ML: Supervised Learning

Tassadaq Hussain

Professor Namal University

Director Centre for AI and Big Data

Collaborations:

Barcelona Supercomputing Center Barcelona, Spain

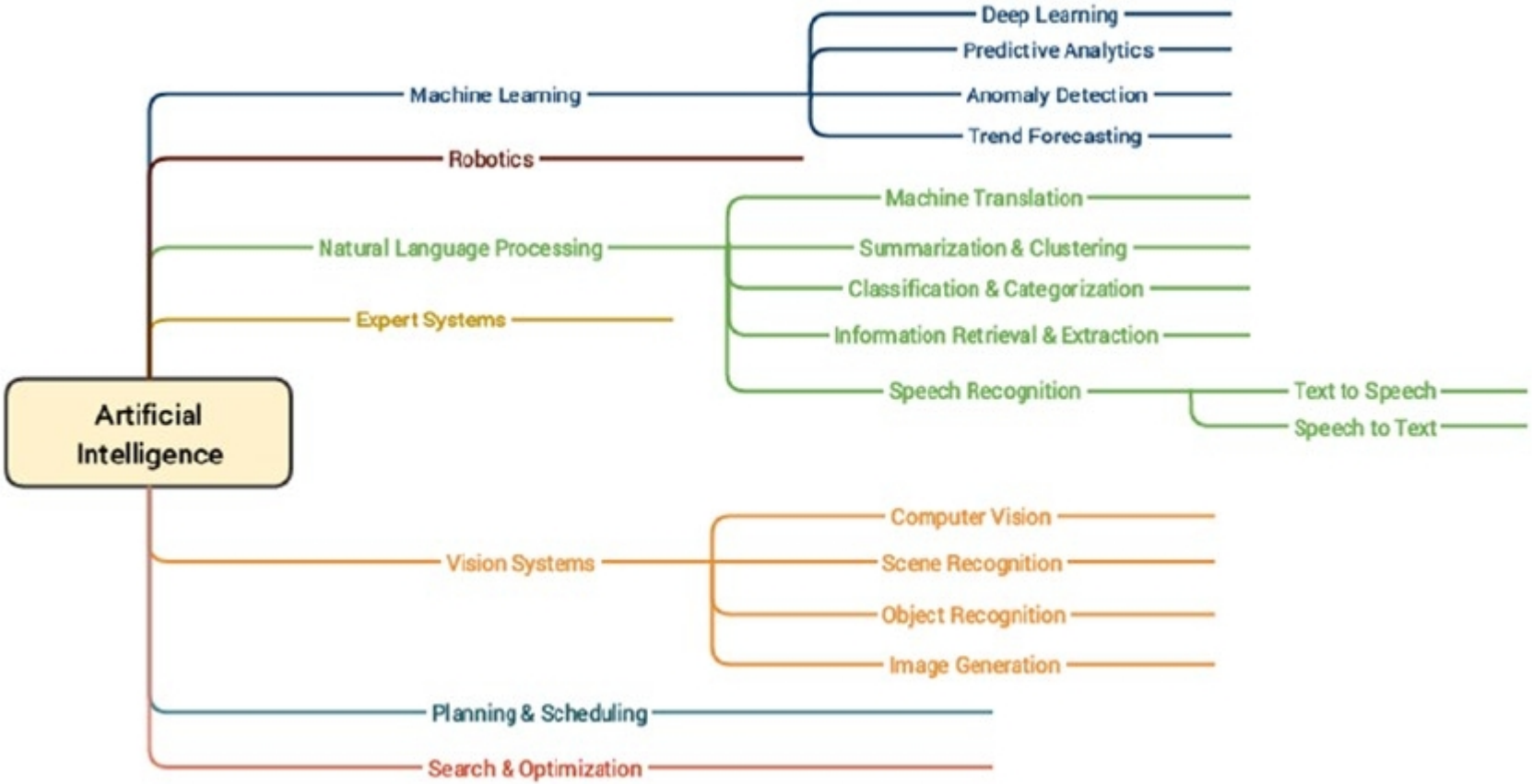
European Network on High Performance and Embedded Architecture and Compilation

Pakistan Supercomputing Center

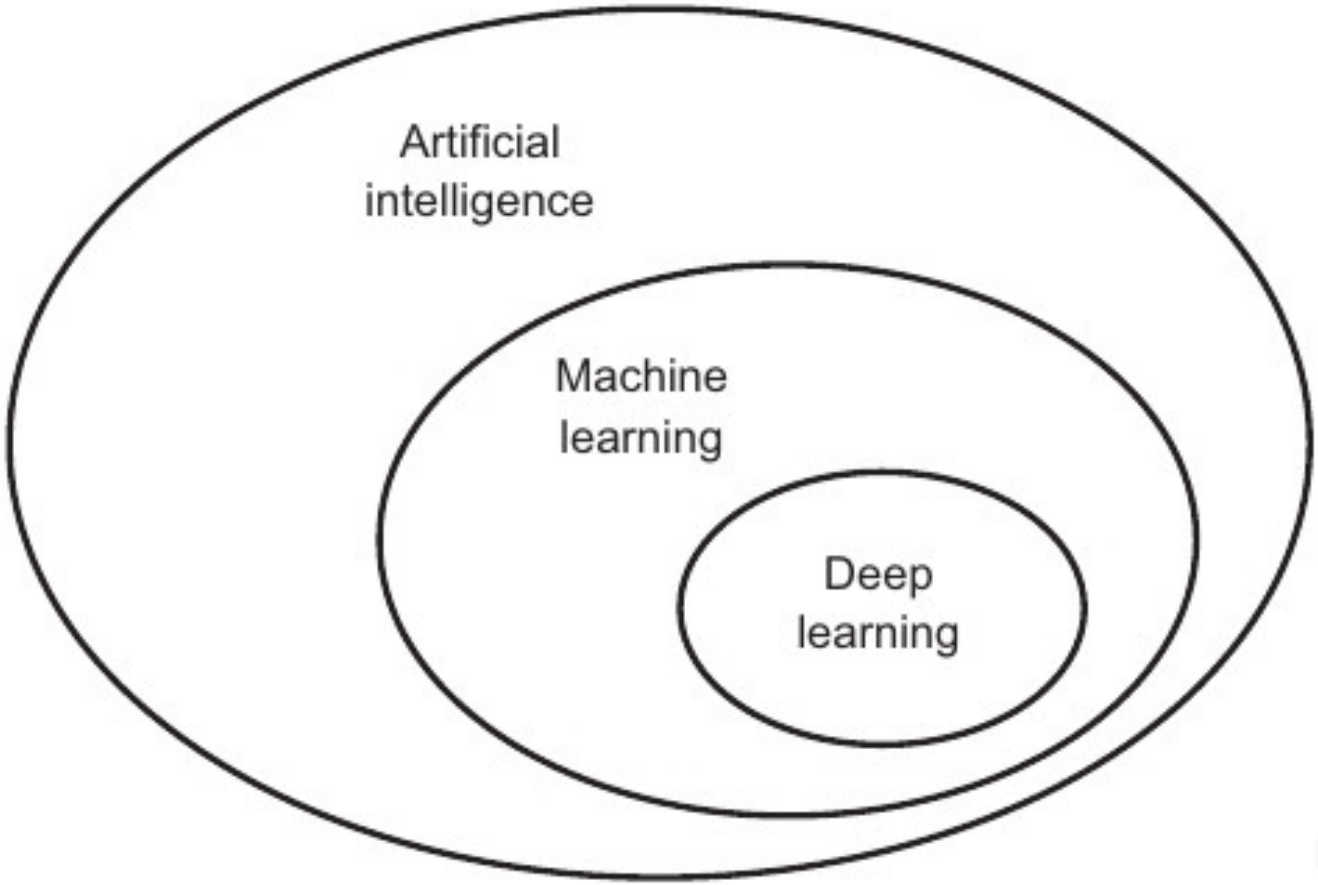
UCERD

Gathering
Intellectuals

www.ucerd.com

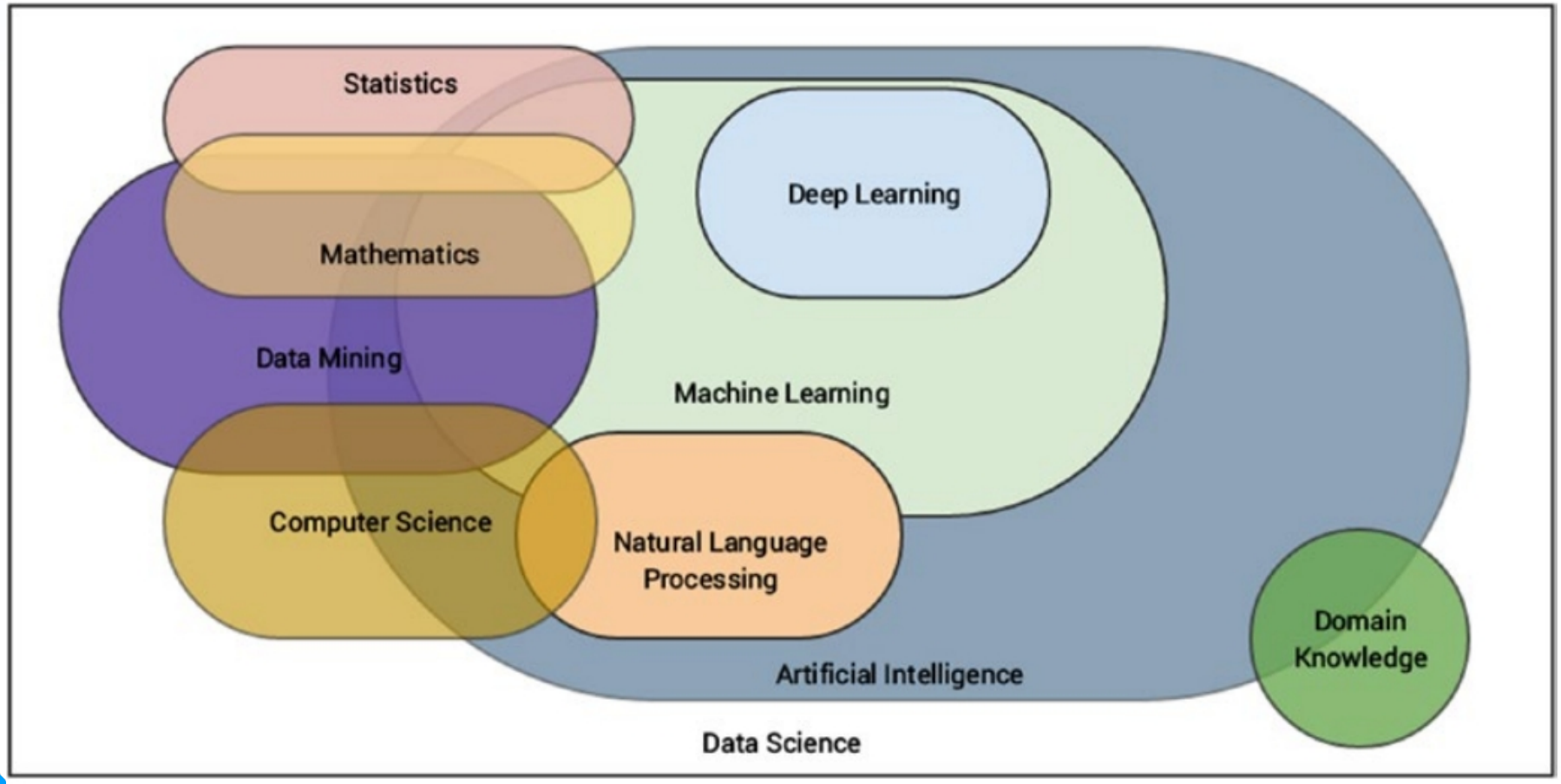

Artificial Intelligence
Machine Learning
Deep Learning
Predictive Analytics
Anomaly Detection
Trend Forecasting
Robotics
Natural Language Processing
Machine Translation
Summarization & Clustering
Classification & Categorization
Information Retrieval & Extraction
Speech Recognition
Text to Speech
Speech to Text
Expert Systems
Vision Systems
Computer Vision
Scene Recognition
Object Recognition
Image Generation
Planning & Scheduling
Search & Optimization

UCERD



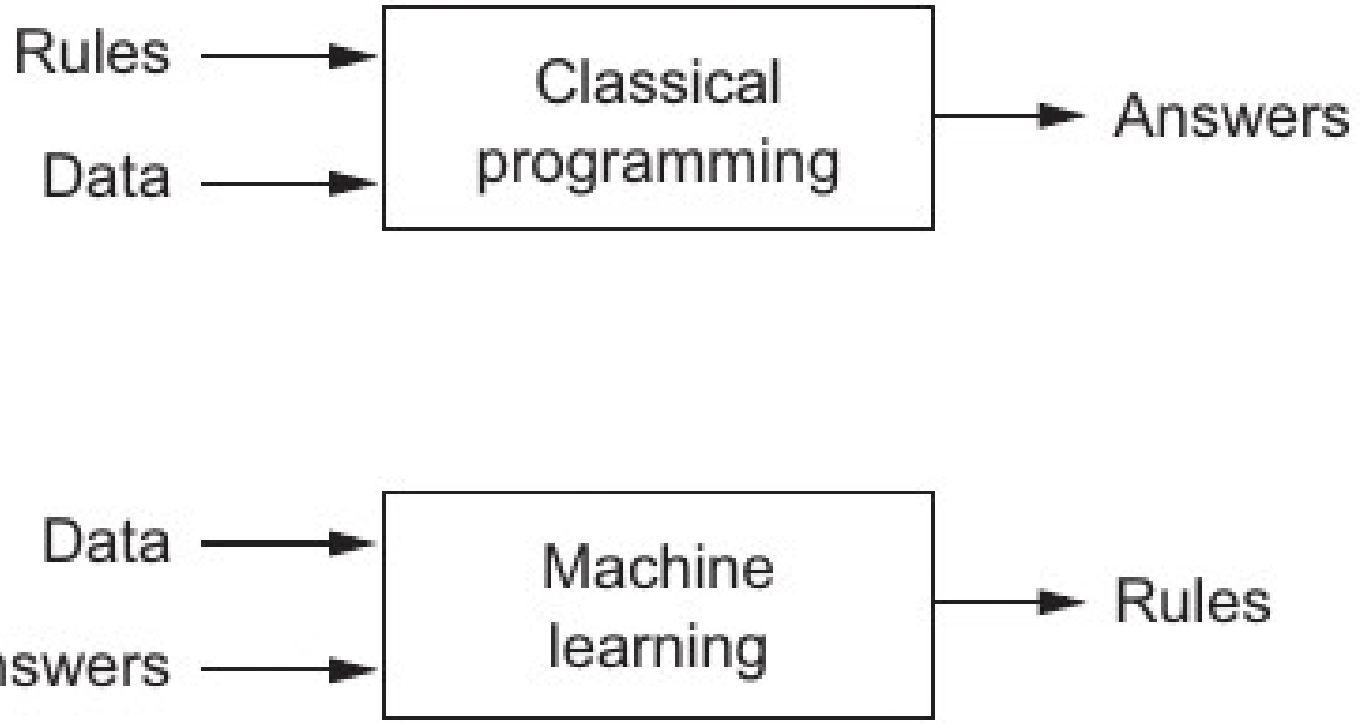
UCERD

ML: A Multi-disciplinary Field



UCERD

Conventional VS ML Programming



UCERD

Why Machine Learning

- Lack of sufficient human expertise in a domain (e.g., simulating navigations in unknown territories or even spatial planets).
- Scenarios and behavior can keep changing over time (e.g., availability of infrastructure in an organization, network connectivity, and so on).
- Humans have sufficient expertise in the domain but it is extremely difficult to formally explain or translate this expertise into computational tasks (e.g., speech recognition, translation, scene recognition, cognitive tasks, and so on).
- Addressing domain specific problems at scale with huge volumes of data with too many complex conditions and constraints.

UCERD

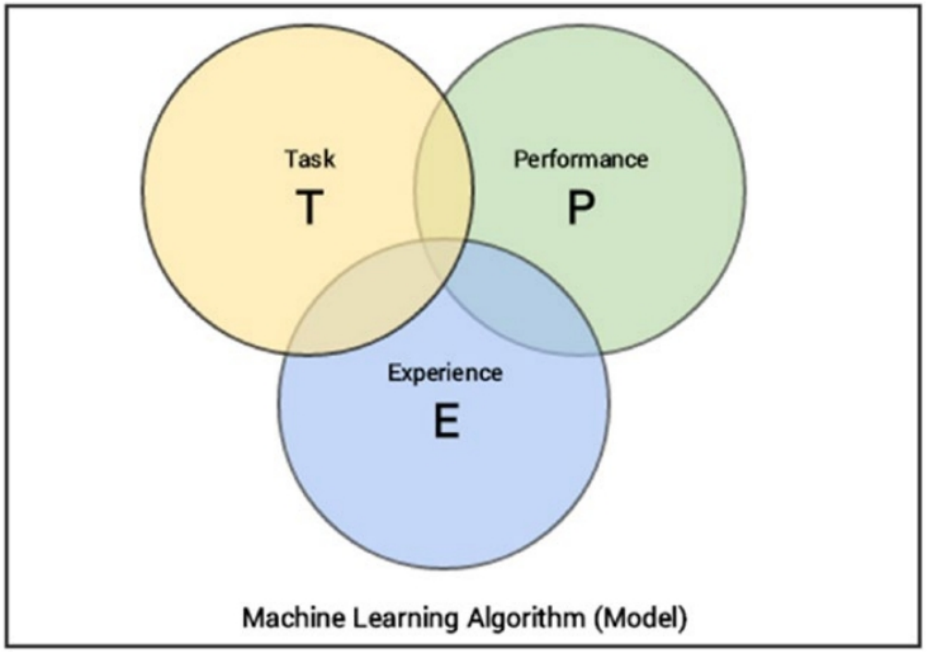
Contents

- Machine Learning Understanding
- **AI Basic Model**
- Machine Learning Methods
- Data Understanding

UCERD

AI Model

- Classification or categorization
- Regression
- Anomaly detection
- Structured annotation
- Translation
- Clustering or grouping
- Transcriptions



UCERD

Machine Learning Tasks

- Classification or categorization: This typically encompasses the list of problems or tasks where the machine has to take in data points or samples and assign a specific class or category to each sample. A simple example would be classifying animal images into dogs, cats, and zebras.
- Regression: These types of tasks usually involve performing a prediction such that a real numerical value is the output instead of a class or category for an input data point. The best way to understand a regression task would be to take the case of a real-world problem of predicting housing prices considering the plot area, number of floors, bathrooms, bedrooms, and kitchen as input attributes for each data point.
- Anomaly detection: These tasks involve the machine going over event logs, transaction logs, and other data points such that it can find anomalous or unusual patterns or events that are different from the normal behavior. Examples for this include trying to find denial of service attacks from logs, indications of fraud, and so on.

- Structured annotation: This usually involves performing some analysis on input data points and adding structured metadata as annotations to the original data that depict extra information and relationships among the data elements. Simple examples would be annotating text with their parts of speech, named entities, grammar, and sentiment. Annotations can also be done for images like assigning specific categories to image pixels, annotate specific areas of images based on their type, location, and so on.
- Translation: Automated machine translation tasks are typically of the nature such that if you have input data samples belonging to a specific language, you translate it into output having another desired language. Natural language based translation is definitely a huge area dealing with a lot of text data.
- Clustering or grouping: Clusters or groups are usually formed from input data samples by making the machine learn or observe inherent latent patterns, relationships and similarities among the input data points themselves. Usually there is a lack of pre-labeled or pre-annotated data for these tasks hence they form a part of unsupervised Machine Learning (which we will discuss later on). Examples would be grouping similar products, events and entities.
- Transcriptions: These tasks usually entail various representations of data that are usually continuous and unstructured and converting them into more structured and discrete data elements. Examples include speech to text, optical character recognition, images to text, and so on.

AI Categories

Machine Learning has multiple algorithms, techniques, and methodologies that can be used to build models to solve real-world problems using data. This section tries to classify these Machine Learning methods under some broad categories to give some sense to the overall landscape of Machine Learning methods that are ultimately used to perform specific Machine Learning tasks we discussed in a previous section. Typically the same Machine Learning methods can be classified in multiple ways under multiple umbrellas. Following are some of the major broad areas of Machine Learning methods.

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Reinforcement learning

UCERD

Supervised Learning

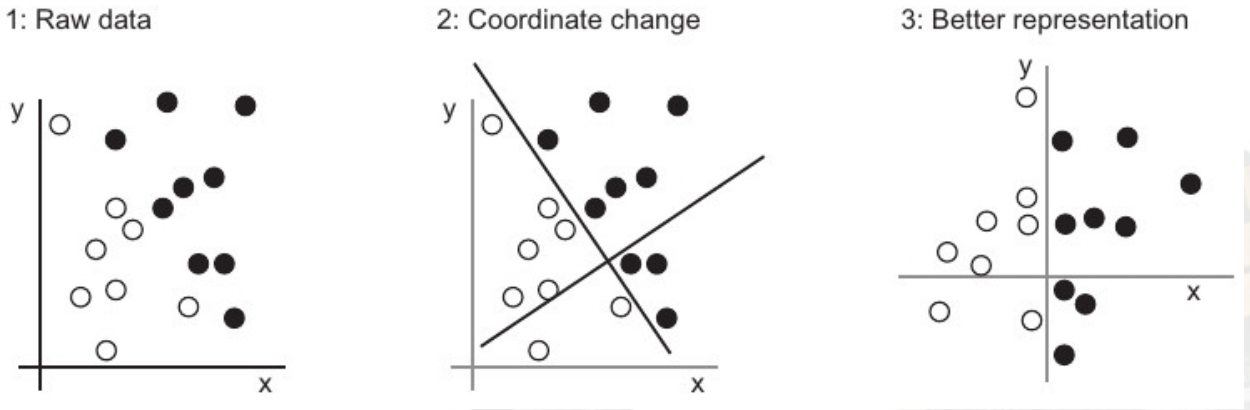
- Supervised learning methods or algorithms include learning algorithms that take in data samples (known as training data) and associated outputs (known as labels or responses) with each data sample during the model training process. The main objective is to learn a mapping or association between input data samples x and their corresponding outputs y based on multiple training data instances. This learned knowledge can then be used in the future to predict an output y′ for any new input data sample x′ which was previously unknown or unseen during the model training process. These methods are termed as supervised because the model learns on data samples where the desired output responses/labels are already known beforehand in the training phase.

UCERD

Supervised Learning

- Supervised learning methods or algorithms include learning algorithms that take in data samples (known as training data) and associated outputs (known as labels or responses) with each data sample during the model training process. The main objective is to learn a mapping or association between input data samples x and their corresponding outputs y based on multiple training data instances. This learned knowledge can then be used in the future to predict an output y′ for any new input data sample x′ which was previously unknown or unseen during the model training process. These methods are termed as supervised because the model learns on data samples where the desired output responses/labels are already known beforehand in the training phase.

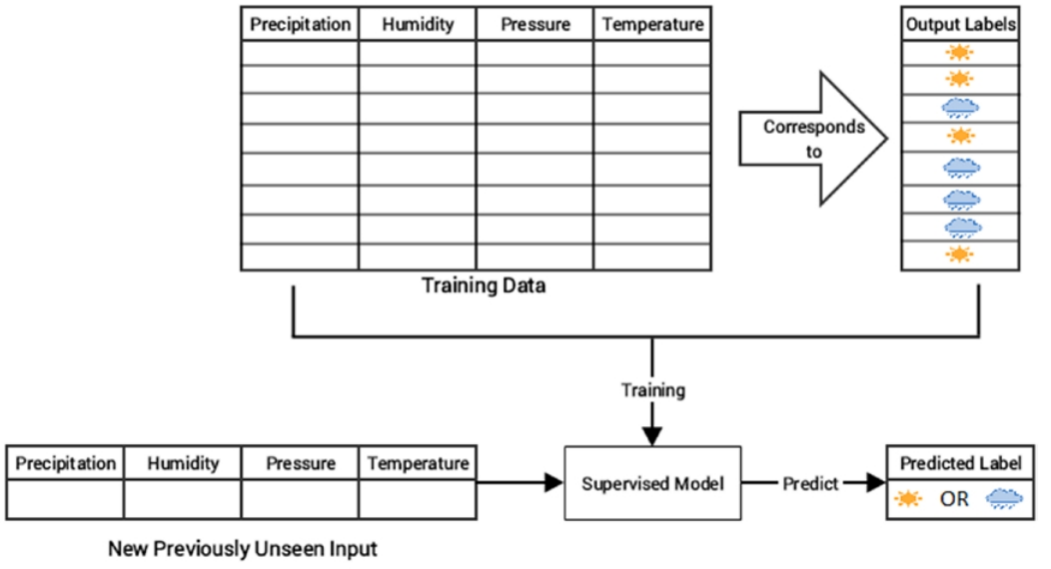
- Classification
- Regression



UCERD

Classification

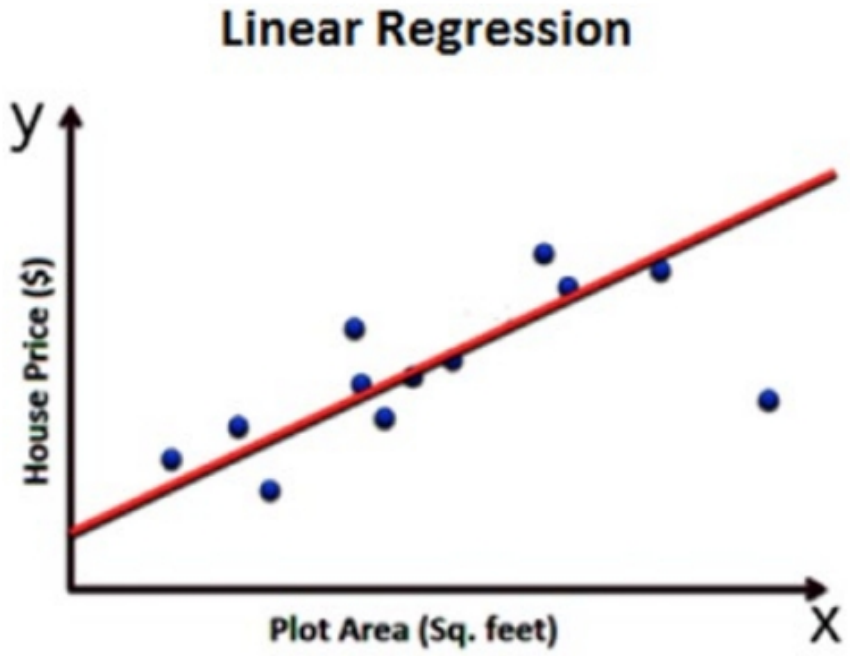
- The classification based tasks are a sub-field under supervised Machine Learning, where the key objective is to predict output labels or responses that are categorical in nature for input data based on what the model has learned in the training phase. Output labels here are also known as classes or class labels are these are categorical in nature meaning they are unordered and discrete values. Thus, each output response belongs to a specific discrete class or category.

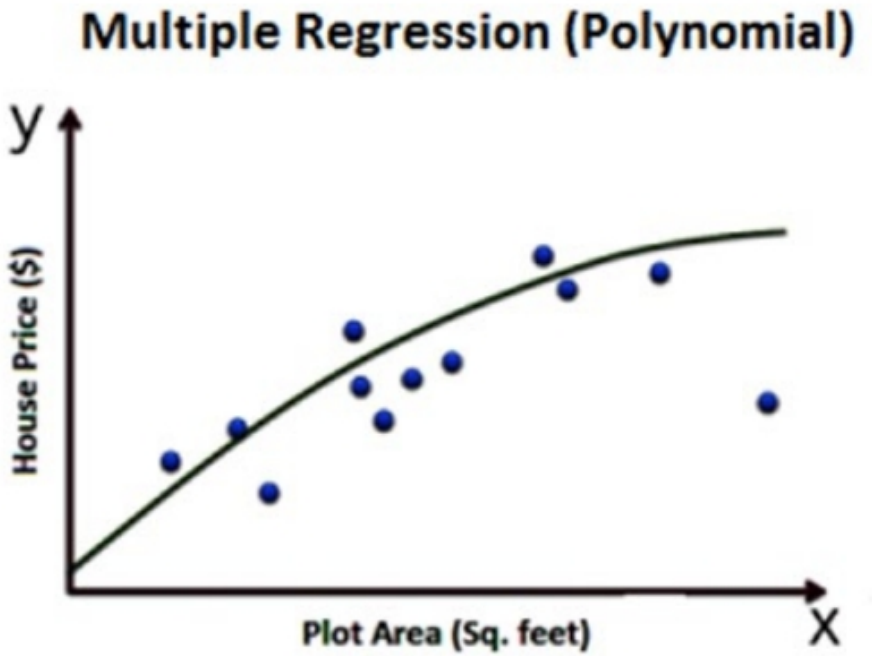


UCERD

ML: Regression

- Machine Learning tasks where the main objective is value estimation can be termed as regression tasks. Regression based methods are trained on input data samples having output responses that are continuous numeric values unlike classification, where we have discrete categories or classes.





Unsupervised Learning

- Unsupervised learning methods usually require some training data where the outcomes which we are trying to predict are already available in the form of discrete labels or continuous values. However, often we do not have the liberty or advantage of having pre-labeled training data and we still want to extract useful insights or patterns from our data. In this scenario, unsupervised learning methods are extremely powerful.
- These methods are called unsupervised because the model or algorithm tries to learn inherent latent structures, patterns and relationships from given data without any help or supervision like providing annotations in the form of labeled outputs or outcomes.

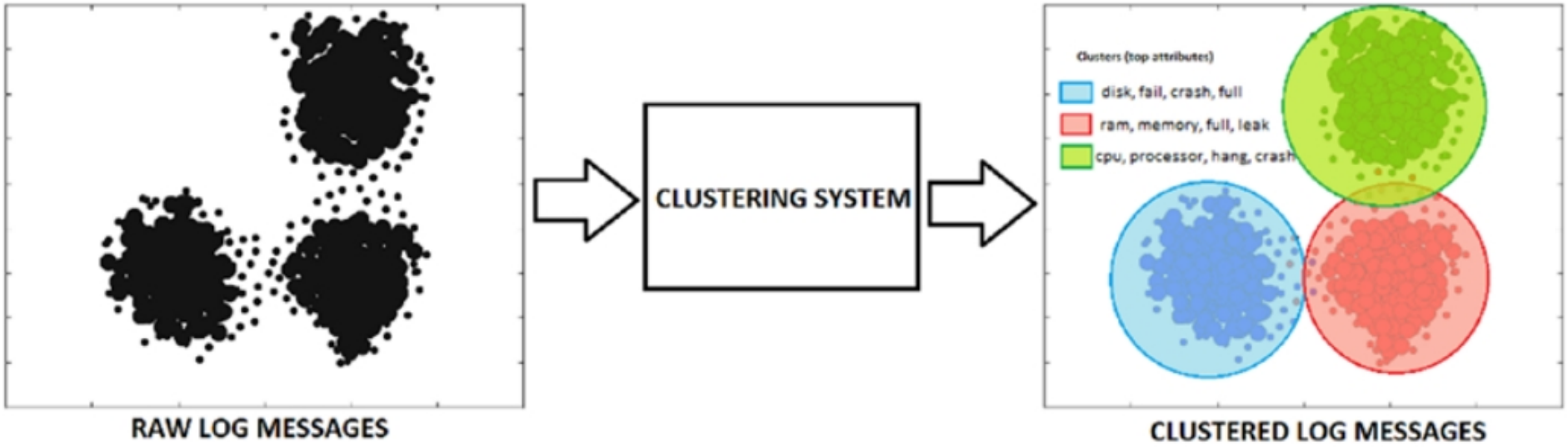
UCERD

- Clustering
- Dimensionality reduction
- Anomaly detection
- Association rule-mining

UCERD

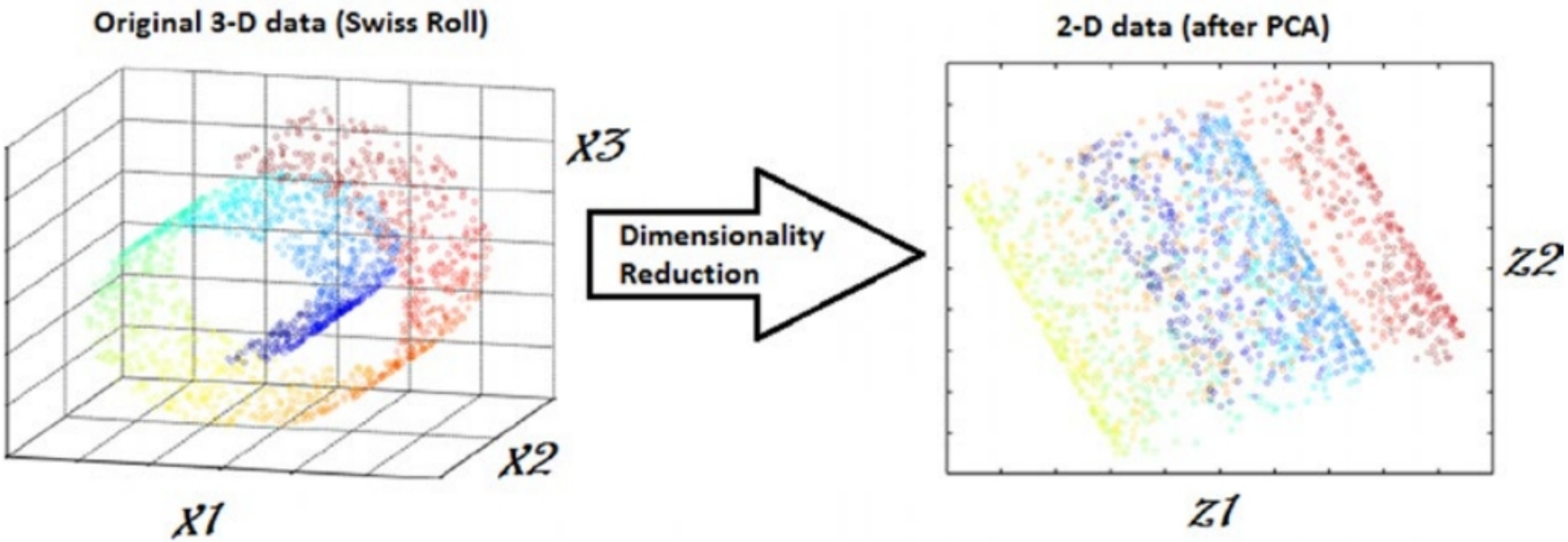
Clustering

- Clustering methods are Machine Learning methods that try to find patterns of similarity and relationships among data samples in our dataset and then cluster these samples into various groups, such that each group or cluster of data samples has some similarity, based on the inherent attributes or features. These methods are completely unsupervised because they try to cluster data by looking at the data features without any prior training, supervision, or knowledge about data attributes, associations, and relationships.



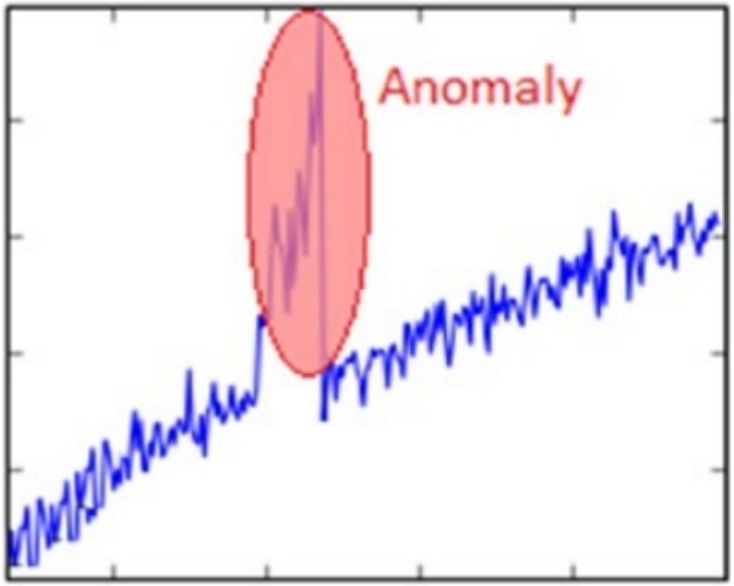
•Dimensionality Reduction

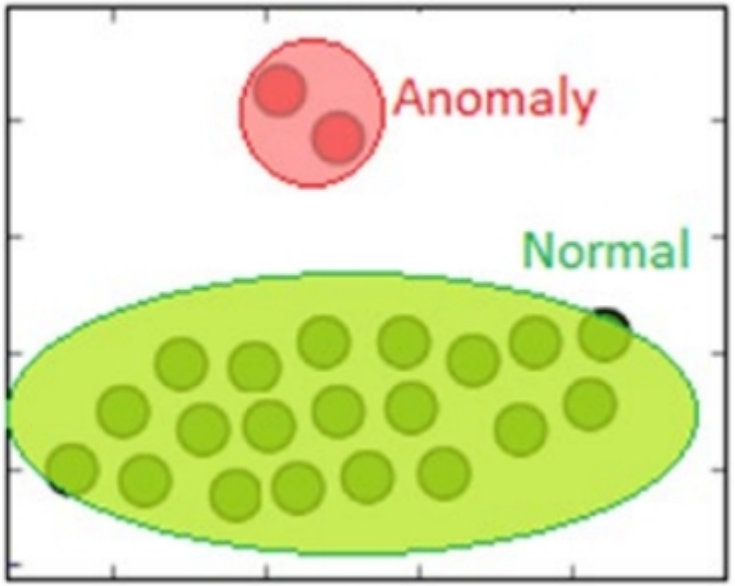
- Dimension reduction is the process of reducing the number of random variables under consideration by obtaining a set of principal variables.
- Once we start extracting attributes or features from raw data samples, sometimes our feature space gets bloated up with a humongous number of features. This poses multiple challenges including analyzing and visualizing data with thousands or millions of features, which makes the feature space extremely complex posing problems with regard to training models, memory, and space constraints. In fact this is referred to as the “curse of dimensionality”. Unsupervised methods can also be used in these scenarios, where we reduce the number of features or attributes for each data sample.
- These methods reduce the number of feature variables by extracting or selecting a set of principal or representative features. There are multiple popular algorithms available for dimensionality reduction like Principal Component Analysis (PCA), nearest neighbors, and discriminant analysis. Figure shows the output of a typical feature reduction process applied to a Swiss Roll 3D structure having three dimensions to obtain a two-dimensional feature space for each data sample using PCA.



Anomaly Detection

The process of anomaly detection is also termed as outlier detection, where we are interested in finding out occurrences of rare events or observations that typically do not occur normally based on historical data samples. Sometimes anomalies occur infrequently and are thus rare events, and in other instances, anomalies might not be rare but might occur in very short bursts over time, thus have specific patterns.

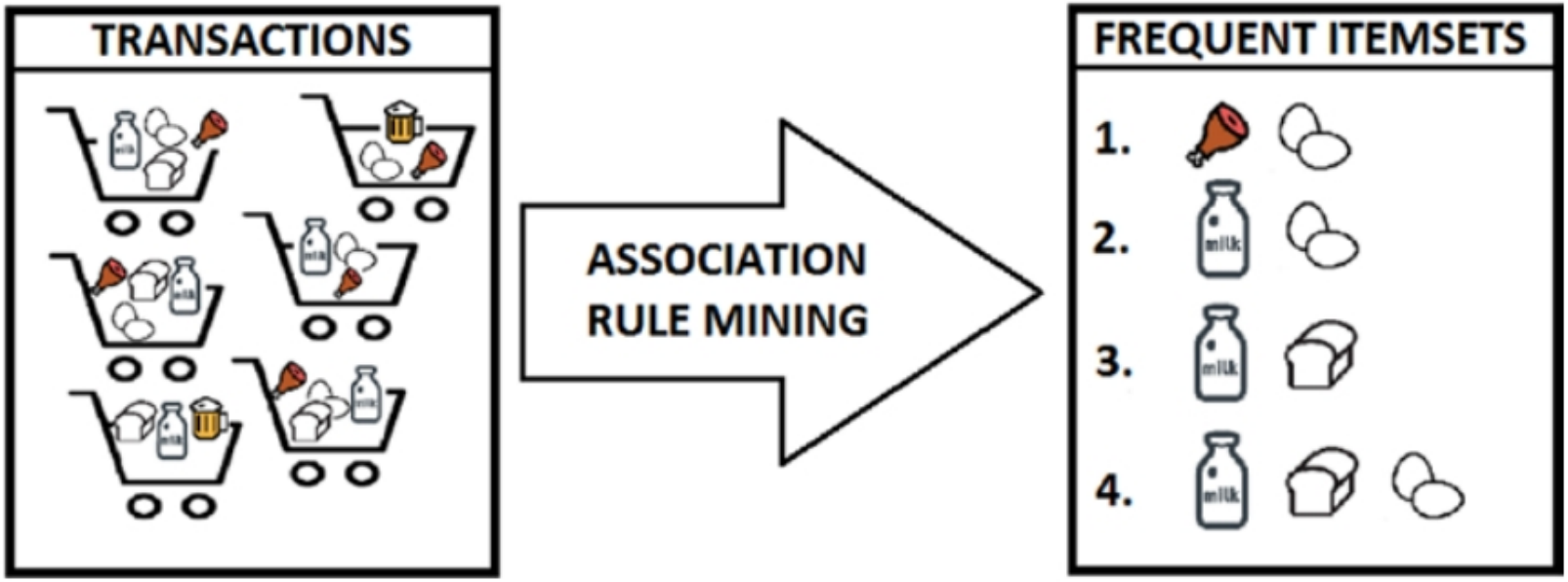




UCERD

Association Rule-Mining

Typically association rule-mining is a data mining method use to examine and analyze large transactional datasets to find patterns and rules of interest. These patterns represent interesting relationships and associations, among various items across transactions. Association rule-mining is also often termed as market basket analysis, which is used to analyze customer shopping patterns.



UCERD

Reinforcement Learning

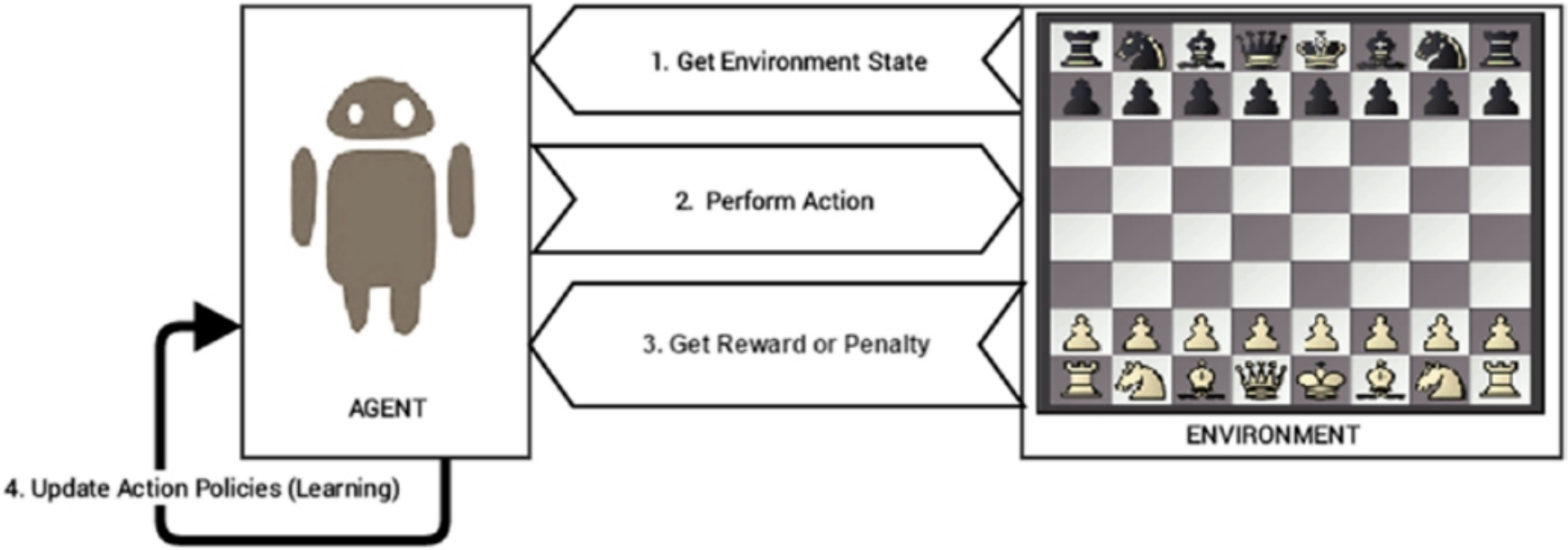
The reinforcement learning methods are a bit different from conventional supervised or unsupervised methods. In this context, we have an agent that we want to train over a period of time to interact with a specific environment and improve its performance over a period of time with regard to the type of actions it performs on the environment. Typically the agent starts with a set of strategies or policies for interacting with the environment. On observing the environment, it takes a particular action based on a rule or policy and by observing the current state of the environment. Based on the action, the agent gets a reward, which could be beneficial or detrimental in the form of a penalty. It updates its current policies and strategies if needed and this iterative process continues till it learns enough about its environment to get the desired rewards.

-

UCERD

- 1. Prepare agent with set of initial policies and strategy
- 2. Observe environment and current state
- 3. Select optimal policy and perform action
- 4. Get corresponding reward (or penalty)
- 5. Update policies if needed
- 6. Repeat Steps 2 - 5 iteratively until agent learns the most optimal policies
-

UCERD


1. Get Environment State
2. Perform Action
3. Get Reward or Penalty
AGENT
ENVIRONMENT
4. Update Action Policies (Learning)

UCERD

Contents

- Machine Learning Understanding
- ML Basic Model
- **Machine Learning Methods**
- Data Understanding

UCERD

- Linear Regression
- Logistic Regression
- Decision Tree
- Random Forest
- SVM
- Naive Bayes
- kNN
- Gradient Boosting algorithms
    - GBM
    - XGBoost
    - LightGBM
    - CatBoost
- **K-Means**
- **Dimensionality Reduction Algorithms**

UCERD

Linear Regression

- It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a *X + b.

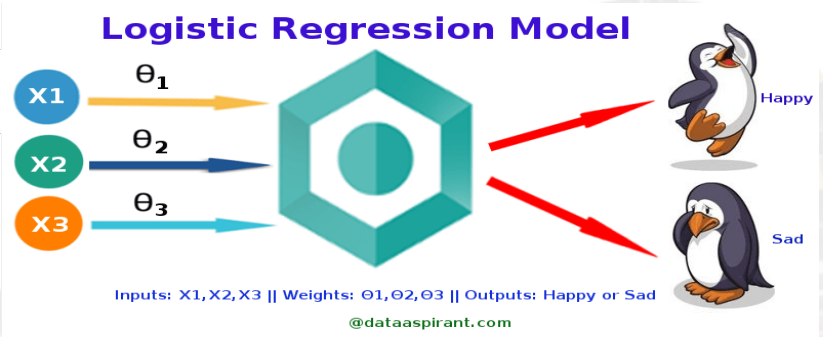
UCERD

```
#Import Library
#Import other necessary libraries like pandas, numpy...
from sklearn import linear_model
#Load Train and Test datasets
#Identify feature and response variable(s) and values must be numeric and numpy arrays
x_train=input_variables_values_training_datasets
y_train=target_variables_values_training_datasets
x_test=input_variables_values_test_datasets
# Create linear regression object
linear = linear_model.LinearRegression()
# Train the model using the training sets and check score
linear.fit(x_train, y_train)
linear.score(x_train, y_train)
#Equation coefficient and Intercept
print('Coefficient: \n', linear.coef_)
print('Intercept: \n', linear.intercept_)
#Predict Output
predicted= linear.predict(x_test)
```

UCERD

Logistic Regression

- It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variable(s).
- In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as logit regression. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).



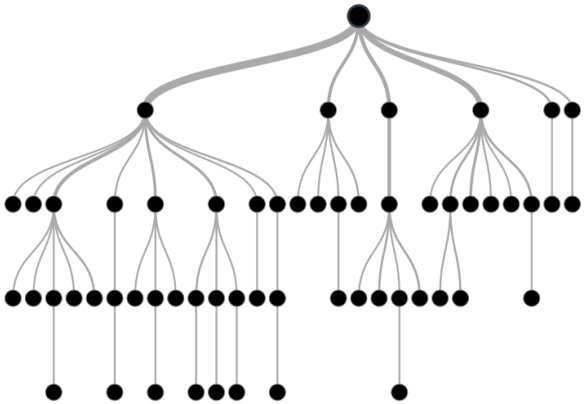
UCERD

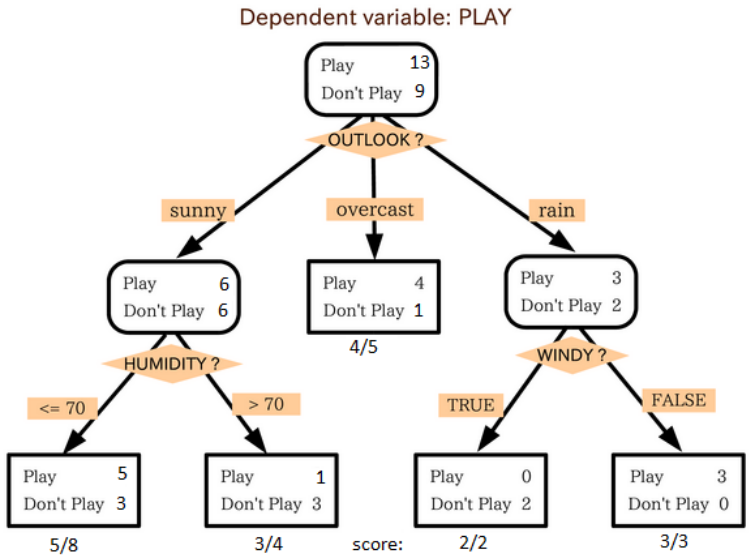
```
#Import Library
from sklearn.linear_model import LogisticRegression
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create logistic regression object
model = LogisticRegression()
# Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Equation coefficient and Intercept
print('Coefficient: \n', model.coef_)
print('Intercept: \n', model.intercept_)
#Predict Output
predicted= model.predict(x_test)
```

UCERD

Decision Tree

- It is a type of supervised learning algorithm that is mostly used for classification problems.
- Surprisingly, it works for both categorical and continuous dependent variables.
- Decision Tree split the population into two or more homogeneous sets. This is done based on most significant attributes/ independent variables to make as distinct groups as possible.





UCERD

```
# Import Library
#I mport other necessary libraries like pandas, numpy...
from sklearn import tree
#Assumed you have, X (predictor) and Y (target) for training data set and
x_test(predictor) of test_dataset
# Create tree object
model = tree.DecisionTreeClassifier(criterion='gini') # for
classification, here you can change the algorithm as gini or entropy
(information gain) by default it is gini
# model = tree.DecisionTreeRegressor() for regression
# Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

UCERD

Random Forest

- Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we've collection of decision trees (so known as "Forest"). To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).
- Each tree is planted & grown as follows:
  - If the number of cases in the training set is N, then sample of N cases is taken at random but with replacement. This sample will be the training set for growing the tree.
  - If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
  - Each tree is grown to the largest extent possible. There is no pruning.

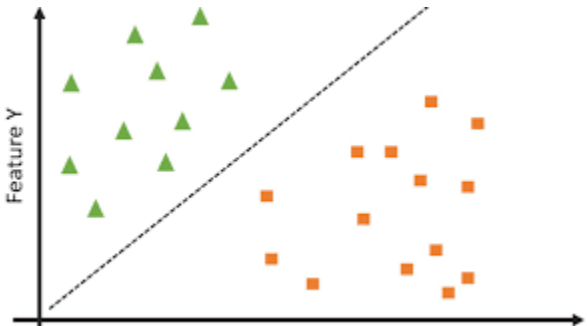
UCERD

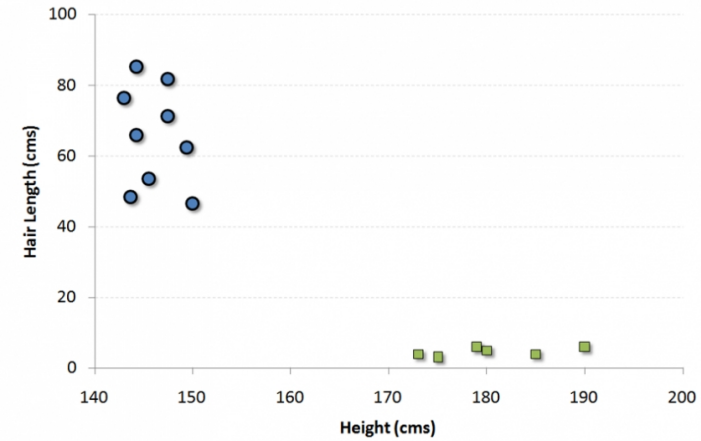
- #Import Library
- from sklearn.ensemble import RandomForestClassifier
- #Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
- # Create Random Forest object
- model= RandomForestClassifier()
- # Train the model using the training sets and check score
- model.fit(X, y)
- #Predict Output
- predicted= model.predict(x_test)

UCERD

SVM (Support Vector Machine)

It is a classification method. The algorithm is used to plot each data item as a point in n-dimensional space (where n is number of features it has) with the value of each feature being the value of a particular coordinate.



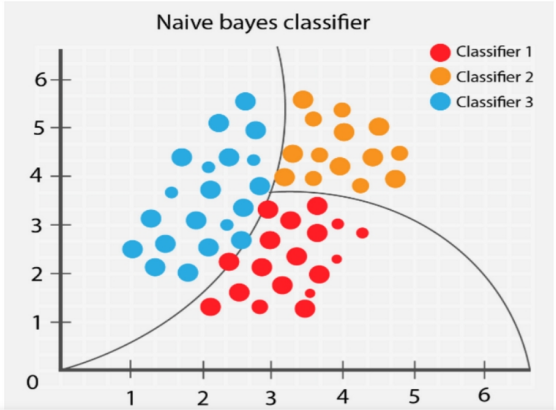


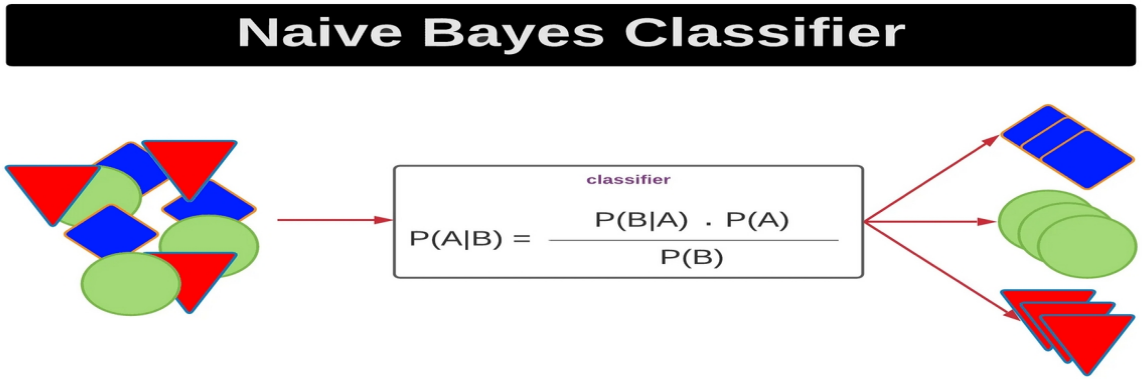
- #Import Library
- from sklearn import svm
- #Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
- # Create SVM classification object
- **model = svm.svc() # there is various option associated with it, this is simple for classification. You can refer link, for mo# re detail.**
- # Train the model using the training sets and check score
- model.fit(X, y)
- model.score(X, y)
- #Predict Output
- predicted= model.predict(x_test)

UCERD

Naive Bayes

It is a classification technique based on Bayes' theorem with an assumption of independence between predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier would consider all of these properties to independently contribute to the probability that this fruit is an apple.





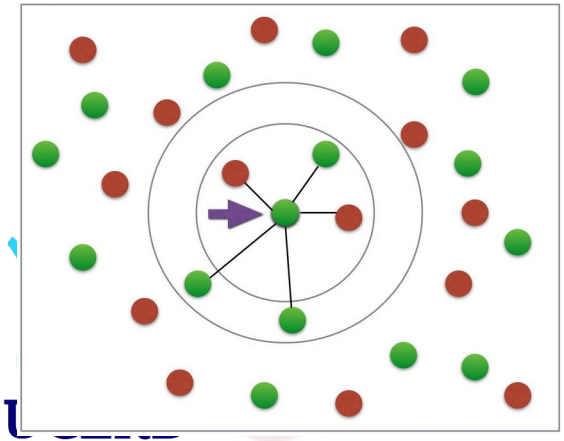
```
#Import Library

from sklearn.naive_bayes import GaussianNB

#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset

# Create SVM classification object

model = GaussianNB() # there is other distribution for multinomial classes like Bernoulli Naive Bayes, Refer link

# Train the model using the training sets and check score

model.fit(X, y)

#Predict Output

predicted= model.predict(x_test)
```

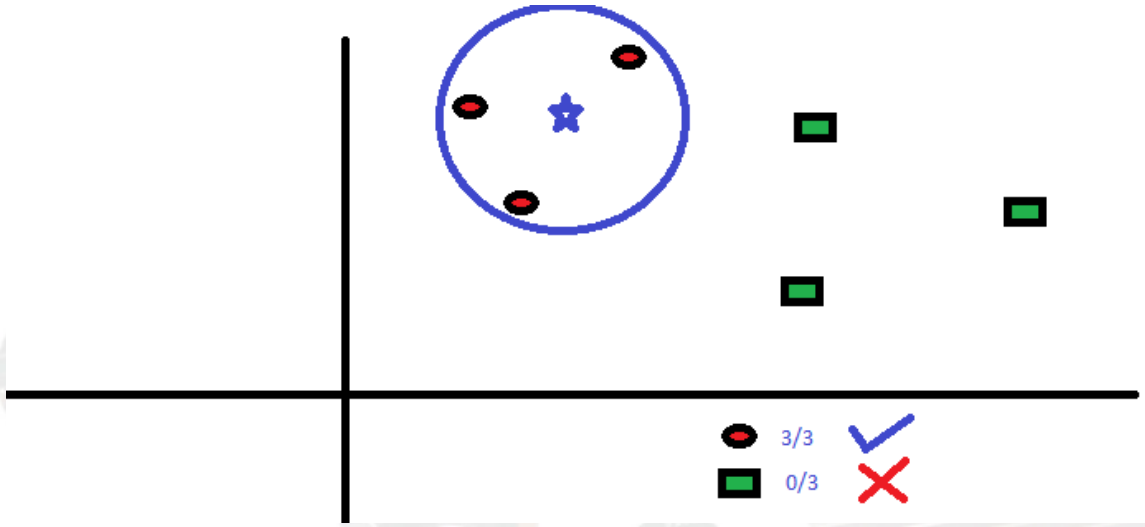
UCERD

kNN (k- Nearest Neighbors)

It can be used for both classification and regression problems.

- However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function.
  - KNN is computationally expensive
  - Variables should be normalized else higher range variables can bias it
  - Works on pre-processing stage more before going for kNN like outlier, noise removal





```
#Import Library
from sklearn.neighbors import KNeighborsClassifier
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create KNeighbors classifier object model
KNeighborsClassifier(n_neighbors=6) # default value for n_neighbors is 5
# Train the model using the training sets and check score
model.fit(X, y)
#Predict Output
predicted= model.predict(x_test)
```

UCERD

Gradient Boosting Algorithms

- GBM is a boosting algorithm used when we deal with plenty of data to make a prediction with high prediction power. Boosting is actually an ensemble of learning algorithms which combines the prediction of several base estimators in order to improve robustness over a single estimator. It combines multiple weak or average predictors to a build strong predictor. These boosting algorithms always work well in data science competitions like Kaggle, AV Hackathon, CrowdAnalytix.

UCERD

```
#Import Library
from sklearn.ensemble import GradientBoostingClassifier
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create Gradient Boosting Classifier object
model= GradientBoostingClassifier(n_estimators=100, learning_rate=1.0, max_depth=1, random_state=0)
# Train the model using the training sets and check score
model.fit(X, y)
#Predict Output
predicted= model.predict(x_test)
```

UCERD

K-Means

- It is a type of unsupervised algorithm which solves the clustering problem. Its procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). Data points inside a cluster are homogeneous and heterogeneous to peer groups.
- How K-means forms cluster:
  - K-means picks k number of points for each cluster known as centroids.
  - Each data point forms a cluster with the closest centroids i.e. k clusters.
  - Finds the centroid of each cluster based on existing cluster members. Here we have new centroids.
  - As we have new centroids, repeat step 2 and 3. Find the closest distance for each data point from new centroids and get associated with new k-clusters. Repeat this process until convergence occurs i.e. centroids does not change.

- How to determine value of K:

  In K-means, we have clusters and each cluster has its own centroid. Sum of square of difference between centroid and the data points within a cluster constitutes within sum of square value for that cluster. Also, when the sum of square values for all the clusters are added, it becomes total within sum of square value for the cluster solution.

  We know that as the number of cluster increases, this value keeps on decreasing but if you plot the result you may see that the sum of squared distance decreases sharply up to some value of k, and then much more slowly after that. Here, we can find the optimum number of cluster.

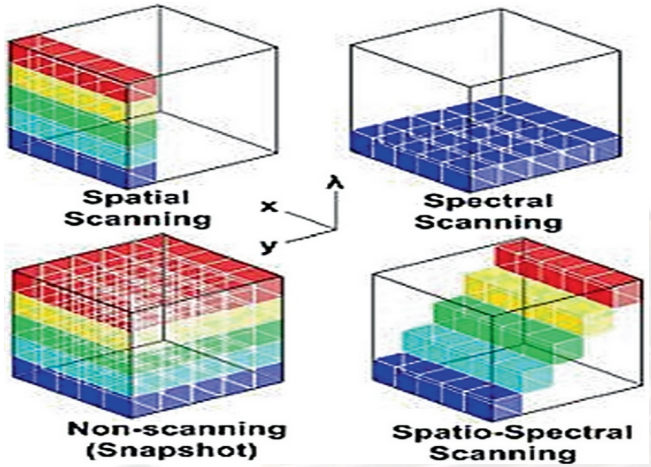
UCERD

```
#Import Library
from sklearn.cluster import KMeans
#Assumed you have, X (attributes) for training data set and
x_test(attributes) of test_dataset
# Create KNeighbors classifier object model
model = KMeans(n_clusters=3, random_state=0)
# Train the model using the training sets and check score
model.fit(X)
#Predict Output
predicted= model.predict(x_test)
```

UCERD

Dimensionality Reduction Algorithms

- Dimension Reduction refers to the process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely.
- These techniques are typically used while solving machine learning problems to obtain better features for a classification or regression task.



UCERD

Importance

- It helps in data compressing and reducing the storage space required
- It fastens the time required for performing same computations. Less dimensions leads to less computing, also less dimensions can allow usage of algorithms unfit for a large number of dimensions
- It takes care of multi-collinearity that improves the model performance. It removes redundant features. For example: there is no point in storing a value in two different units (meters and inches).
- Reducing the dimensions of data to 2D or 3D may allow us to plot and visualize it precisely. You can then observe patterns more clearly. Below you can see that, how a 3D data is converted into 2D. First it has identified the 2D plane then represented the points on these two new axis z1 and z2.

UCERD

- #Import Library
- from sklearn import decomposition
- #Assumed you have training and test data set as train and test
- # Create PCA obeject pca= decomposition.PCA(n_components=k) #default value of k =min(n_sample, n_features)
- # For Factor analysis
- #fa= decomposition.FactorAnalysis()
- # Reduced the dimension of training dataset using PCA
- train_reduced = pca.fit_transform(train)
- #Reduced the dimension of test dataset
- test_reduced = pca.transform(test)

UCERD

Contents

- Machine Learning Understanding
- ML Basic Model
- Machine Learning Methods
- **Data Understanding**

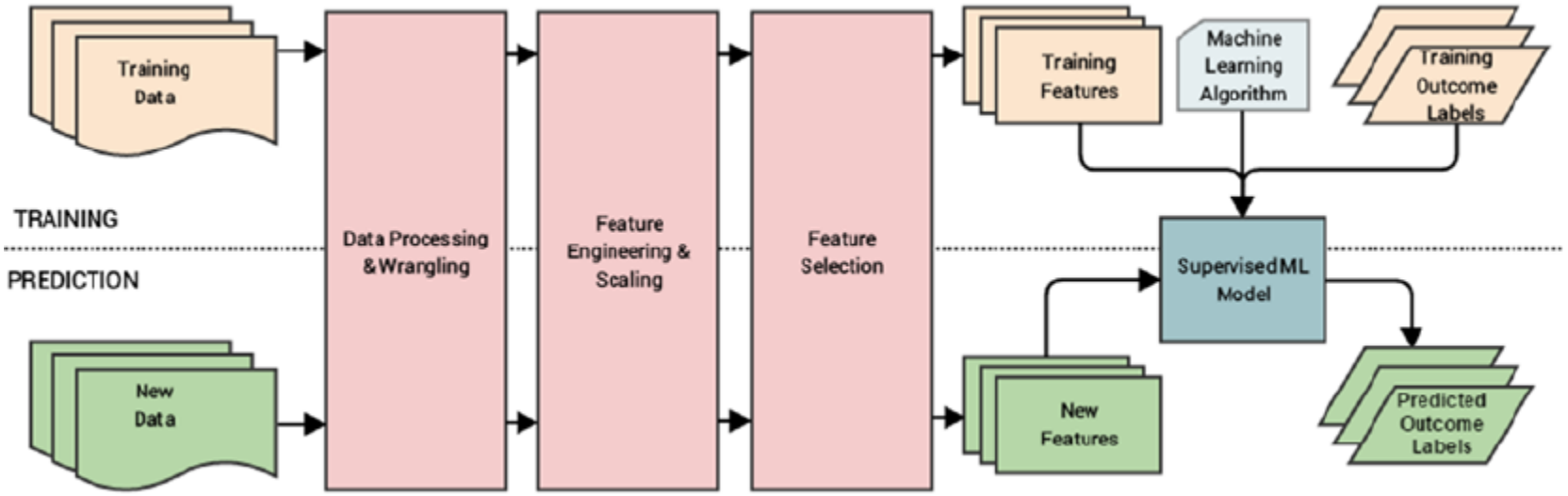
UCERD

Data

- Data Collection
- Data Description
- Exploratory Data Analysis
- Data Quality Analysis
- Data Preparation
- Data Integration
- Data Wrangling

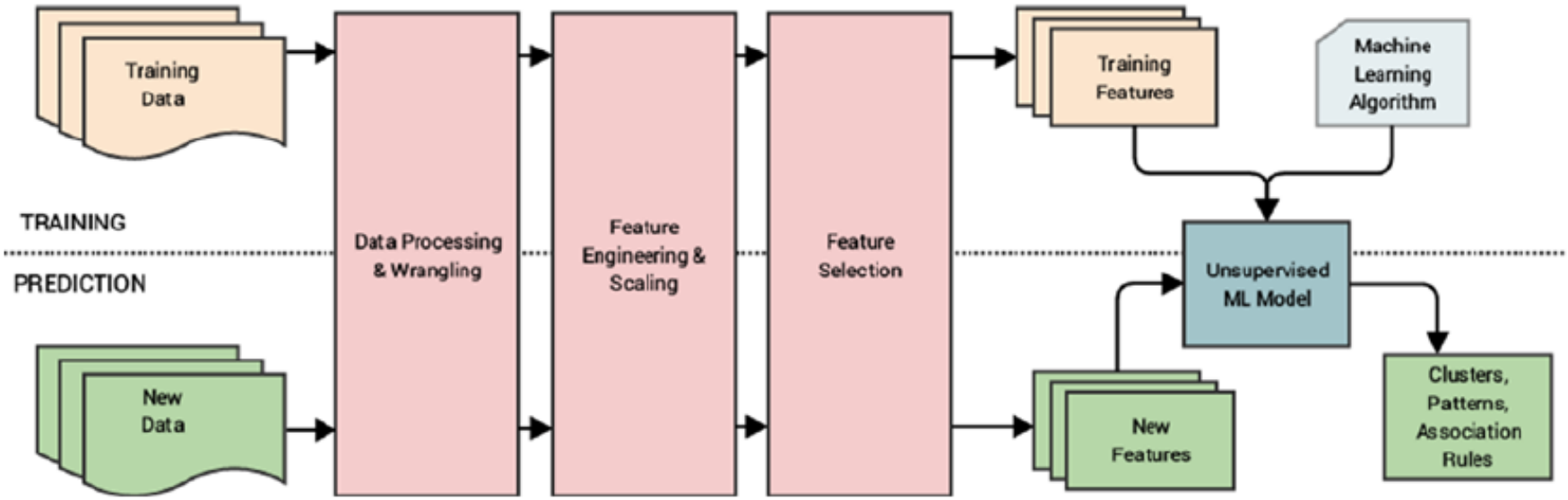
UCERD

Supervised Machine Learning Pipeline



UCERD

Unsupervised Machine Learning Pipeline



UCERD

Datasets: Datatypes

- Scalar
- Vector
- Matrix
- Tensor
  - You can think of a tensor as a generic array. Tensors are basically arrays with a variable number of axes. An element in a three-dimensional tensor T can be denoted by Tx,y,z where x, y, z denote the three axes for specifying element T.

UCERD

Databases Format

- A CSV data file is one of the most widely available formats of data.
- Java Script Object Notation (JSON) is one of the most widely used data interchange formats across the digital realm. JSON is a lightweight alternative to legacy formats like XML.
- XML
- SQL => sqlalchemy and pyodbc

UCERD

Data Description

- Numeric
- Text
- Categorical

UCERD

Data Wrangling / Munging

- Understand Data
- Filtering Data
  - Clean
  - Typecasting,Duplicate Handling
- Transforming
- Normalize
- Visualization

	Date	Price	Product ID	Quantity Purchased	Serial No	User ID	User Type
0	NaN	3021.06	417	13	1000	5958	NaN
1	NaN	1822.62	731	1	1001	5351	c
2	2016-07-01	542.36	829	2	1002	5799	a
3	2016-01-20	2323.30	905	0	1003	5480	d
4	2016-01-19	243.43	158	37	1004	5790	a
5	2016-01-16	274.26	754	33	1005	5820	a
6	NaN	5836.68	341	18	1006	5468	c
7	2016-01-19	NaN	819	34	1007	5486	b
8	2016-01-23	1171.88	929	12	1008	5143	a
9	2016-07-01	668.80	718	31	1009	5510	d

UCERD

Statistics

- The field of statistics can be defined as a specialized branch of mathematics that consists of frameworks and methodologies to collect, organize, analyze, interpret, and present data. Generally this falls more undeapplied mathematics and borrows concepts from linear algebra, distributions, probability theory, and inferential methodologies. There are two major areas under statistics that are mentioned as follows.
- Descriptive statistics
- Inferential statistics

UCERD

Challenges in Machine Learning

- Data quality issues lead to problems, especially with regard to data processing and feature extraction.
- Data acquisition, extraction, and retrieval is an extremely tedious and time consuming process.
- Lack of good quality and sufficient training data in many scenarios.
- Formulating business problems clearly with well-defined goals and objectives.
- Feature extraction and engineering, especially hand-crafting features, is one of the most difficult yet important tasks in Machine Learning. Deep Learning seems to have gained some advantage in this area recently.
- Overfitting or underfitting models can lead to the model learning poor representations and relationships from the training data leading to detrimental performance.
- The curse of dimensionality: too many features can be a real hindrance. Complex models can be difficult to deploy in the real world.

UCERD

Real-World Applications of Machine Learning

- Product recommendations in online shopping platforms
- Sentiment and emotion analysis
- Anomaly detection
- Fraud detection and prevention
- Content recommendation (news, music, movies, and so on)
- Weather forecasting
- Stock market forecasting
- Market basket analysis
- Customer segmentation
- Object and scene recognition in images and video
- Speech recognition
- Churn analytics
- Click through predictions
- Failure/defect detection and prevention
- E-mail spam filtering

UCERD