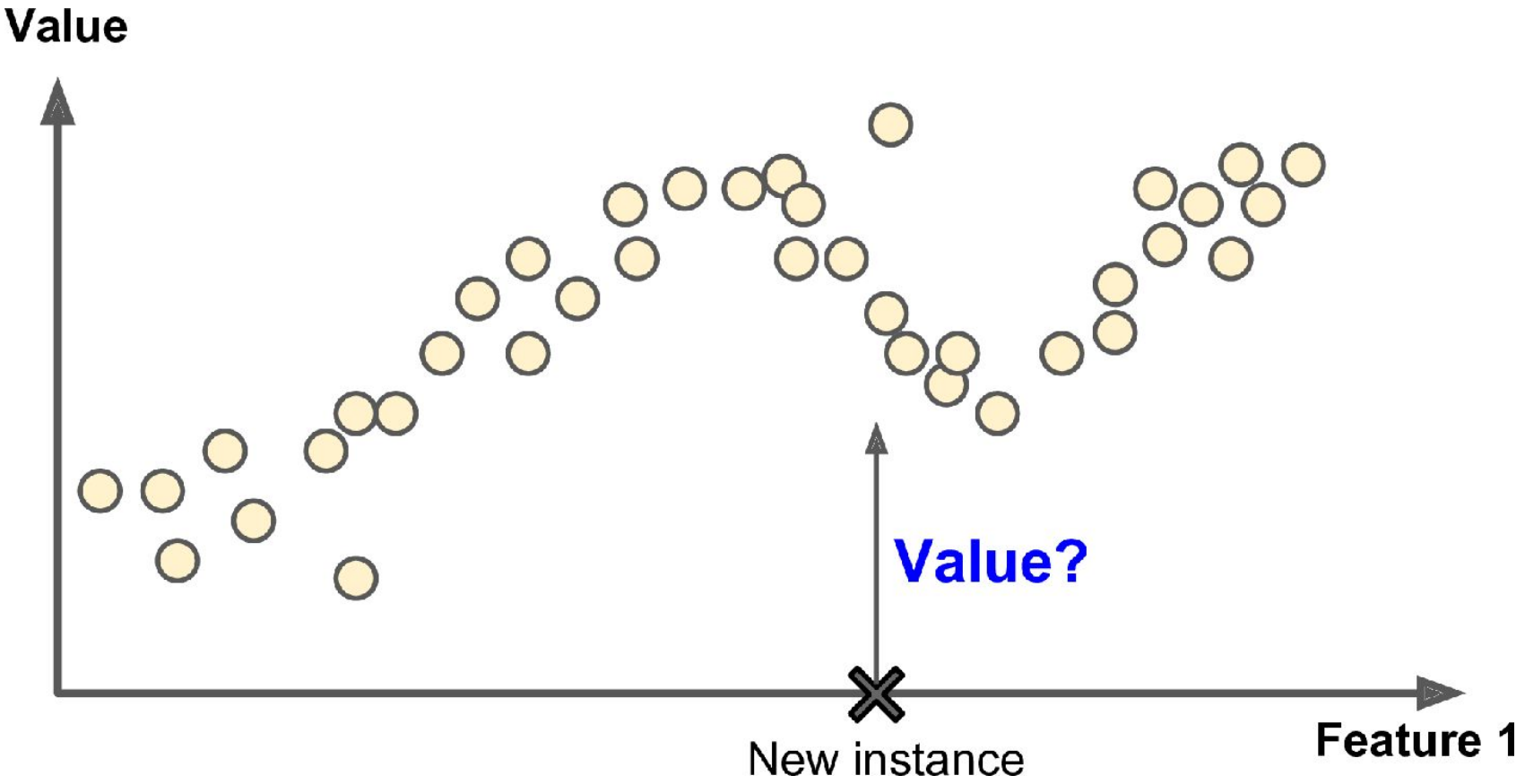
    2. **Regression Task:** When the label is numeric value for a given set of *features* (attributes). Eg. Predicting Price of a Car

Note: The labels can be numeric values like 0 or 1 in a classification task. In such a problem, we try to predict a label as either 0 or 1. If our prediction can be any value between 0 and 1 (like 0.15), then this problem becomes like a regression task.

Supervised Learning (Classification)



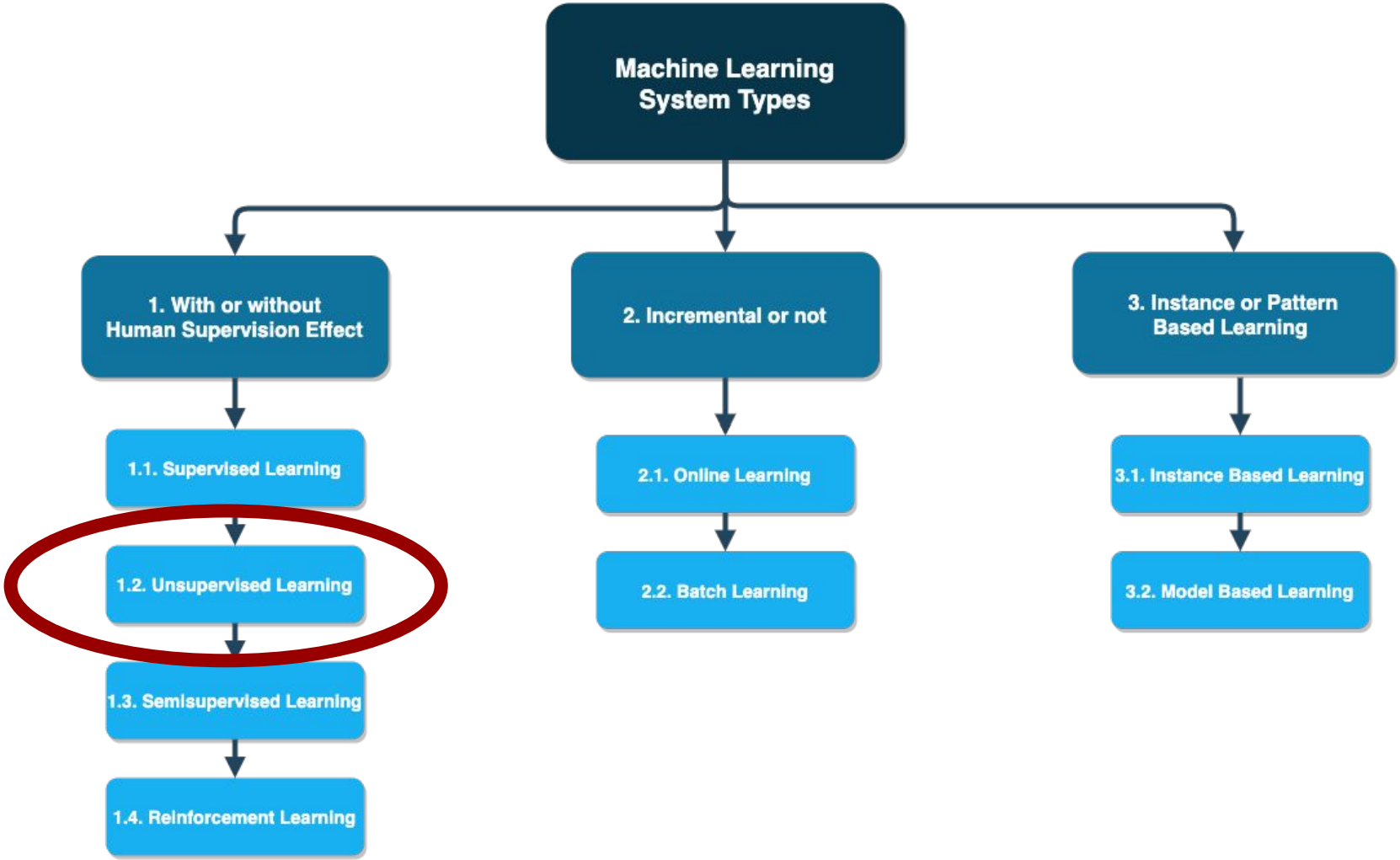
Supervised Learning (Regression)



Supervised Learning

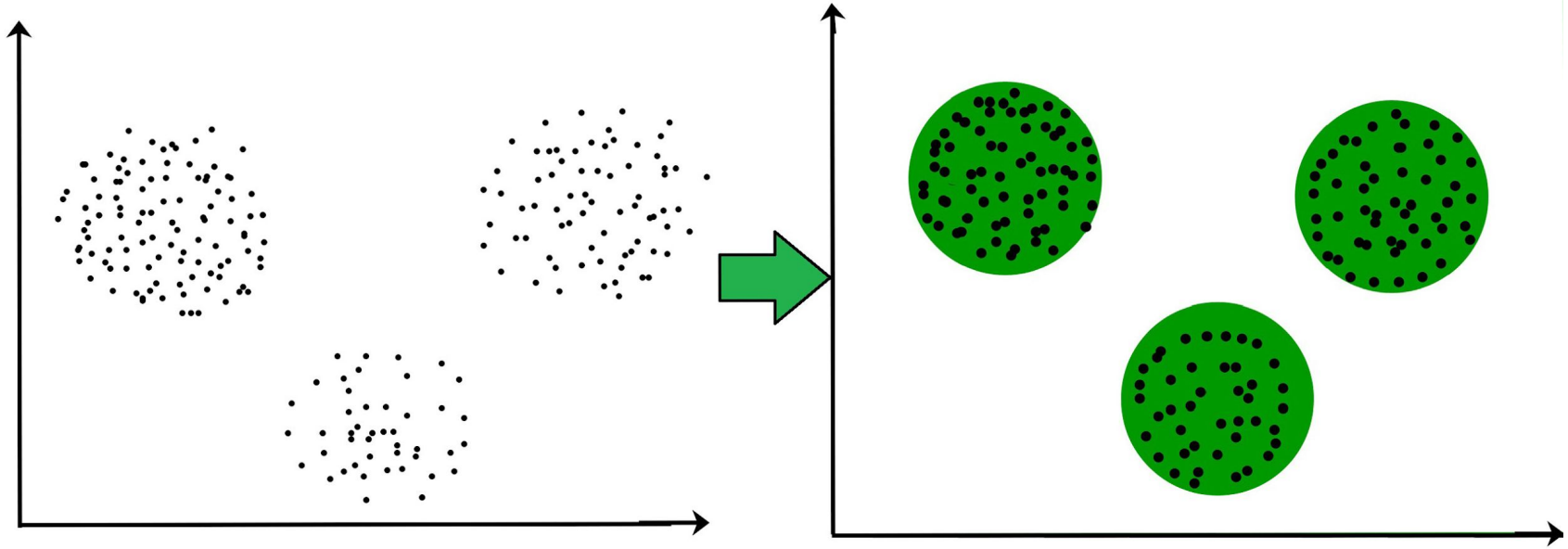
Some supervised learning methods:

- k-NN (k-Nearest Neighbors)
- Linear Regression
- Logistic Regression
- Decision Tree
- Random Forests
- SVMs (Support Vector Machines)
- ANNs (Artificial Neural Networks)


Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Unsupervised Learning

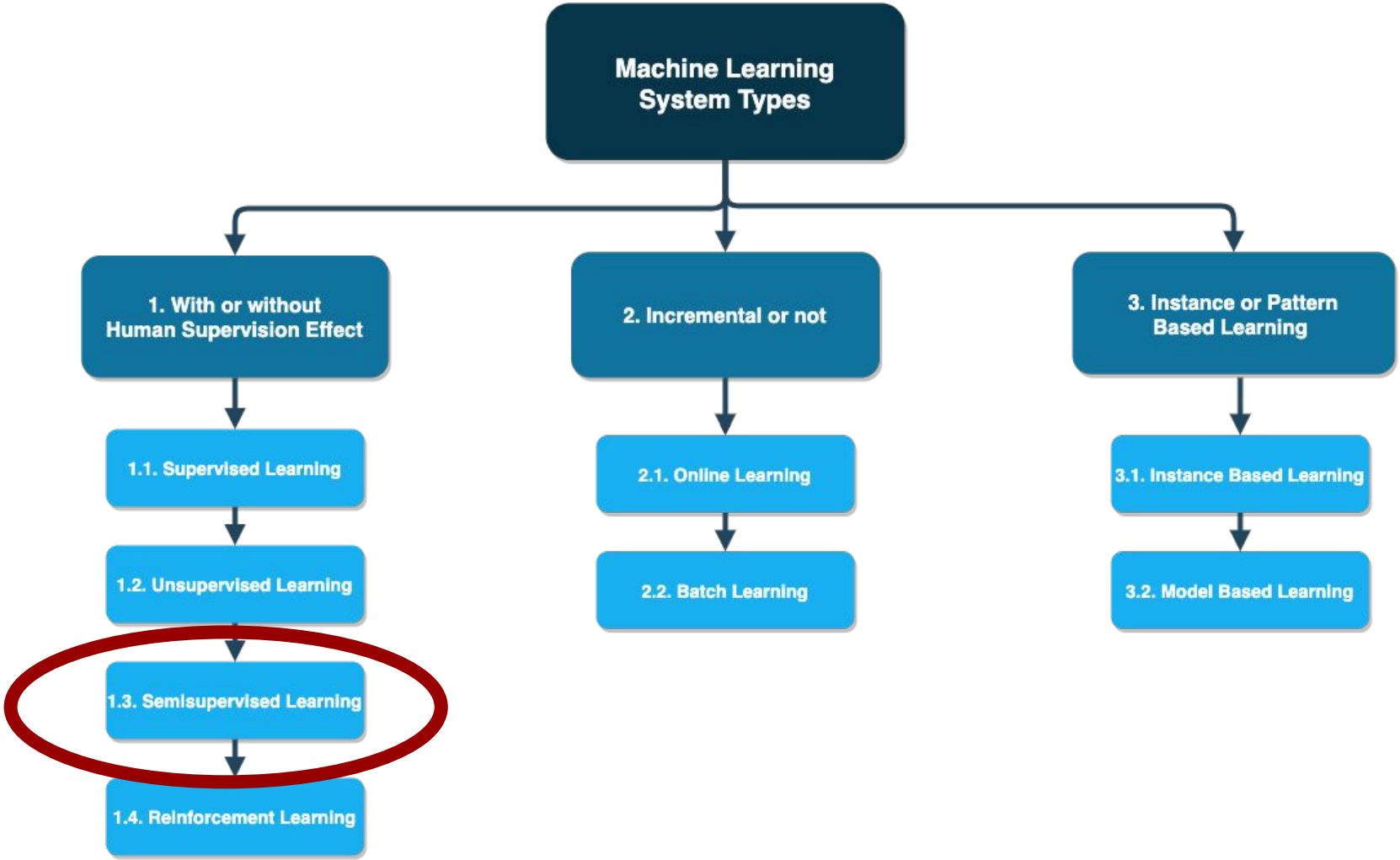
We do **not** use (maybe do **not** have) **the labels** in our training data for unsupervised learning.



Unsupervised Learning

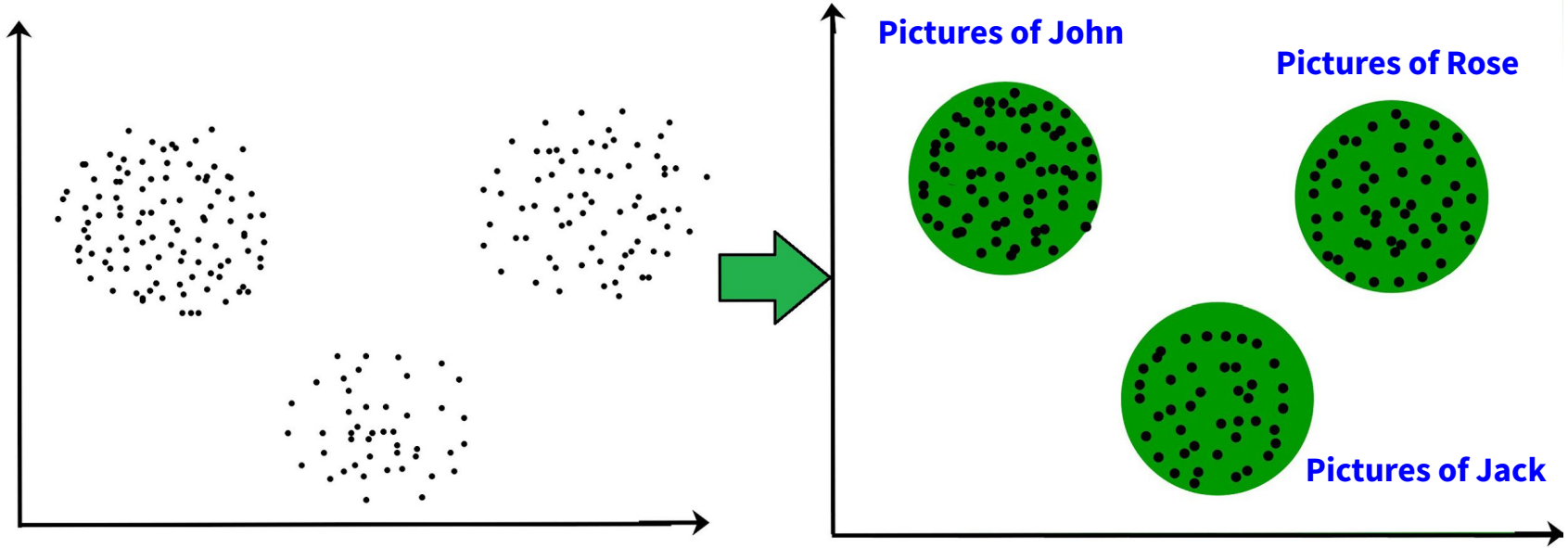
Some unsupervised learning methods:

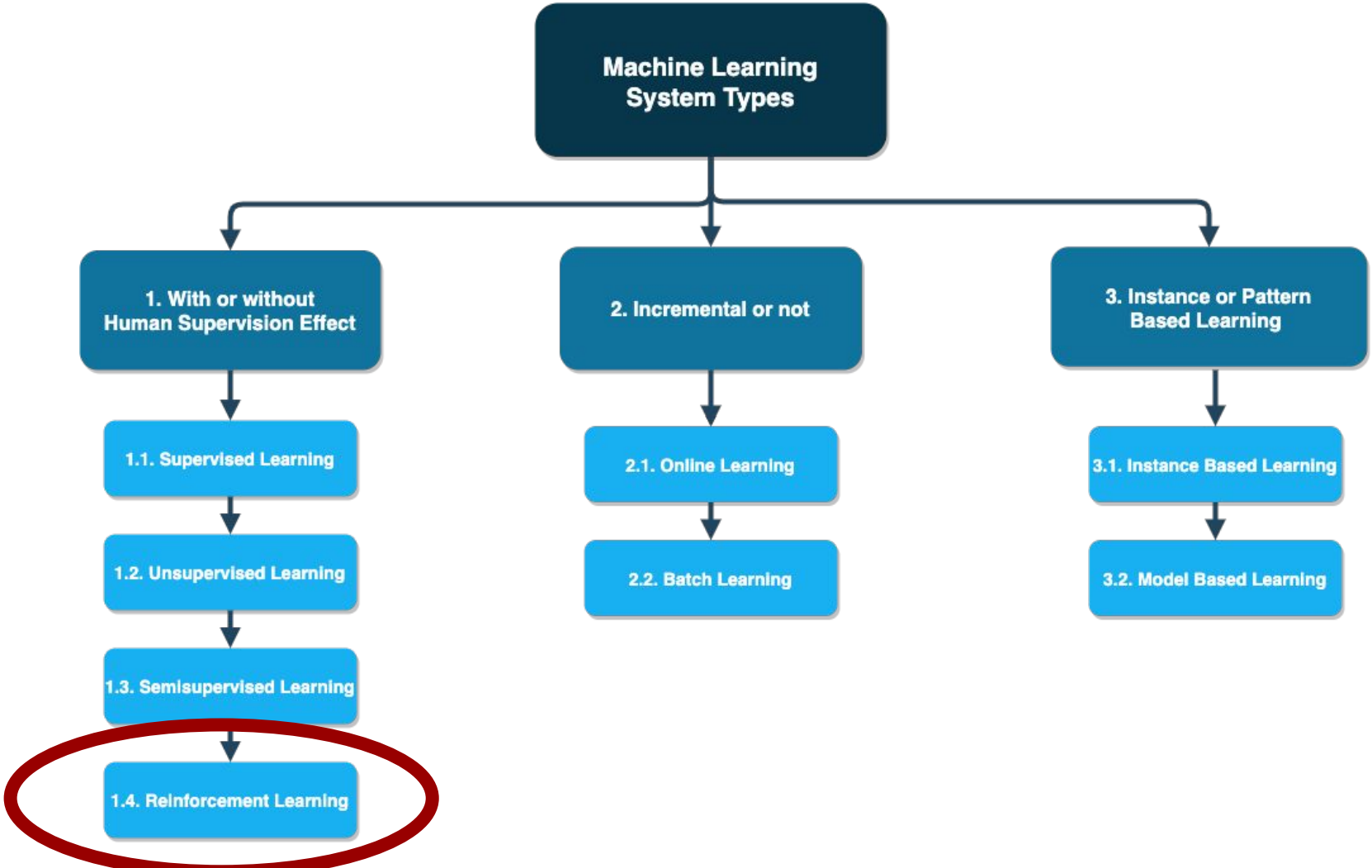
- **Clustering** → KMeans, Agglomerative Clustering, DBSCAN, etc
- **Dimensionality Reduction** → Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE)
- **Association Rule Mining** → Apriori algorithm
- **Anomaly Detection** → One-class SVM


Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Semisupervised Learning

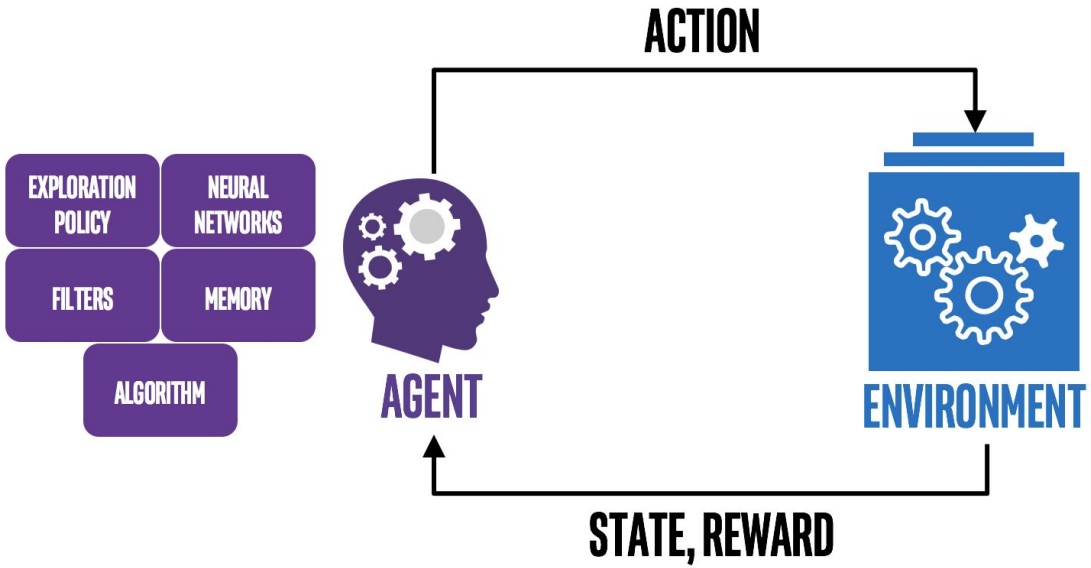
When we have only few labelled instances, we might use some intelligent ways to label more number of instances in an automatic way (like Google Photos is doing).

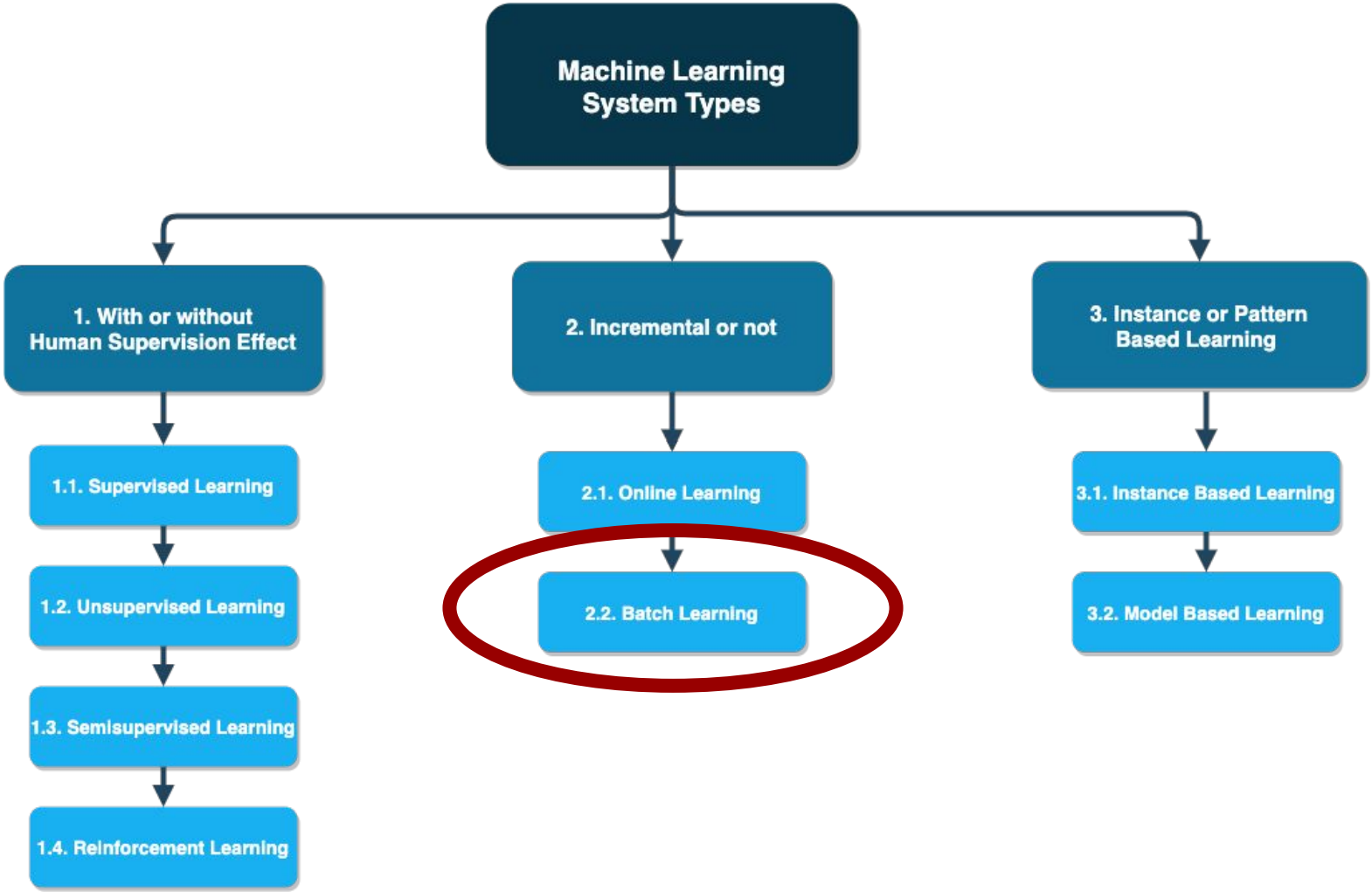



Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Reinforcement Learning

It’s completely different learning method which is based on performing some actions in an environment, and getting rewards/penalties in return.



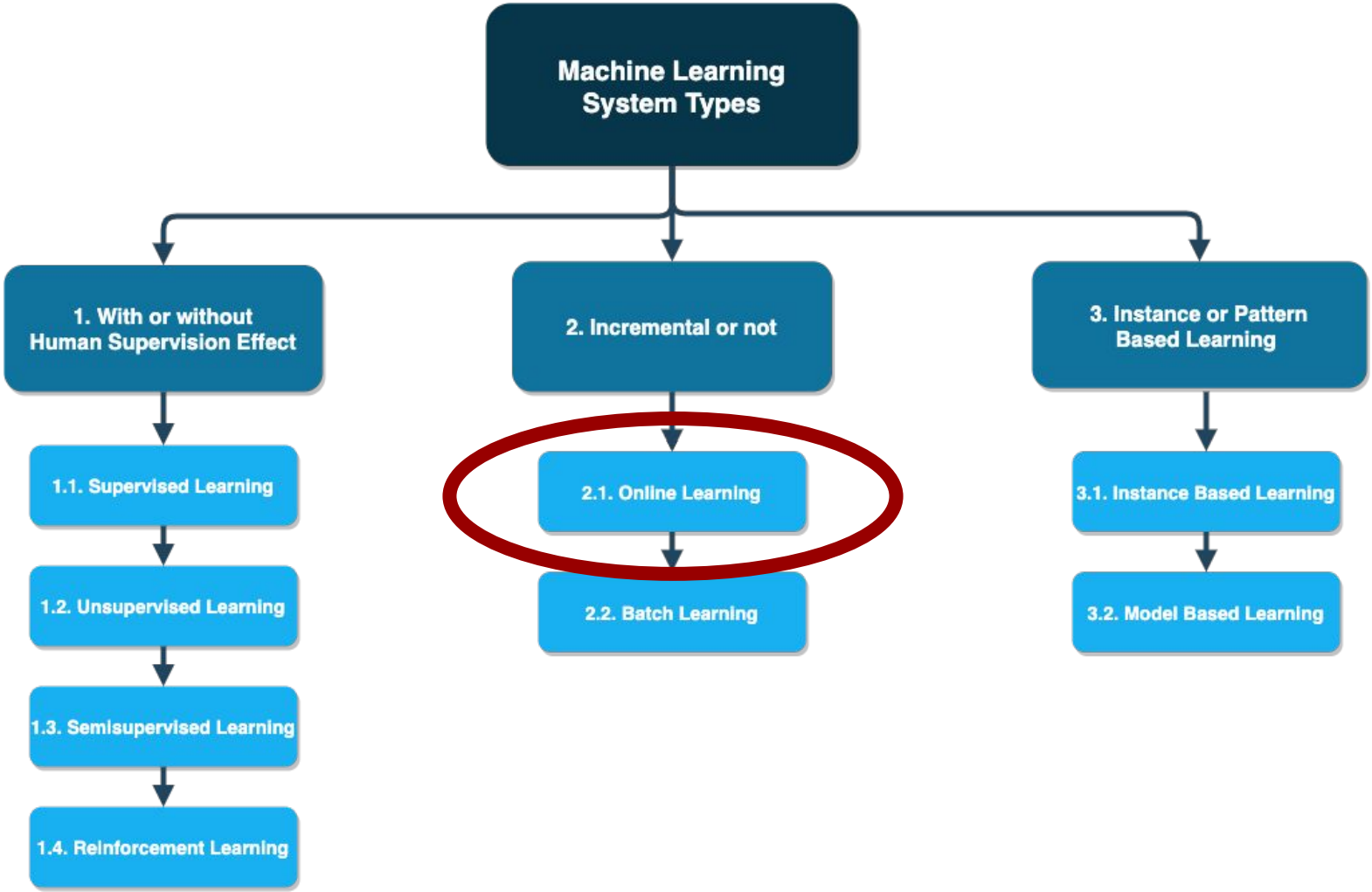

Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Batch Learning

- The system needs to use **all available data** in order to compose a model.

- Once we have additional labelled data, we should use the previous and new coming data together and learn from scratch.

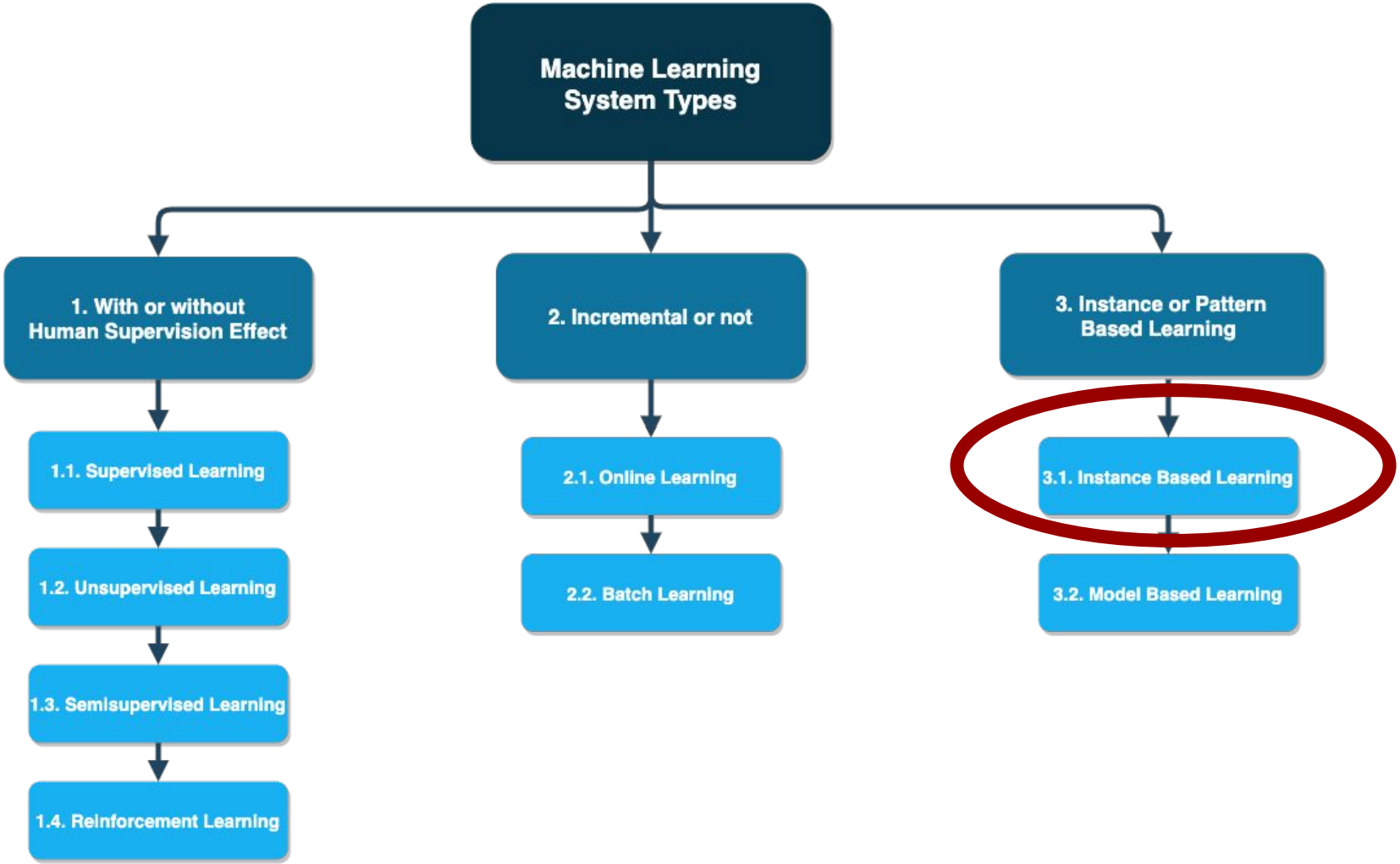
- Learning requires huge computing resources, however this process can be automated.


Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Online Learning

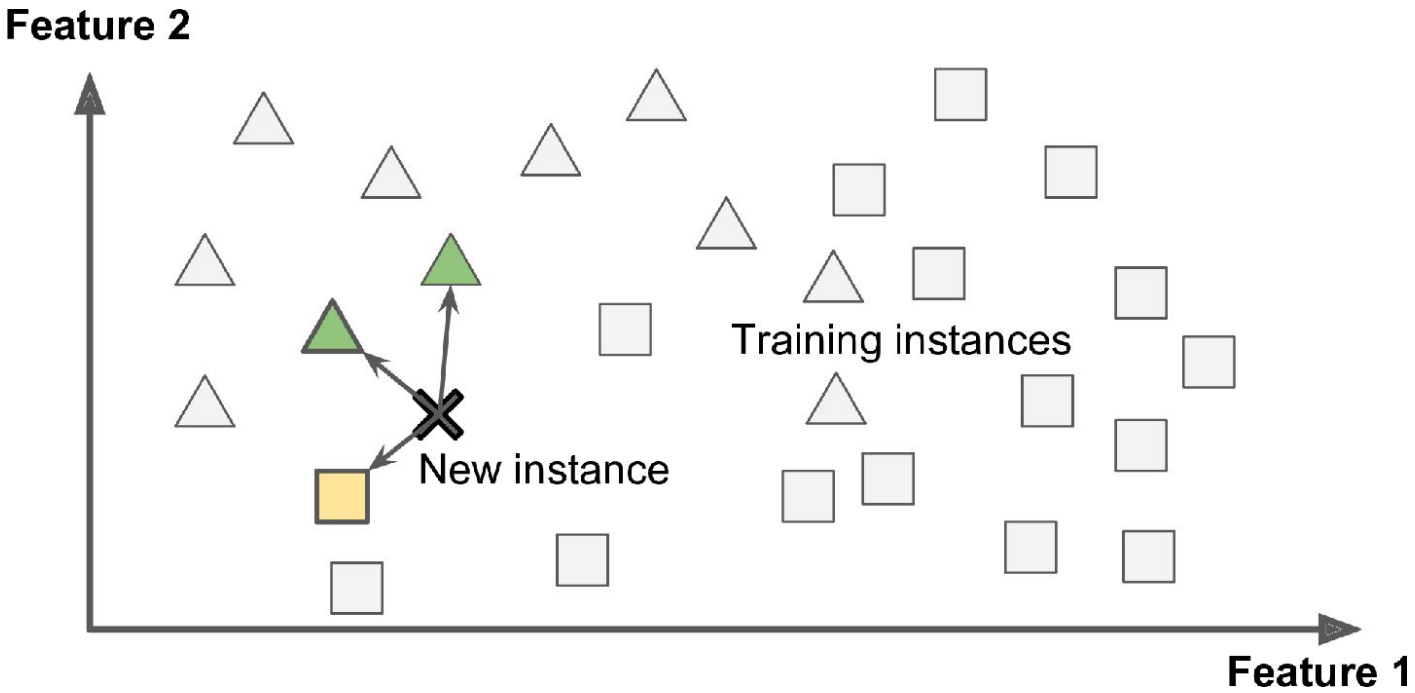
- In *online learning*, we learn the data incrementally. It can be done in 2 ways:
  - We can learn the new data on the fly (as we have them)
  - We can learn the data with mini-batches (it's also offline, maybe need to be called as *incremental learning*)

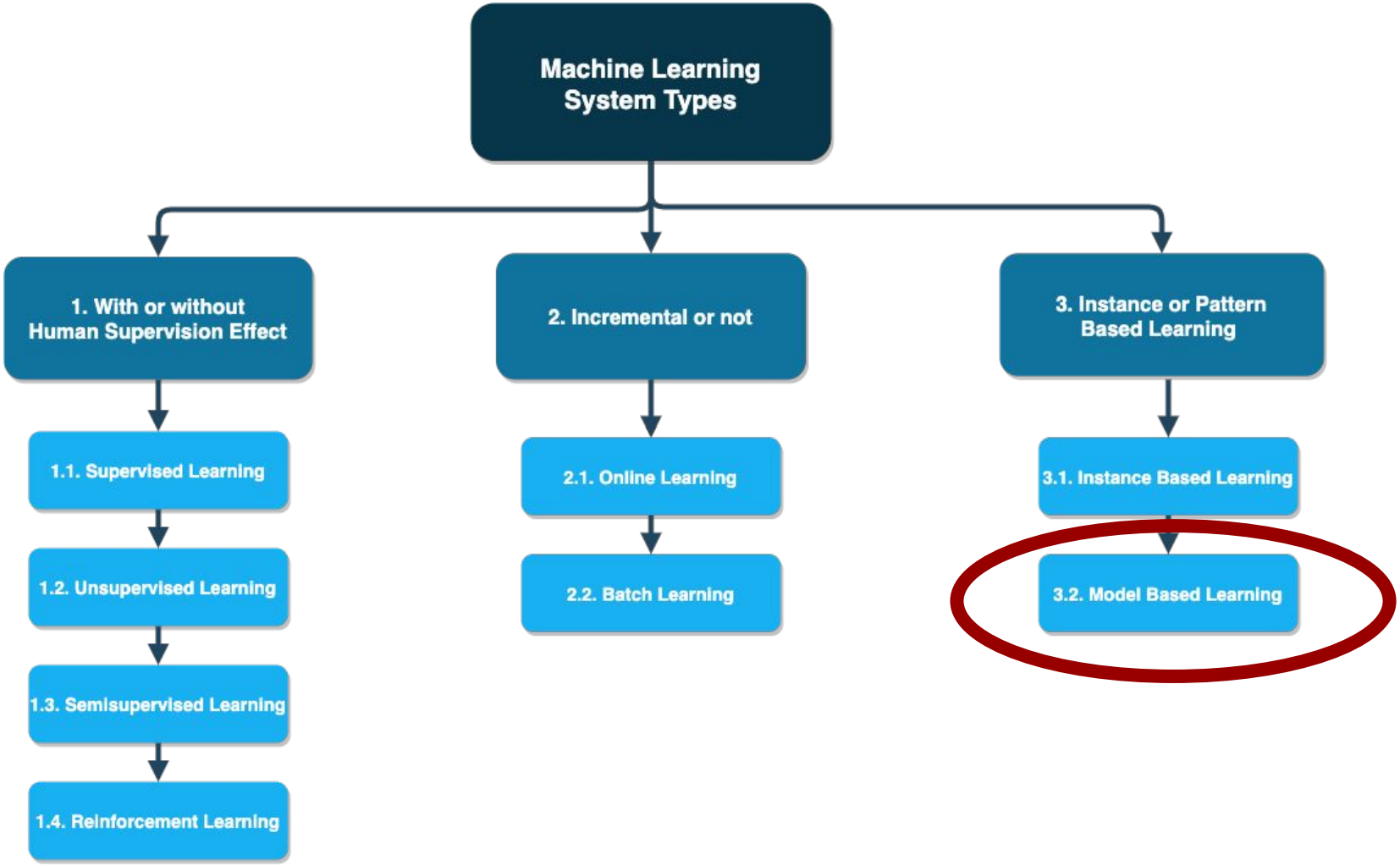
- *Incremental learning* can be used when we have huge amount of data that cannot fit into the memory (called as out-of-core learning).


Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Instance Based Learning

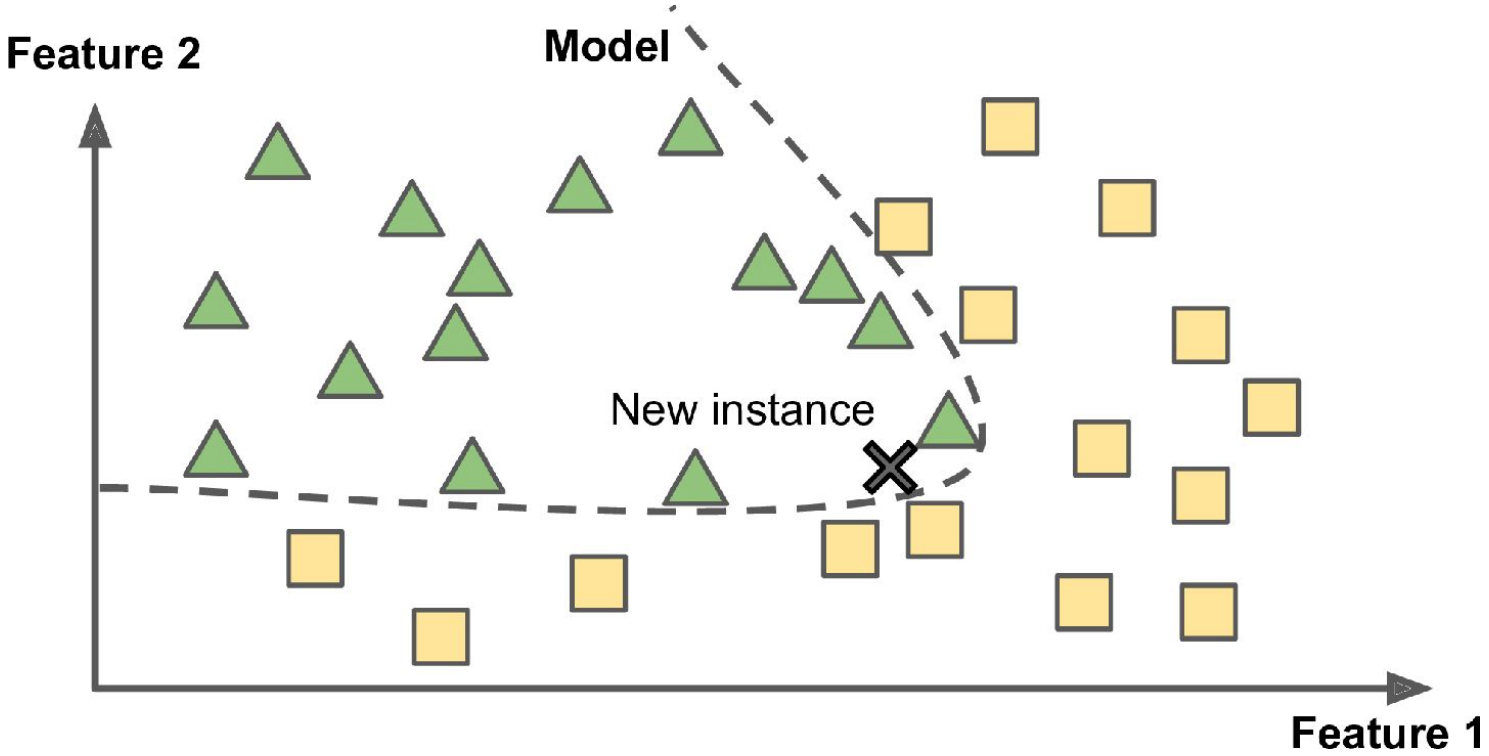
- A similarity metric is used in *Instance Based Learning* to make some predictions.




Machine Learning System Types
1. With or without Human Supervision Effect
2. Incremental or not
3. Instance or Pattern Based Learning
1.1. Supervised Learning
1.2. Unsupervised Learning
1.3. Semisupervised Learning
1.4. Reinforcement Learning
2.1. Online Learning
2.2. Batch Learning
3.1. Instance Based Learning
3.2. Model Based Learning

Model Based Learning

- Aim is to generalize learning model to make some predictions.



Main Challenges in ML

- Bad Data
  - Insufficient Amount of Data
  - Nonrepresentative Training Data (missing representatives)
  - Poor Quality Data (like wrong labels)
  - Irrelevant Features (Feature selection, feature extraction)

- Bad Algorithm
  - Overfitting
  - Underfitting

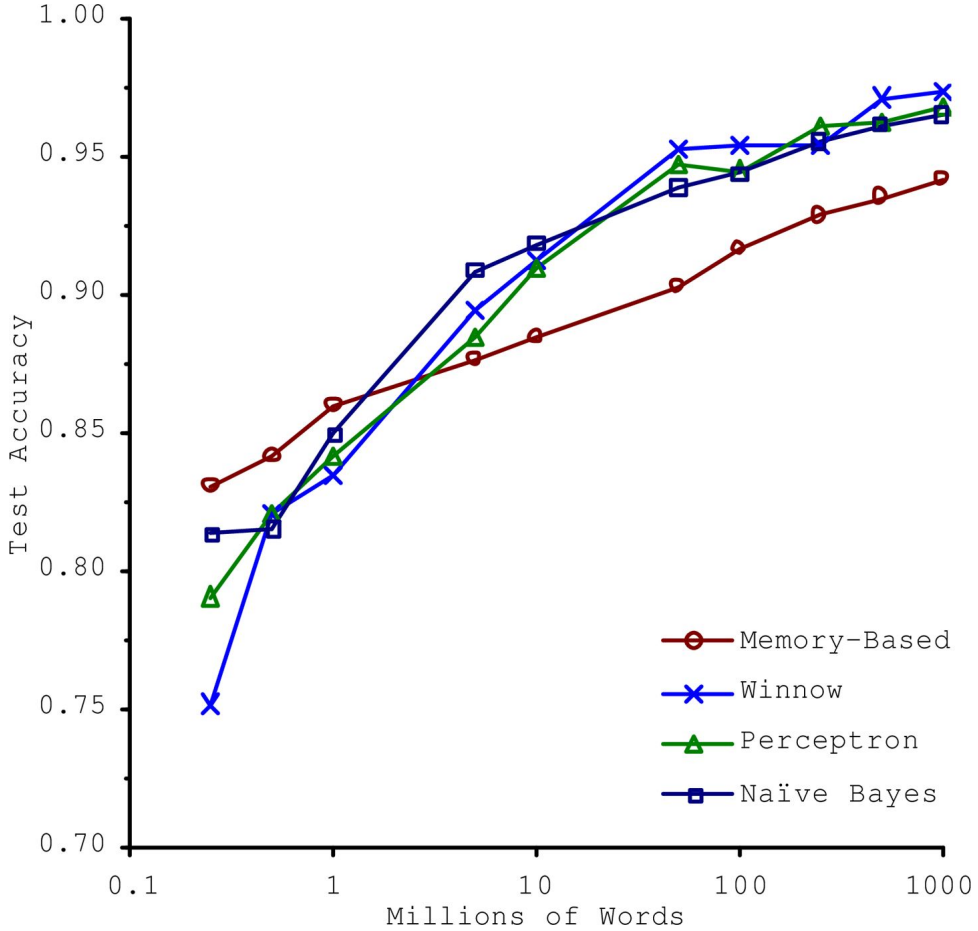
Insufficient Amount of Data

- The Unreasonable Effectiveness of Data

- Even for a simple task, we need thousands of training instance. This number can scale up to millions depending on
  - the task
  - the complexity

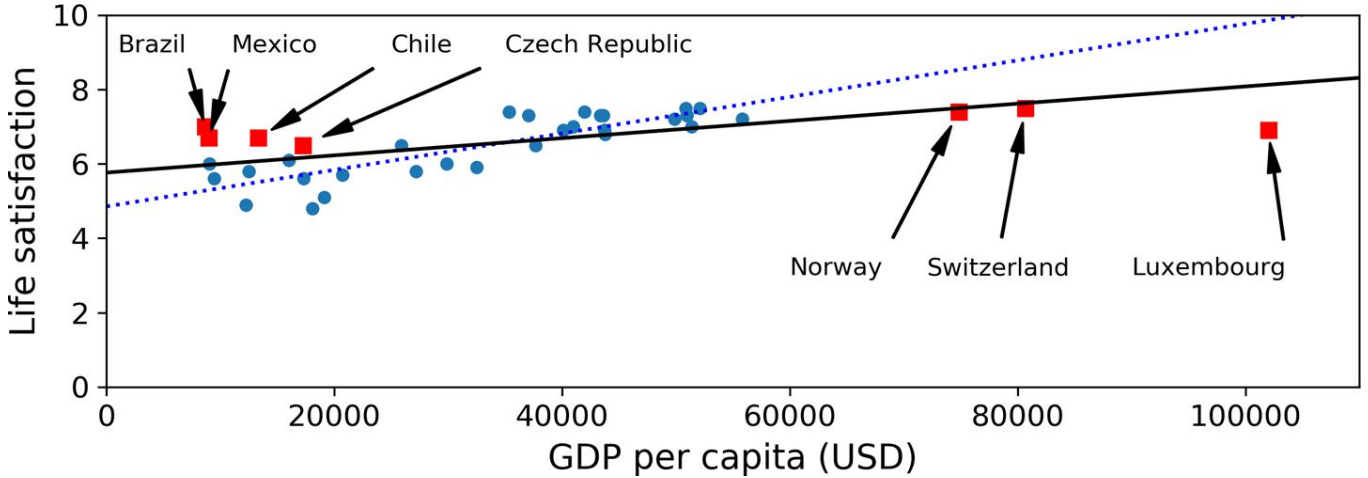
- Dilemma → investing on
  - better algorithm or
  - more data?

Insufficient Amount of Data



Nonrepresentative Training Data

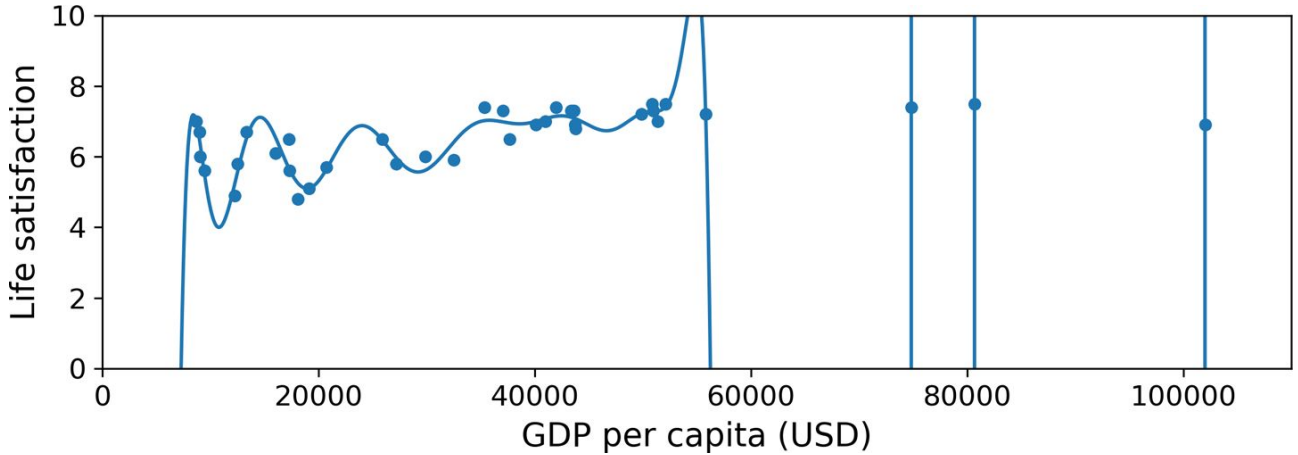
When we have missing records in our training data, then our generalized model might not be representative.



- 1936 Election Poll (Landon vs Roosevelt) (or this link) → Sampling Bias

Overfitting

- Sometimes we learn (actually memorize) training data so well, however we cannot have the same performance on unseen data. In other words, the learning model does not generalize well. This is called *overfitting*.



Overfitting

- Possible solutions once we have overfitting
  - Increase the amount of training data
  - Simplify the model
    - Reduce the complexity of the model (eg. use linear model instead of higher-degree polynomial model or update the parameter like in the decision tree)
    - Reduce the number of features
  - Regularization (restricting some of the parameters)

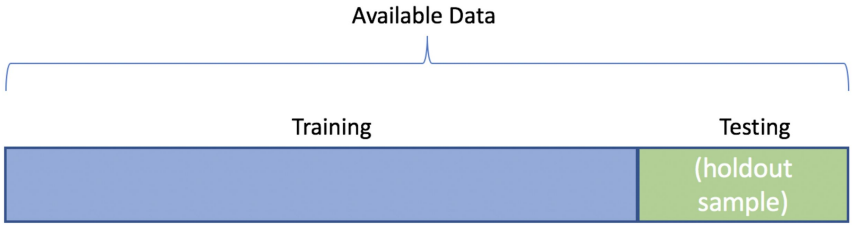
Underfitting

- *Underfitting* is just opposite of *overfitting*. Our learning model cannot learn well, because it’s too simple.

- Once we have *underfitting*, we can
  - Try to use a more complex algorithm with more parameter
  - Increase the number of features
  - Reduce regularization

Evaluation

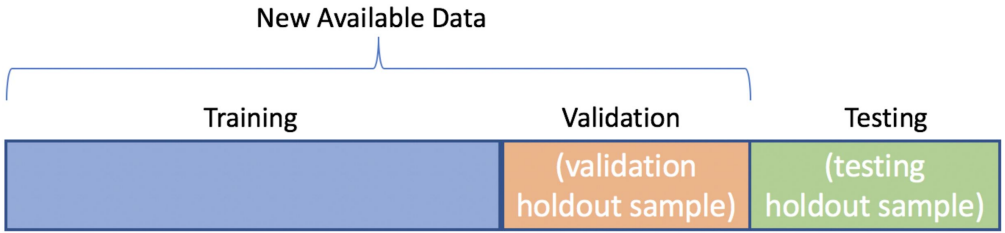
- How can we test our model? How can we understand whether we have overfitting/underfitting?
- Train all the data, and directly launch it to the customers without evaluating? **NO!**
- Train all the data, and evaluate on the same data? **NO!**
- We should split our all available (labelled) data into **train** and **test** sets. We can try to learn the train data, and then evaluate this model on the test data.



- Still, it's not the perfect way. We might find the good parameters just for the testing; however, it might not be a good generalization.

Evaluation

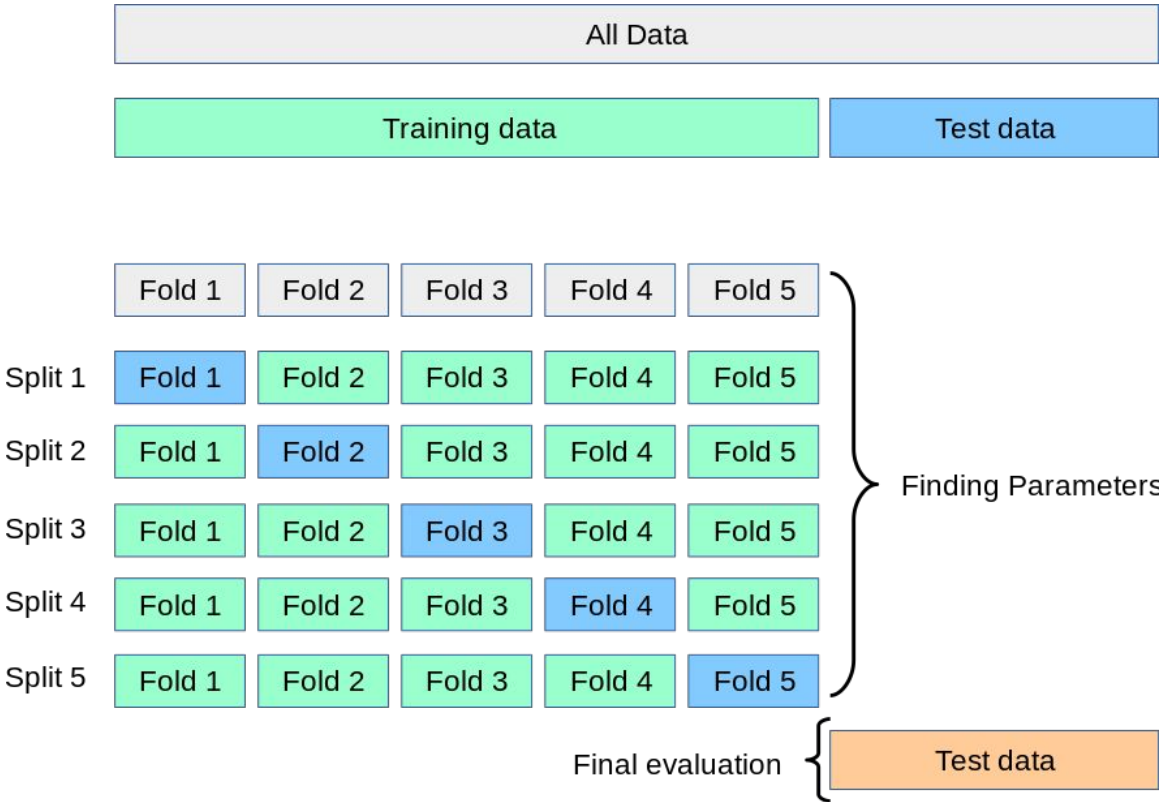
- A better solution would be having a **validation set** as well.



  - We will again
    - learn the training data
    - then evaluate on validation data
      - If results are not satisfactory, tweak the parameters (or change algorithm), and go back step 1.
    - After finding the best model for the validation data, evaluate it as well on test (unseen) data.
  - Although this technique generally works well, we might have some problems if the validation data is not large enough.

Evaluation

- We could use another technique called as **cross-validation** (check this source).



Evaluation

- Let’s mention some important remarks:
  - We need to make sure that test and validation sets represents the real world as much as possible (class distributions etc).
  - Shuffling the data before splitting into train/val/test might be a good idea (if it’s not a time series problem).
    - Random seed can be used in order to produce same experimental results.
  - Check this link for Stratified Cross-Validation.