

# Hands-on Introduction to Machine Learning with IBM Cloud Pak for Data

March 25, 2021

The session starts at 9:00am EST

# Hands on Introduction to Machine Learning with IBM Cloud Pak for Data




Power of data. Simplicity of design. Speed of innovation.

**Bernie Beekman**  
**Michael Cronk**  
**Frank Greco**  
**James Parry**

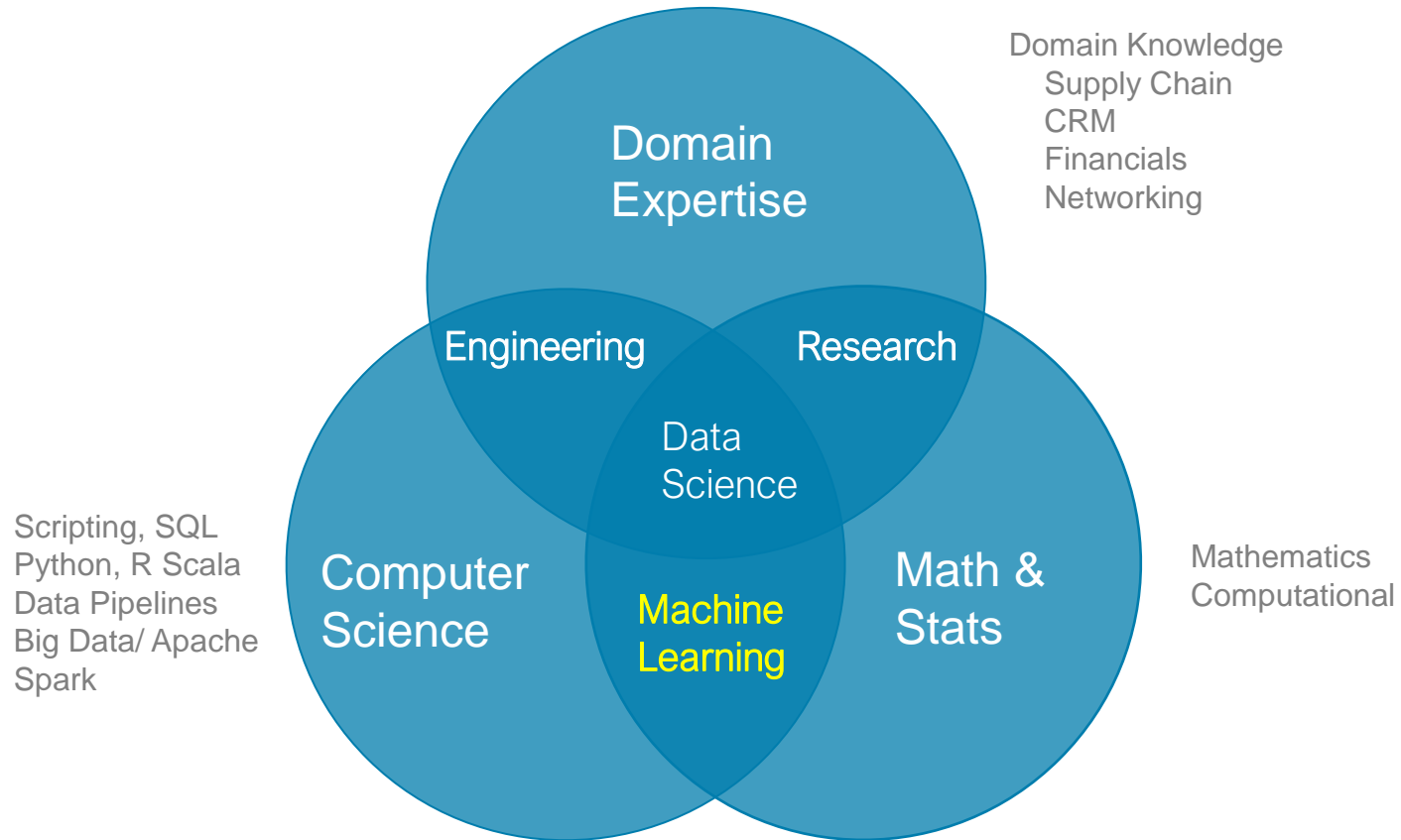
# Agenda

Time	Description
09:00 AM – 09:05 AM	Welcome, Agenda
09:05 AM – 10:00 AM	Introduction to Machine Learning (Overview, Data) Lab 1-2 Overview
10:00 AM – 10:30 AM	Lab-1 Set up Environment
10:30 AM – 11:15 AM	Lab-2 Data Refinery
11:15 AM – 12:00 PM	Introduction to Machine Learning (Modeling/Evaluation)
12:00 PM – 12:15 PM	Lab 3,4,5, Overview
12:15 PM – 12:45 PM	Lunch Break
12:45 PM – 01:30 PM	Lab 3 – SPSS Modeler
01:30 PM – 02:00 PM	Lab 4 – Auto AI
02:00 PM – 02:30 PM	Lab 5 – Heart Disease Notebook
02:30 PM – 02:45 PM	Introduction to Trusted AI Lab 6 Overview
02:45 PM – 03:30 PM	Lab-6 – Watson OpenScale
03:30 PM – 04:00 PM	Introduction to Neural Networks, Adversarial Robustness Toolkit Lab 7-8 Overview
04:00 PM – 04:30 PM	Lab 7- Neural Network Lab
04:30 PM – 05:00 PM	Lab 8 – Adversarial Robustness Toolkit

# Introduction to Machine Learning

- Overview 
- Data Science Methodology
- Data Understanding/Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Evaluation
- Trusted AI
- Deep Learning

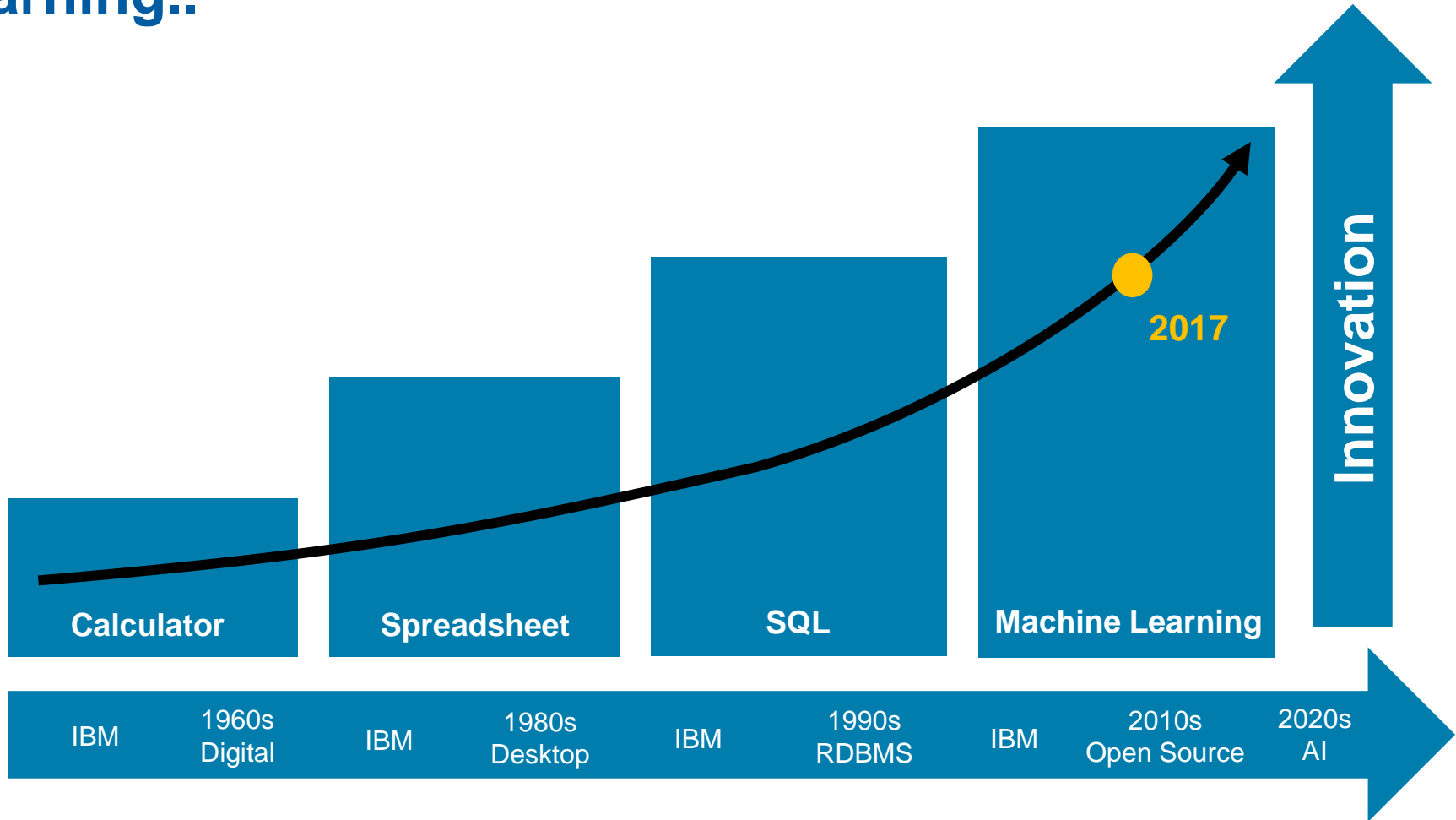
# Machine Learning and Data Science....



*Data Science Projects Require Multiple Skills*

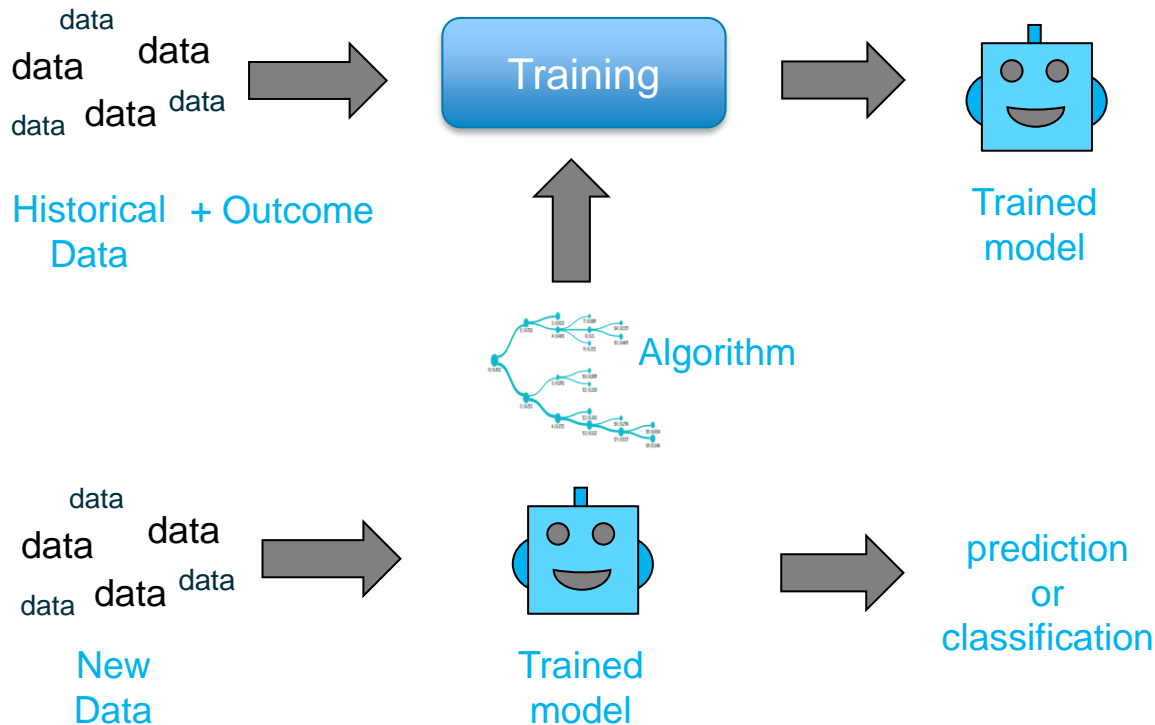
Modified from Drew Conway's Venn Diagram

# Future of Data Science is Democratizing Machine Learning..



# But what is Machine Learning?

*“Computers that learn without being **explicitly programmed**”*



1

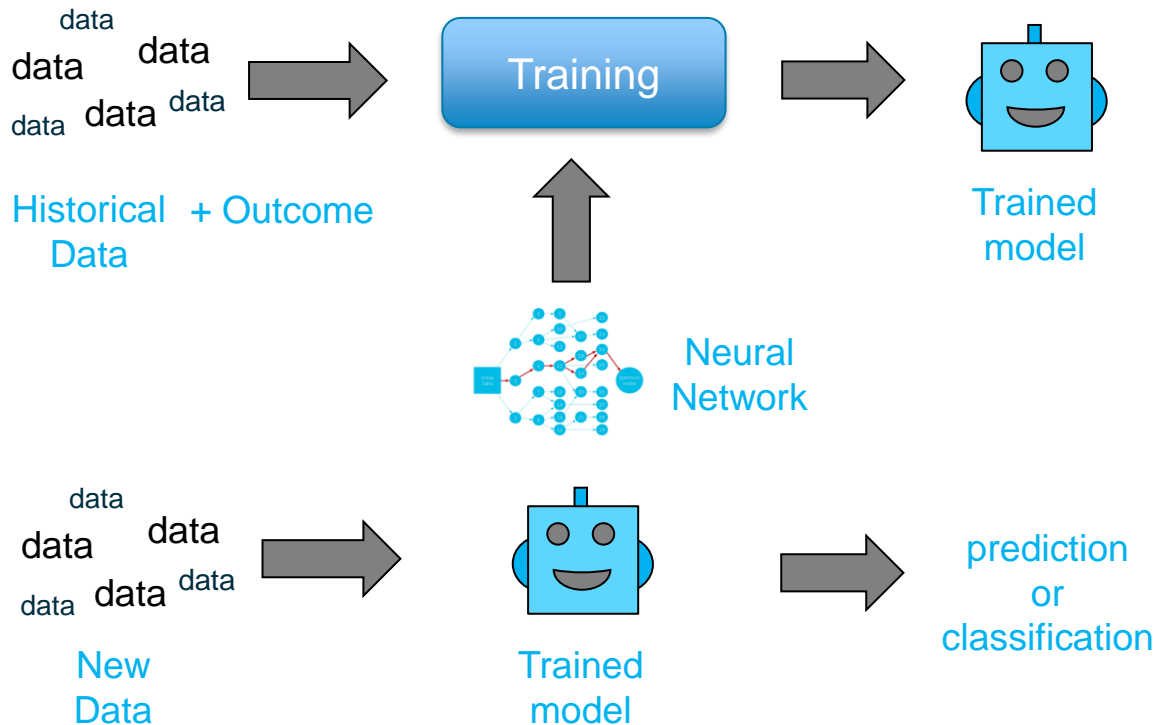
A machine learning model is trained to recognize patterns in historical data

2

The model is then shown new data and asked to predict or classify it. If the patterns in the new data match the training data then the model makes accurate predictions.

# But what is Deep Learning?

*“Computers that learn without being **explicitly programmed**”*



1

A machine learning model is trained to recognize patterns in historical data

2

The model is then shown new data and asked to predict or classify it. If the patterns in the new data match the training data then the model makes accurate predictions.

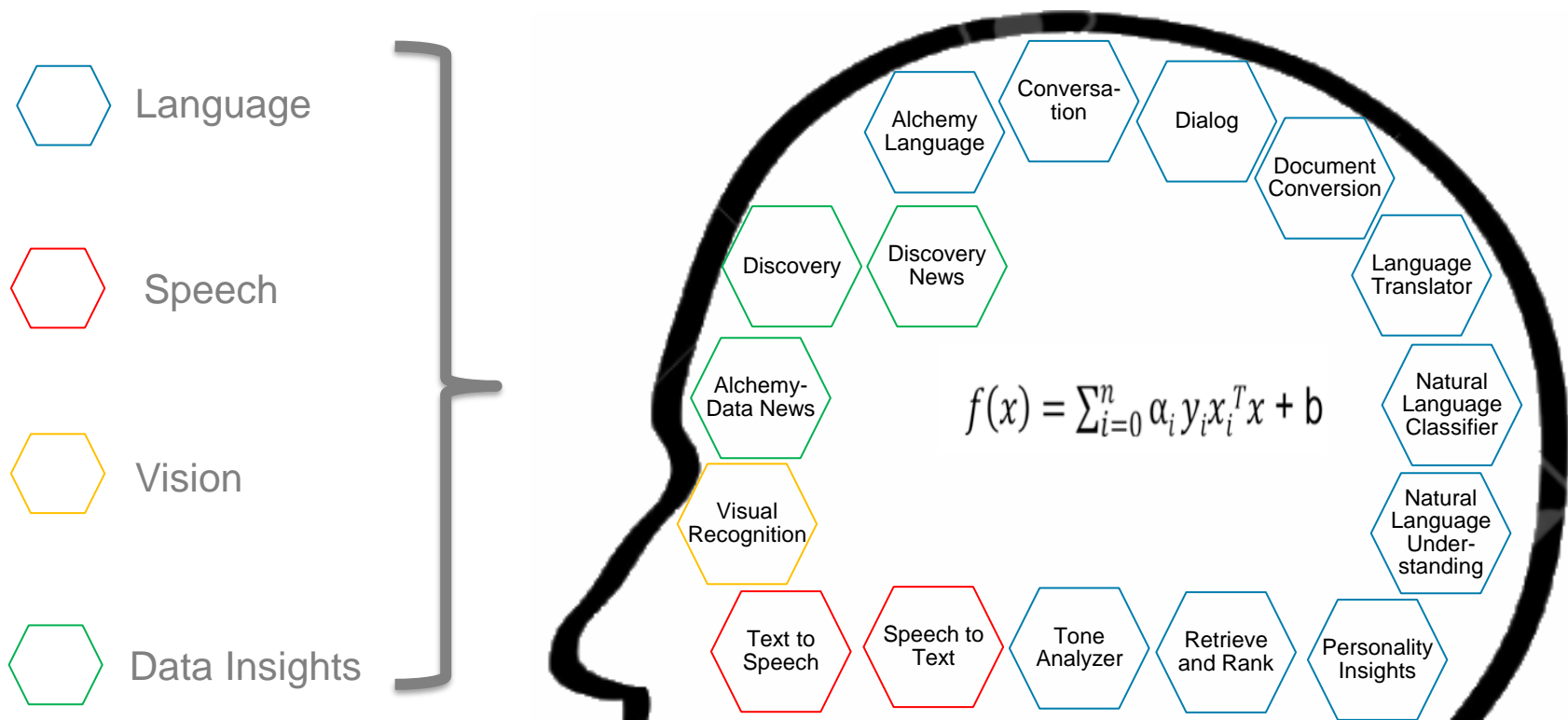


# But what is Artificial Intelligence?

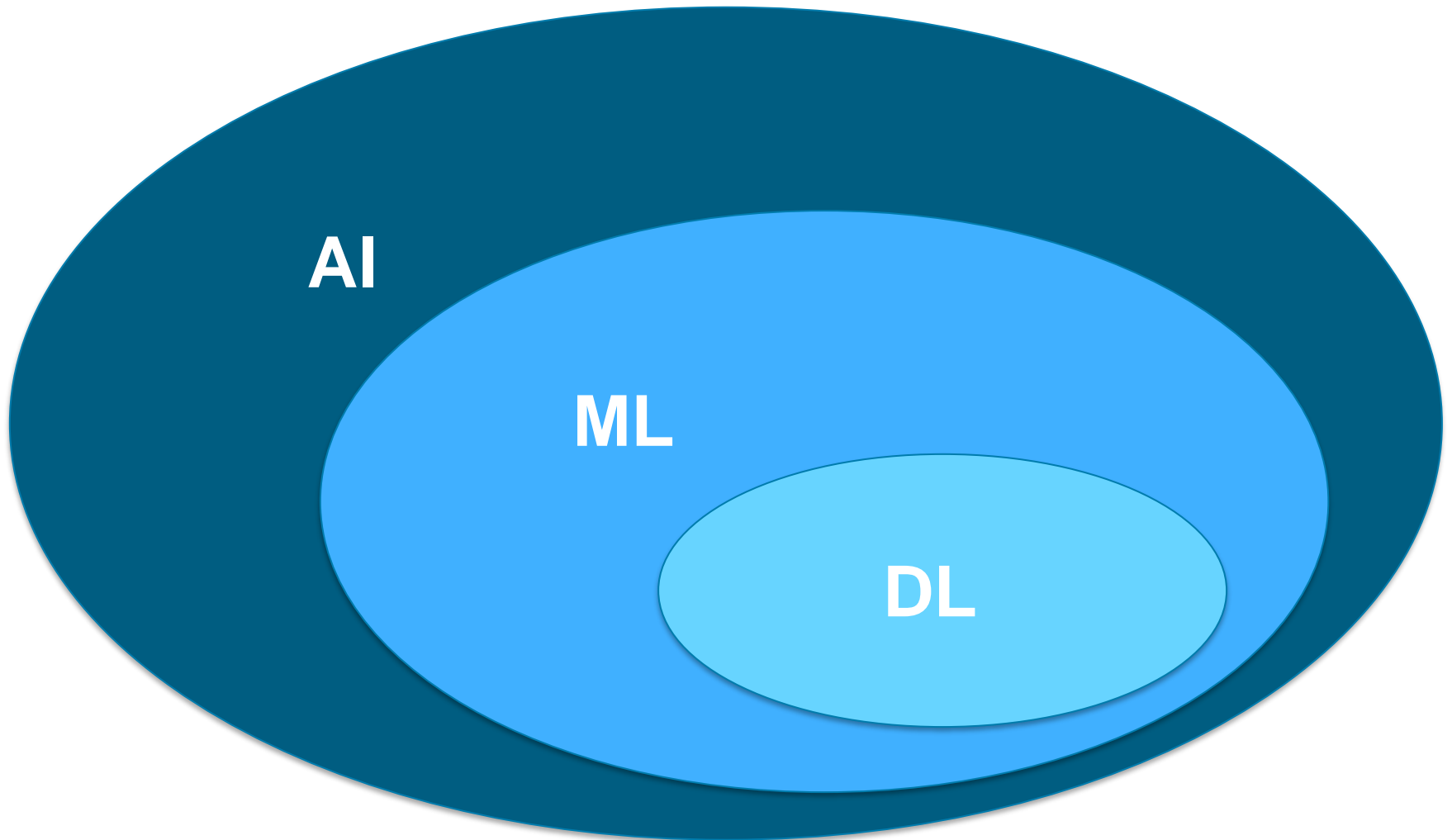
A theory and development of computer systems able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages..

# Machine Learning = Artificial Intelligence???

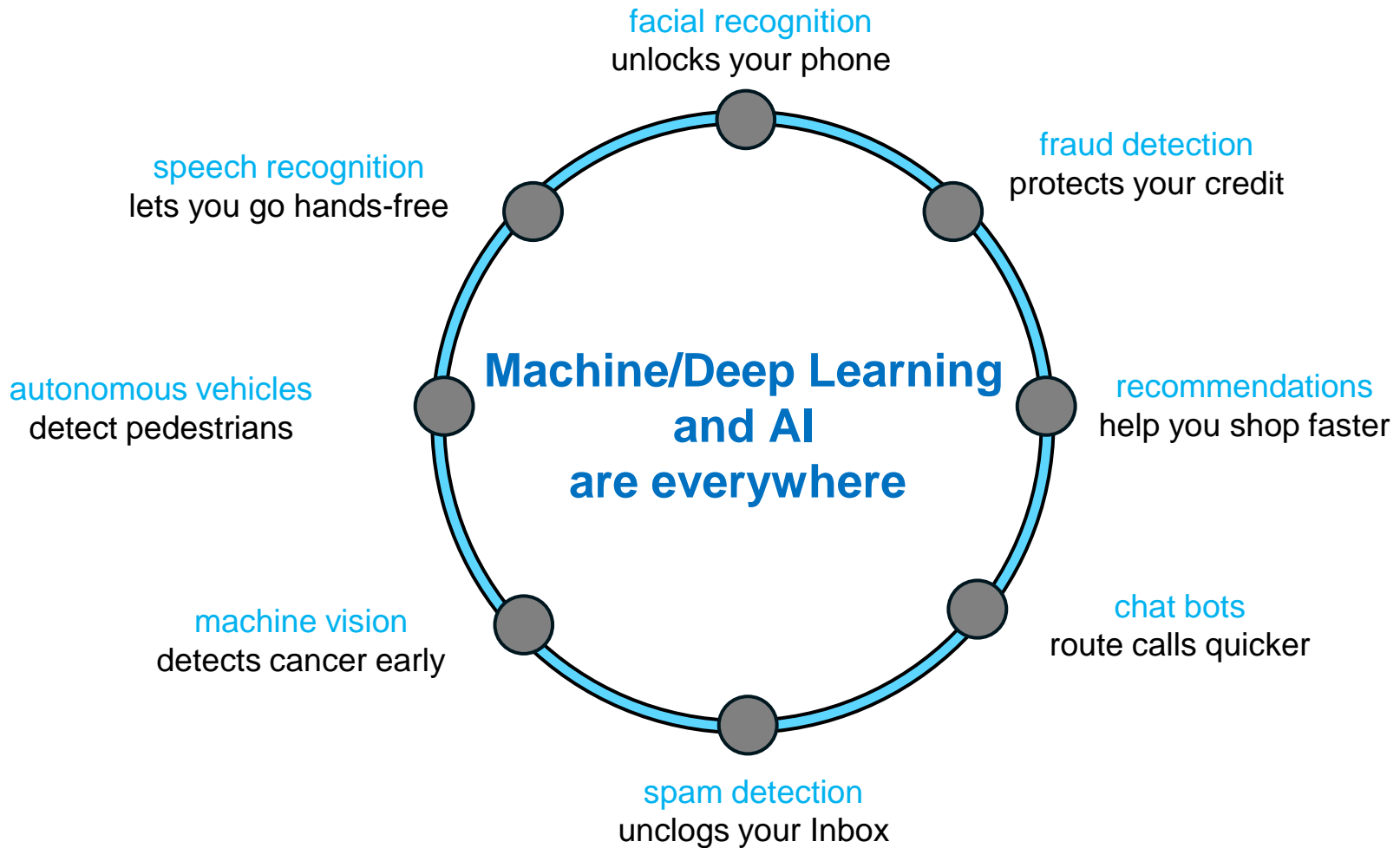
Data + Algorithms = Scored AI Models




# Understanding AI, ML & DL Relationship...



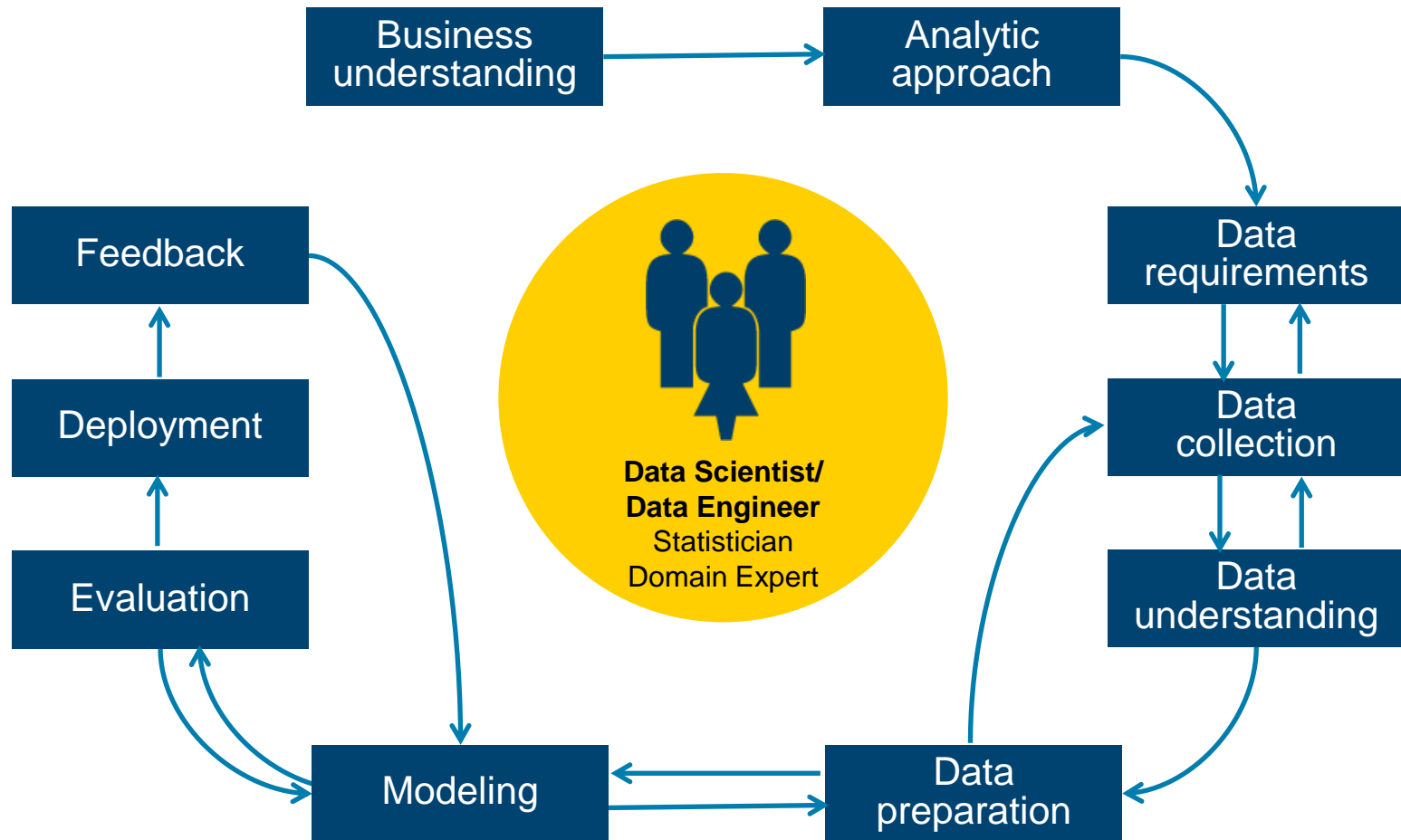
# The future is now



# Introduction to Machine Learning

- Overview
- Data Science Methodology 
- Data Understanding
- Data Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI
- Deep Learning

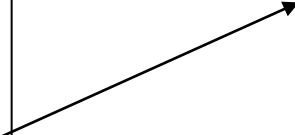
# Data Science Methodology



Known as:

- Scale variables:

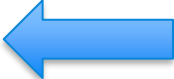
- ### Categorical variables:

- | a1  | a2 | a3 | a4 | a5 | a6 | a7 | a8 | a9 | t |
|---|----|----|----|----|----|----|----|----|---|
|  |    |    |    |    |    |    |    |    |   |

- Known as:

- Label
  - Target variable
  - Dependent variable
- Scale or Categorical

# Introduction to Machine Learning

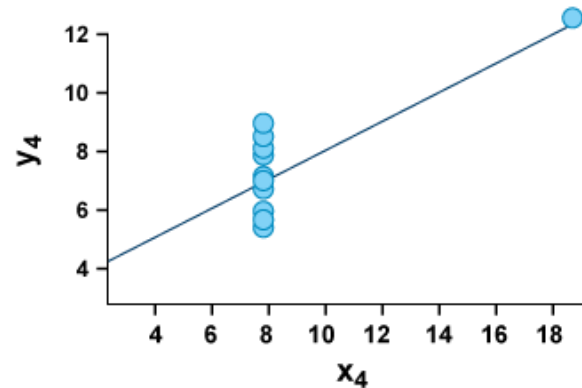
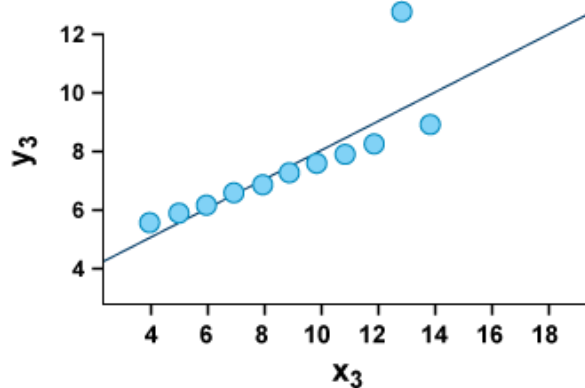
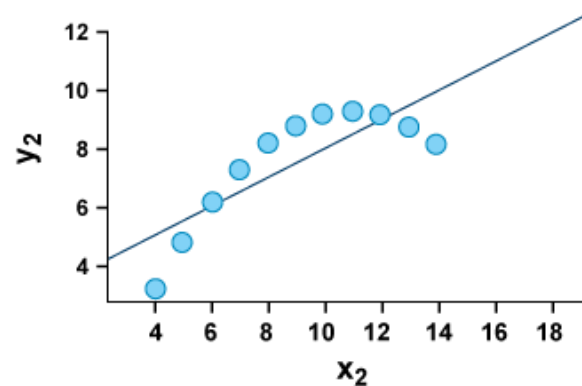
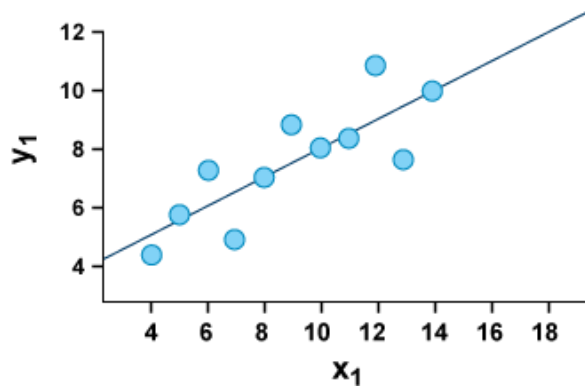
- Overview
- Data Science Methodology
- Data Understanding 
- Data Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI
- Deep Learning



# Data Understanding – Data Audit

- **Data can be missing values**
  - Blank fields
  - Fields with dummy values (9999)
  - Fields with “U” or “Unknown”
- **Data can be corrupt or inconsistent or anomalous:**
  - Data fields can be in the wrong format (strings where numbers are expected)
  - Spurious “End of Line” characters can chop original lines of data into several lines and cause data fields in the wrong place
  - Data entered in different formats: USA / US / United States
  - Data can be anomalous – outlier detection
- **Data can be duplicated**
- **Handling these data quality issues (as part of data preparation) is often referred to as:**
  - Data cleansing

# Data Understanding: Visualizations



The four data sets have similar statistical properties:

- The mean of  $x$  is 9
- The variance of  $x$  is 11
- The mean of  $y$  is approx. 7.50
- The variance of  $y$  is approx. 4.12
- The correlation is 0.816

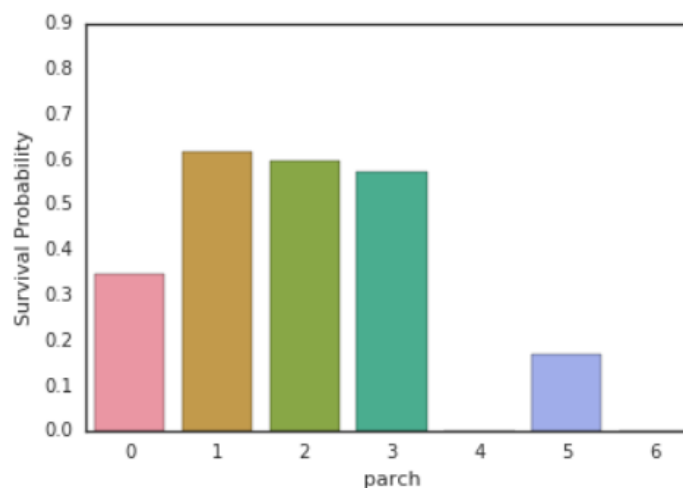
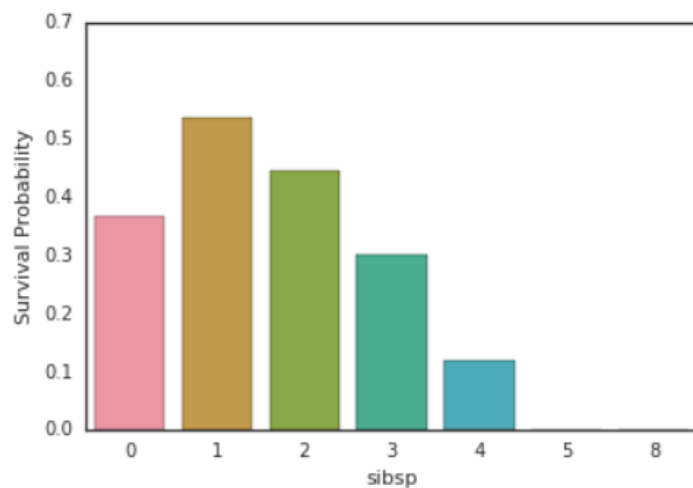
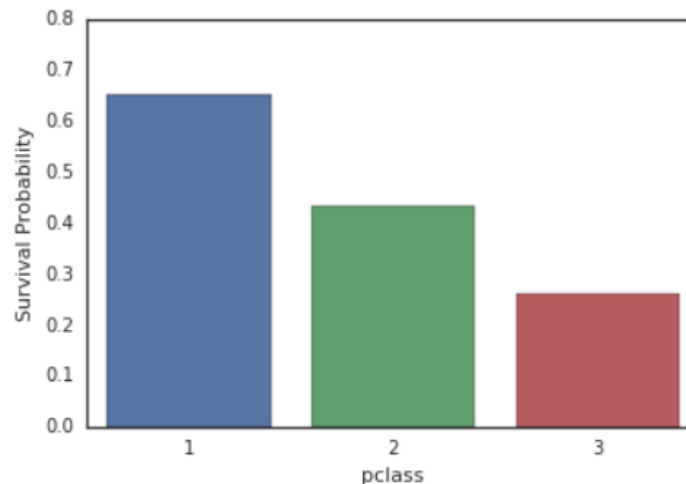
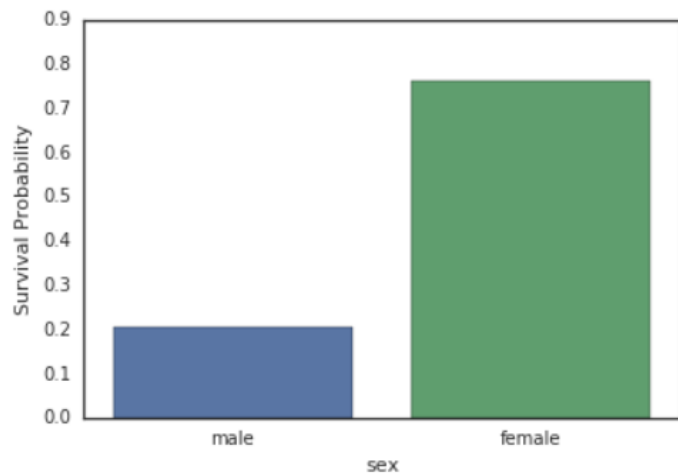
As shown the linear regression lines are approx.  $y = 3.00 + 0.500x$ .

## ■ Anscombe's quartet

- The four datasets have nearly identical statistical properties (mean, variance, correlation), yet the differences are striking when looking at the simple visualization

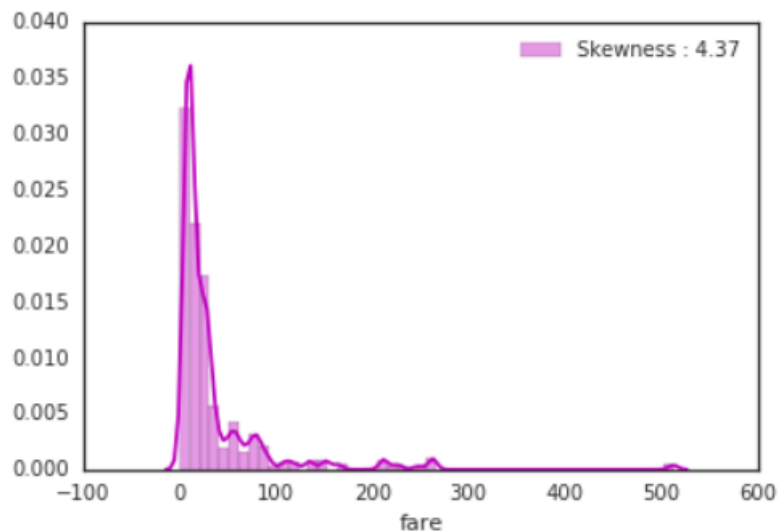
# Data Understanding: Visualizations

- Titanic Data
- Univariate Relationships

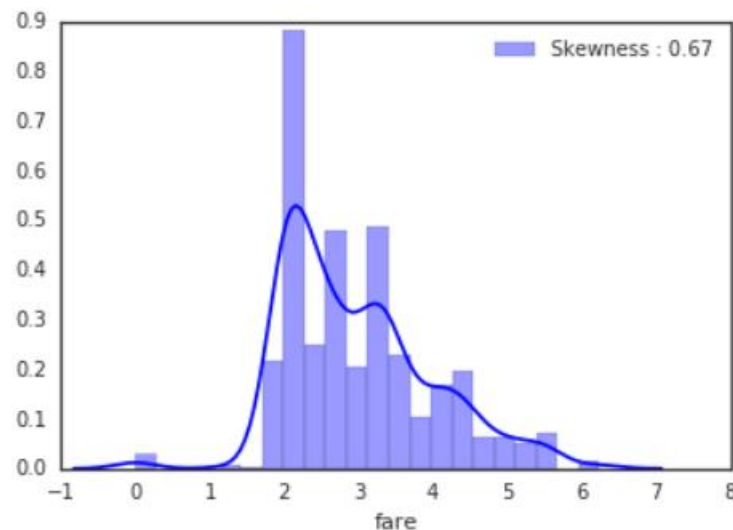


# Data Understanding: Visualizations

- Titanic Data
- Skewed Data




Original Data



After Log Transform

# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation 
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI
- Deep Learning

# Data Preparation

- **Data preparation can be very time consuming depending on:**
  - The state of the original data
    - Data is typically collected in a “human” friendly format
  - The desired final state of the data (as required by the machine learning models and algorithms)
    - The desired final state is typically some “algorithm” friendly format
  - There may be a need for a (long) pipeline of transformations before the data is ready to be consumed by a model:
    - These transformations can be done manually (write code)
    - These transformations can be done through tools

# Data Preparation – Transformation

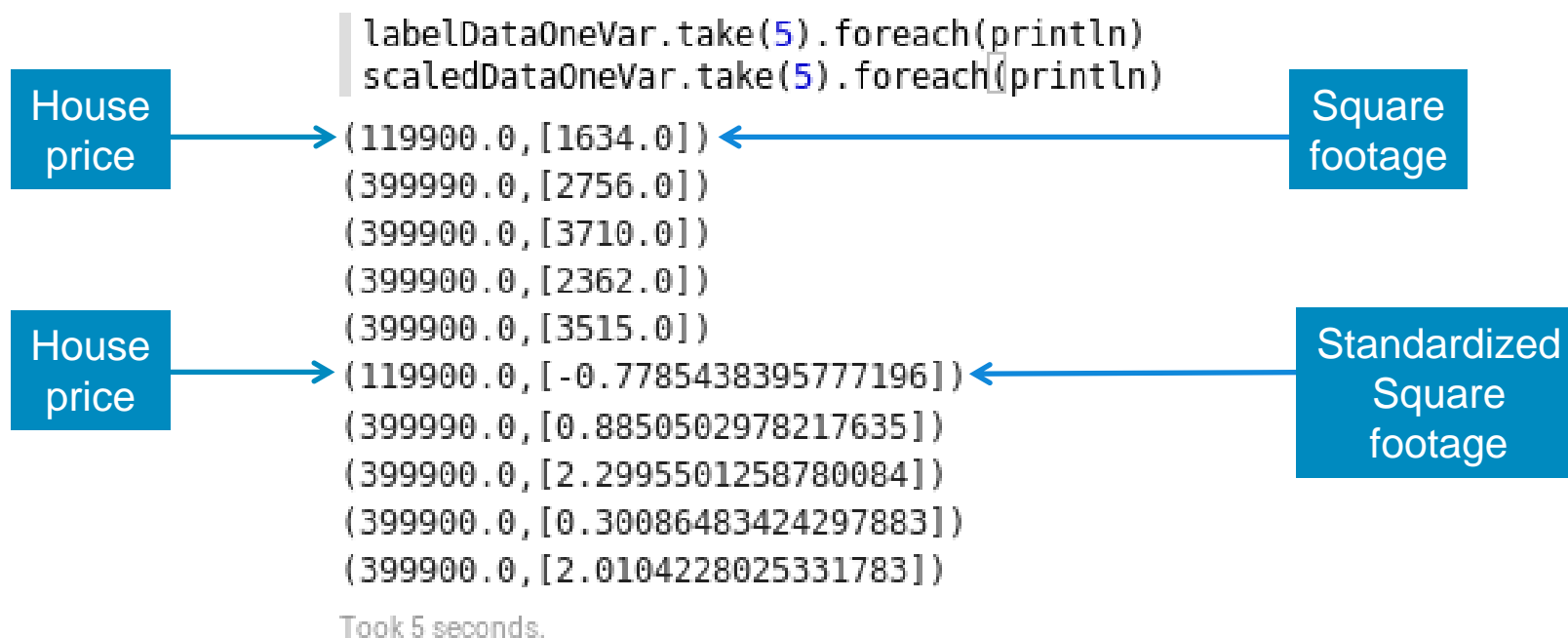
- **Data may need to be transformed to match algorithms requirements:**
  - Tokenizing (typical in text processing)
  - Vectorizing (several algorithms in Spark MLlib require this)
    - Transform data into Vector arrays
    - Can be done manually (write Python or Scala code)
    - Can be done using tools (VectorAssembler in the new ML package)
  - Bucketizing
    - Transform a range of continuous values into a set of buckets

# Data Preparation – Transformation

## ▪ Data may need to be transformed to match algorithms requirements:

### – Standardization

- Transform numerical data to values with zero mean and unit standard deviation
- Linear Regression with SGD in Spark MLlib requires this





# Data Preparation – Transformation

- **Data may need to be transformed to match algorithms requirements:**

- Normalization

- Transform data so that each Vector has a Unit norm.

$$x' = \frac{x}{||x||}$$

- Transform data so that each feature has a value between 0 and 1

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

- Categorical values need to be converted to numbers

- This is required by Spark MLlib classification trees
- Marital Status: {"Widowed", "Married", "Divorced", "Single"}
- Marital Status: {0, 1, 2, 3}
- You cannot do this if the algorithm could infer: Single = 3 X Married 😊

# Data Preparation – Transformation

- **Data may need to be transformed to match algorithms requirements:**
  - Dummy encoding
    - When categorical values cannot be converted to consecutive numbers
    - Marital Status: {"Single", "Married", "Divorced", "Widowed"}
    - Marital Status: {"0001", "0010", "0100", "1000"}
    - This is necessary if the algorithm could make some wrong inference from the numerical based categorical encoding:

# Data Preparation – Dimensionality Reduction

- **Data dimensionality may need to be reduced:**
- **The idea behind reducing data dimensionality is that raw data tends to have two subcomponents:**
  - “Useful features” (aka structure)
  - Noise (random and irrelevant)
  - Extracting the structure makes for better models
  - Examples of applications of dimensionality reduction
    - Extracting the important features in face/pattern recognition
    - Removing stop words when working on text classification
    - Stemming: fishing, fished, fisher → fish
  - Examples methods of dimensionality reduction
    - Principal Component Analysis
    - Singular Value Decomposition
    - Autoencoders

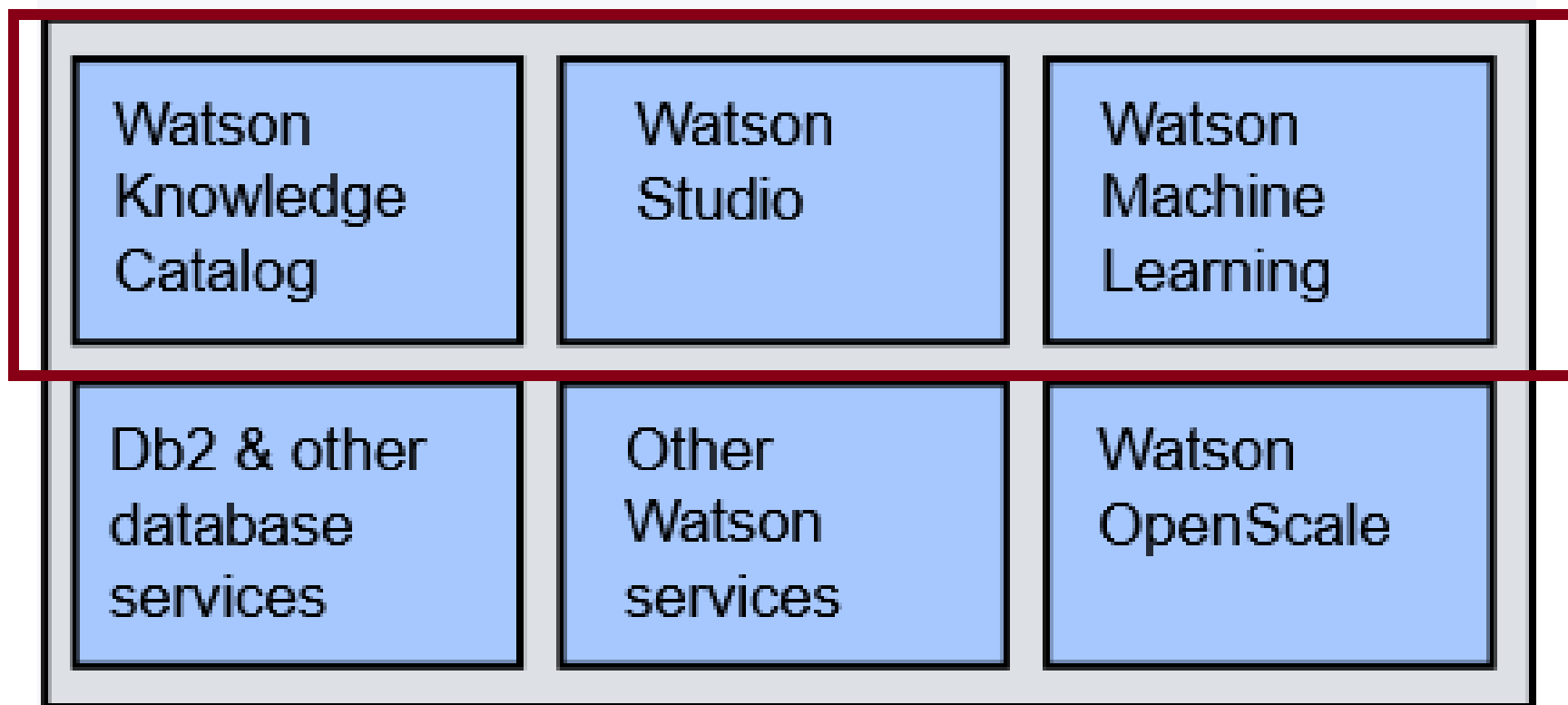
# Lab Overview

## Labs 1, 2

# Cloud Pak for Data Deployment Options

- Cloud Pak for Data as a Service
  - Managed offering provided by IBM
  - Used for today's labs
  
- Cloud Pak for Data
  - Available anywhere Red Hat OpenShift is supported
  - Public Clouds – IBM, Amazon Web Service, Microsoft Azure, Google Cloud
  - On-premise
  
- Cloud Pak for Data System
  - Pre-configured hardware
  - Same capabilities as Cloud Pak for Data
  - On-premise

## Cloud Pak for Data as a Service




## Lab Tips


- Cloud Pak for Data url: [datapatform.cloud.ibm.com](https://datapatform.cloud.ibm.com)
- Labs are in [www.github.com/bleonardb3/ML\\_POT\\_03-25-2021](https://www.github.com/bleonardb3/ML_POT_03-25-2021) repository.
- Instructions for each Lab are in the [README](#) file in the respective Lab folder.
- Cloud development enables making frequent improvements in the user interface. We reviewed the lab instructions and made screen updates so they should be pretty faithful to the user interface. Small differences may occur but shouldn't get in the way of successfully completing the labs.
- Do not use Internet Explorer or Edge as the browser. For Mac users do not use Safari.
- All of the Labs should be done in the project that you created in Lab-1
- **For Lab-1 make sure that you uncheck the “restrict who can be a collaborator” checkbox when creating the project.**

# Github Repository

 bleonardb3 / ML\_POT\_03-25-2021

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

 main ▾

 1 branch

 0 tags


[Go to file](#)

[Add file ▾](#)

[Code ▾](#)



**bleonardb3** Update README.md

ceb3fb8 35 minutes ago  56 commits



Lab-1

Update README.md

35 minutes ago



Lab-2

Update README.md

5 days ago



Lab-3

Update README.md

yesterday



Lab-4

Update README.md

yesterday



Lab-5

Update README.md

4 days ago



Lab-6

Add files via upload

1 hour ago



Lab-7

Add files via upload

21 hours ago



Lab-8

Update README.md

5 days ago



# Github Repository

## Hands-on Introduction to Machine Learning using Watson Studio

### Description:

Work with IBM's Watson Studio in this workshop to build, train, and test machine learning/deep learning models. Participants will be led through the following nine hands-on labs. Note, the first lab is a prerequisite for the other labs. Once Lab-1 is completed, the other labs can be done in any order.

1. **Lab-1** - This lab will set up the environment for the subsequent labs.
2. **Lab-2** - This lab will feature the Watson Studio Data Refinery to demonstrate data profiling, visualization, and data preparation.
3. **Lab-3** - This lab will feature the Watson Studio SPSS modeler to demonstrate visual drag and drop creation of a machine learning model.
4. **Lab-4** - This lab will demonstrate the exciting AutoAI capability to build and deploy an optimized model based on the Titanic data set.
5. **Lab-5** - This lab will use a Jupyter Notebook and the XGBoost library to apply machine learning to a classification problem in the healthcare profession. The Watson Machine Learning API will then be used to save and deploy the model.
6. **Lab-6** - This lab will feature Watson OpenScale. IBM Watson OpenScale is an open platform that helps remove barriers to enterprise-scale AI by supporting bias mitigation, accuracy, and explainability of outcomes among other features.
7. **Lab-7** - This lab will use the MNIST computer vision data set to train a deep learning model to recognize handwritten digits. A simple convolutional neural network built using Keras will be submitted to Watson Machine Learning for training. The trained model will be saved in the model repository, deployed, and scored. The lab will use a notebook to programmatically accomplish these tasks.
8. **Lab-8** - This lab will feature IBM's Adversarial Robustness Toolbox (ART). ART is a library dedicated to adversarial machine learning. Its purpose is to allow rapid crafting and analysis of attacks and defense methods for machine learning models. ART provides an implementation for many state-of-the-art methods for attacking and defending classifiers.

# Github Repository

README.md



## Introduction

This lab will set up the Watson Studio environment for subsequent labs and introduce you to the Project features of Watson Studio. Watson Studio is an integrated platform of tools, services, data, and meta-data to help companies and agencies accelerate their shift to be data driven organizations. The platform enables data professionals such as data scientists, data engineers, business analysts, and application developers collaboratively work with data to build, train, deploy machine learning and deep learning models at scale to infuse AI into business to drive innovation. Watson Studio is designed to support the development and deployment of data and analytics assets for the enterprise.

## Objectives:

Upon completing the lab, you will have:

1. Created a project
2. Created an object storage instance and associated it with the project
3. Associated an existing Watson Machine Learning service instance with the project
4. Added a collaborator to the project
5. Created a deployment space.
6. Provision Watson OpenScale

**Step 1. Please click on the link below to download the instructions to your machine.**

[Instructions.](#)

# Lab-1: Set up Environment

[Create Project](#)

## Introduction:

This lab will set up the Watson Studio environment for subsequent labs and introduce you to the Project features of Watson Studio.

## Objectives:

Upon completing this lab, you will know how to:

- Create a project
- Create an object storage instance and associate it with the project
- Associate an existing Watson Machine Learning instance with the project
- Add collaborators to the project
- Add deployment space
- Set up Watson OpenScale

## Labs: 2,3,4 Titanic Data

### Variable Descriptions:

survival	Survival (0 = No; 1 = Yes)
pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)



PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0		113803	53.1	C123
5	0	3	Allen, Mr. William Henry	male	35	0	0		373450	8.05	S
6	0	3	Moran, Mr. James	male		0	0		330877	8.4583	Q
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0		17463	51.8625	E46
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1		349909	21.075	S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2		347742	11.1333	S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0		237736	30.0708	C

## Lab-2: Introduction to the Data Refinery

Prepare  
Data

### Introduction:

In this lab, you will use the Watson Studio Data Refinery to profile data, visualize data, and prepare data for modeling.

### Objectives:

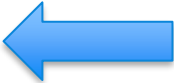
Upon completing the lab, you will know how to:

- Profile the data
- Visualize the data to gain a better understanding
- Prepare the data for modeling
- Run the sequence of data preparation operations on the entire data set.

Proceed with Lab-1 and Lab-2

Return for Presentation at  
11:15 AM EST

# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation
- Categories of Machine Learning 
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI
- Deep Learning

# Categories of Machine Learning

## ■ Supervised learning

- The program is “trained” on a pre-defined set of “training examples”, which then facilitate its ability to reach an accurate conclusion when given new data
- The algorithm is presented with example inputs and their desired outputs (correct results)
- The goal is to learn a general rule that maps inputs to outputs

## ■ Unsupervised learning

- No labels are given to the learning algorithm, leaving it on its own to find structure (patterns and relationships) in its input
- Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning)

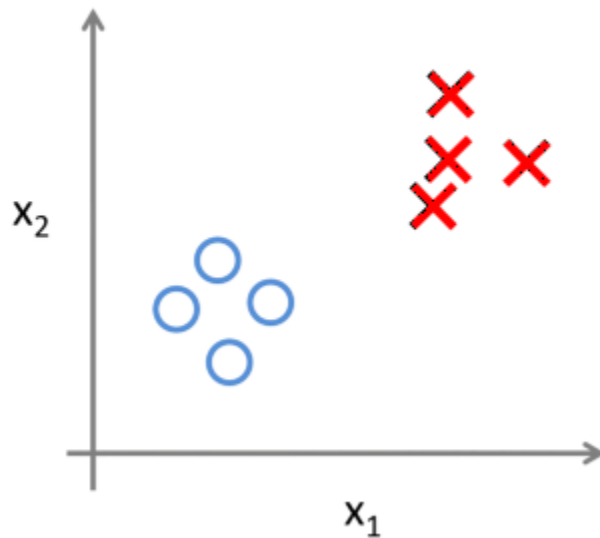
## ■ Reinforcement learning

- Models an agent interacting with an environment. The agent observes the environment, takes an action, and may receive an award (+ or -). The goal is to learn the set of actions to maximize the reward.

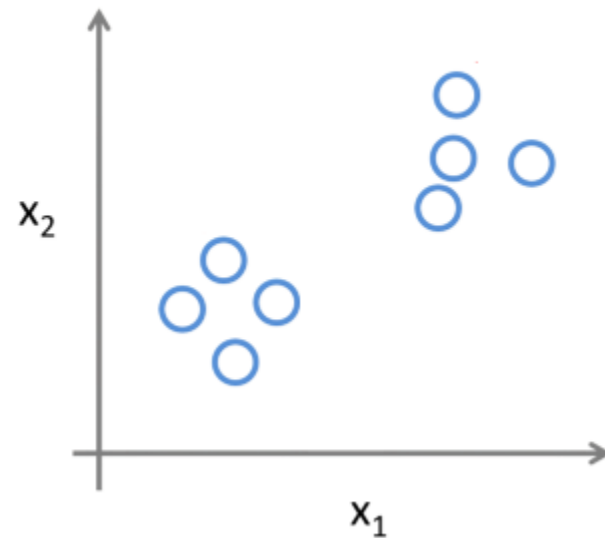


# Supervised vs. Unsupervised Learning

Supervised Learning

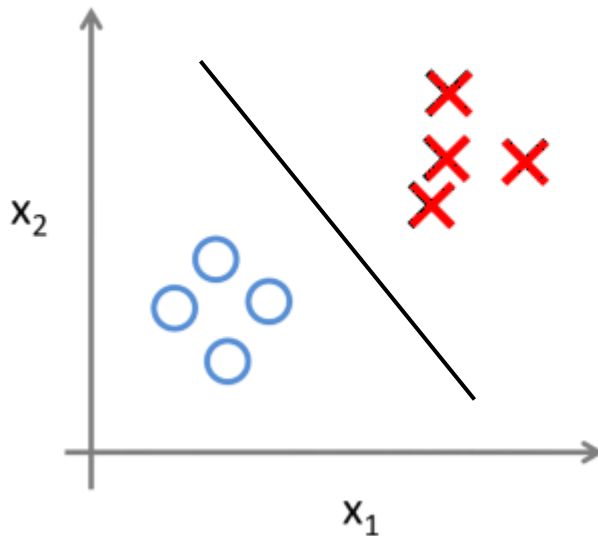


Unsupervised Learning

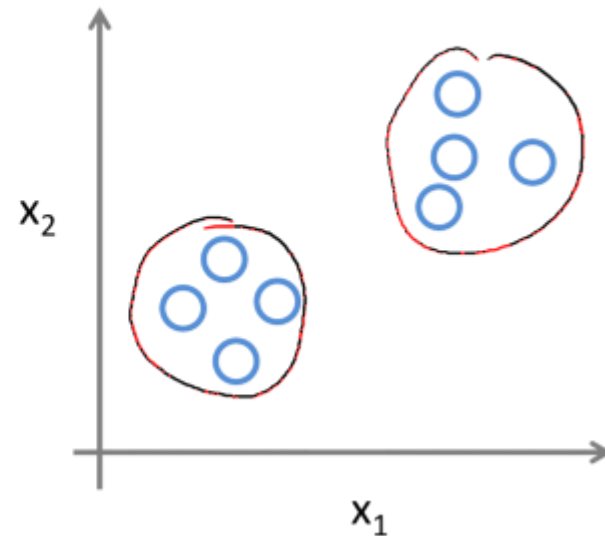


# Supervised vs. Unsupervised Learning

## Supervised Learning




## Unsupervised Learning



# Categories of Machine Learning

Technique	Usage	Algorithms
Classification (or prediction)	<ul style="list-style-type: none"><li>• Used to predict group membership (e.g., will this employee leave?) or a number (e.g., how many widgets will I sell?)</li></ul>	<ul style="list-style-type: none"><li>• Decision Trees</li><li>• Logistic Regression</li><li>• Random Forests</li><li>• <b>Naïve Bayes</b></li><li>• Linear Regression</li><li>• Lasso Regression</li><li>etc</li></ul>
Segmentation	<ul style="list-style-type: none"><li>• Used to classify data points into groups that are internally homogenous and externally heterogeneous.</li><li>• Identify cases that are unusual</li></ul>	<ul style="list-style-type: none"><li>• K-means</li><li>• Gaussian Mixture</li><li>• Latent Dirichlet allocation</li><li>etc</li></ul>
Association	<ul style="list-style-type: none"><li>• Used to find events that occur together or in a sequence (e.g., market basket)</li></ul>	<ul style="list-style-type: none"><li>• FP Growth</li><li>etc</li></ul>

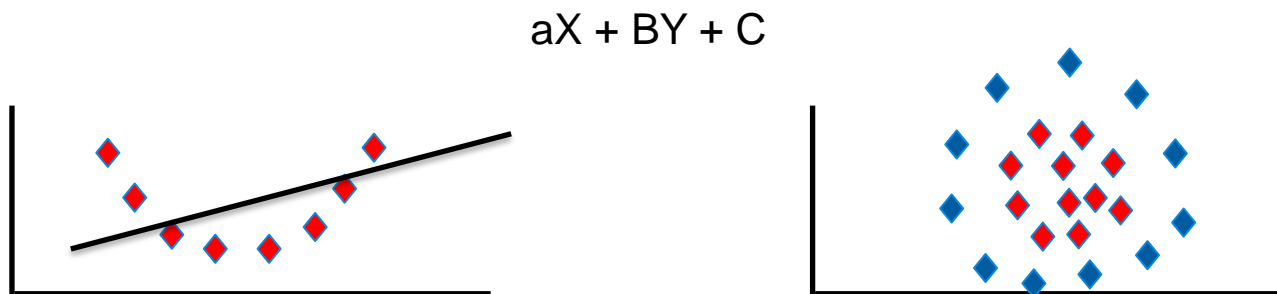
# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation
- Categories of Machine Learning
- Learning Challenges 
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI
- Deep Learning

# Learning challenges

## ■ Under fitting:

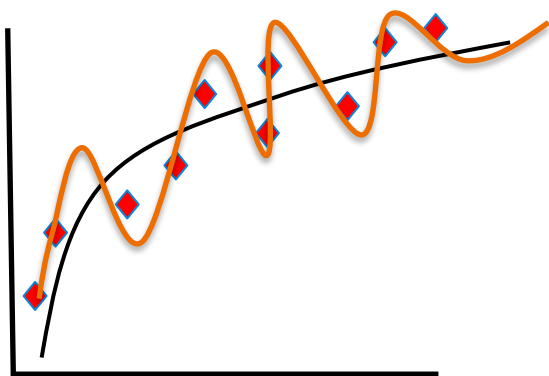
- Not knowing enough “basic” concepts, i.e. not being well-equipped enough to tackle learning at hand:
  - You can’t study calculus without knowing some algebra.
  - You can’t learn playing hockey without knowing how to skate.
  - You can’t learn polo without knowing how to ride.
- This can lead to under fitting in Machine Learning: The chosen model is just not “sophisticated”, “rich”, enough to capture the concept.



# Learning challenges

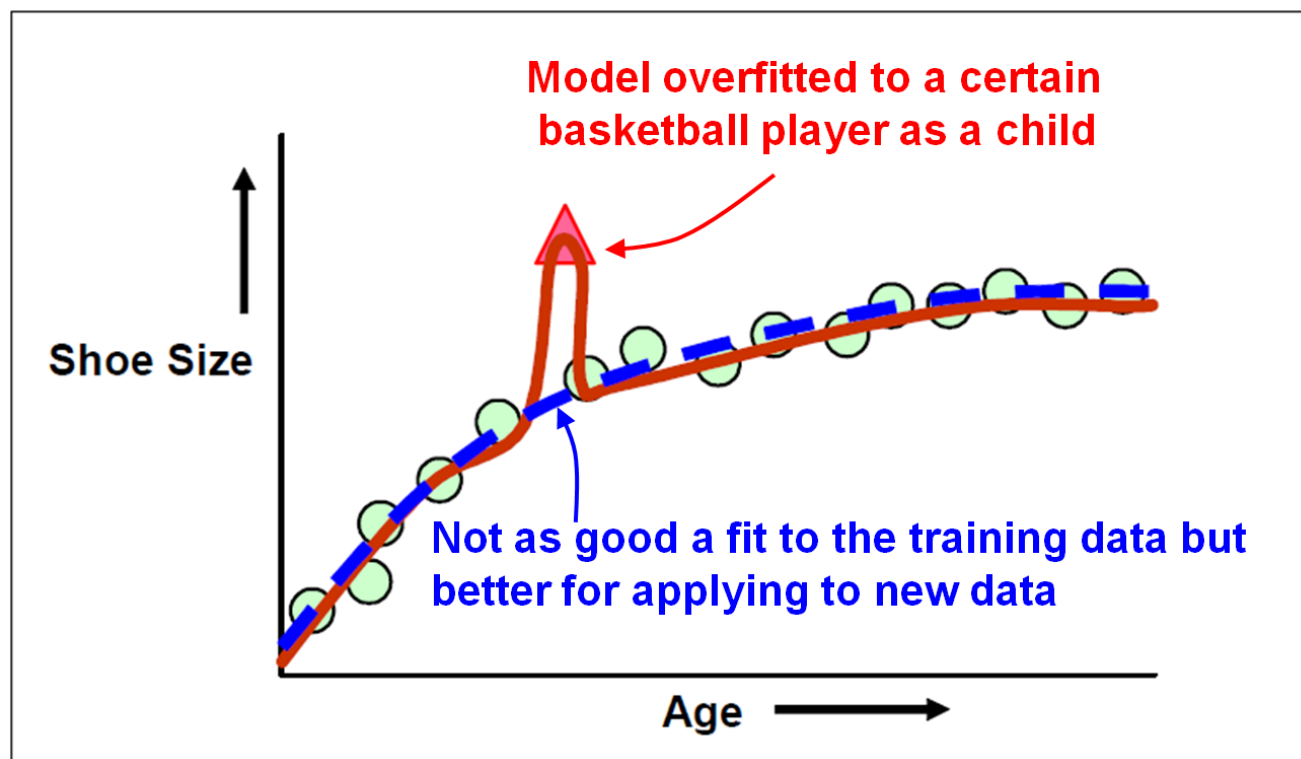
- Over fitting:

- Hyper-sensitivity to minor fluctuations, ending up in modeling a lot of the unwanted noise in the data:
- This can lead to over fitting in Machine Learning.



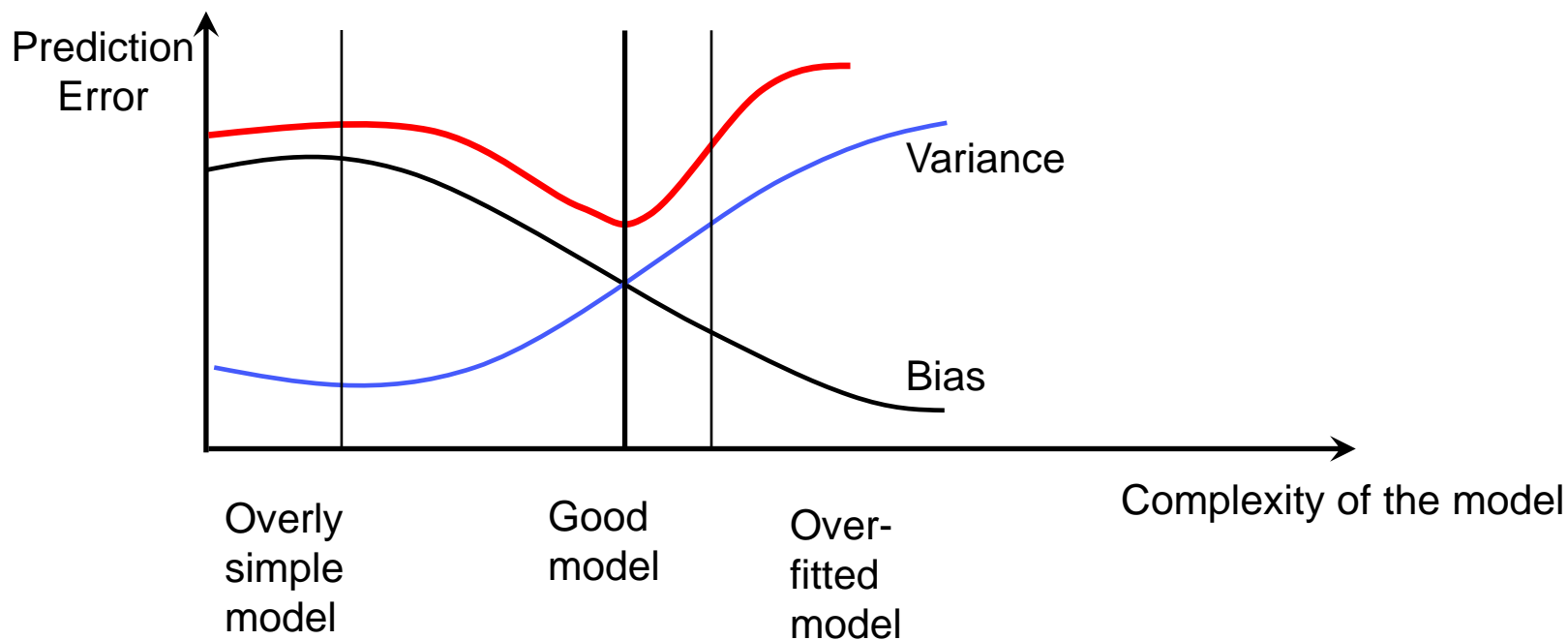
# Model overfitting

- When building a predictive model, there is a risk of overfitting the model to the training data.
- The model fits the training data very well, but it does not perform well when applied to new data.



# Learning challenges

- Compromise between bias and variance:





# Graphical illustration of bias vs variance

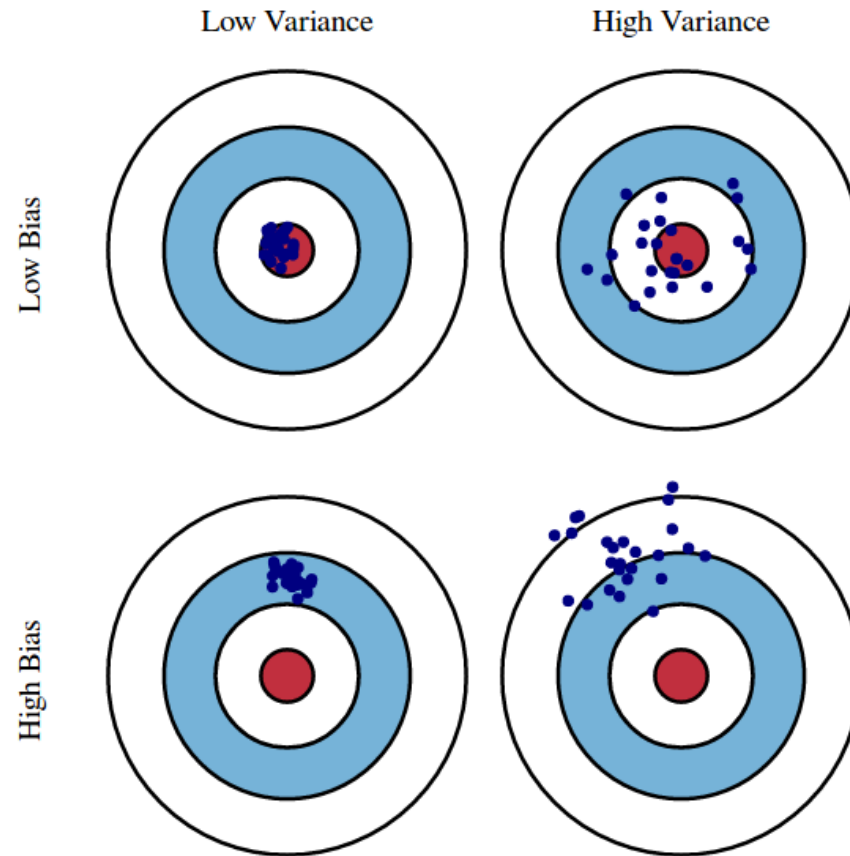
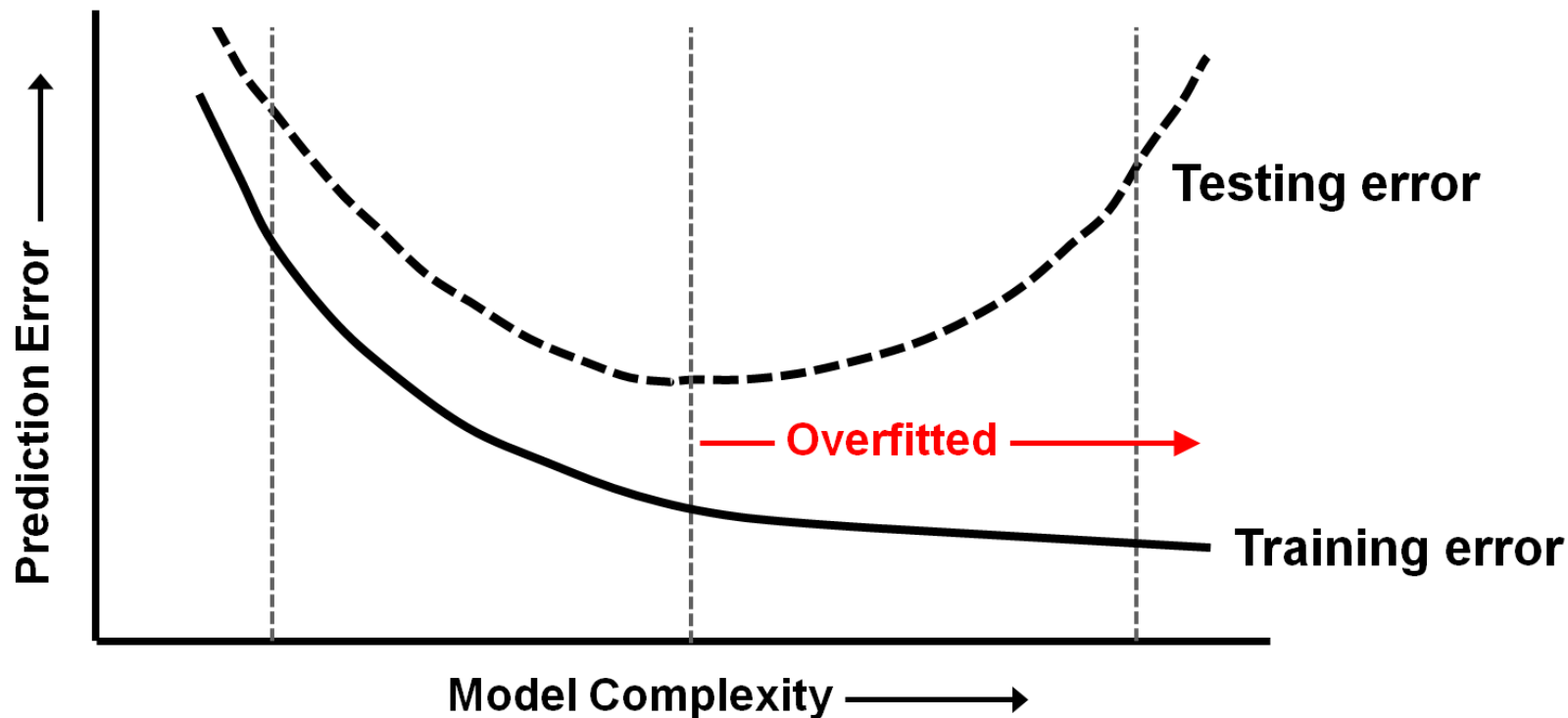
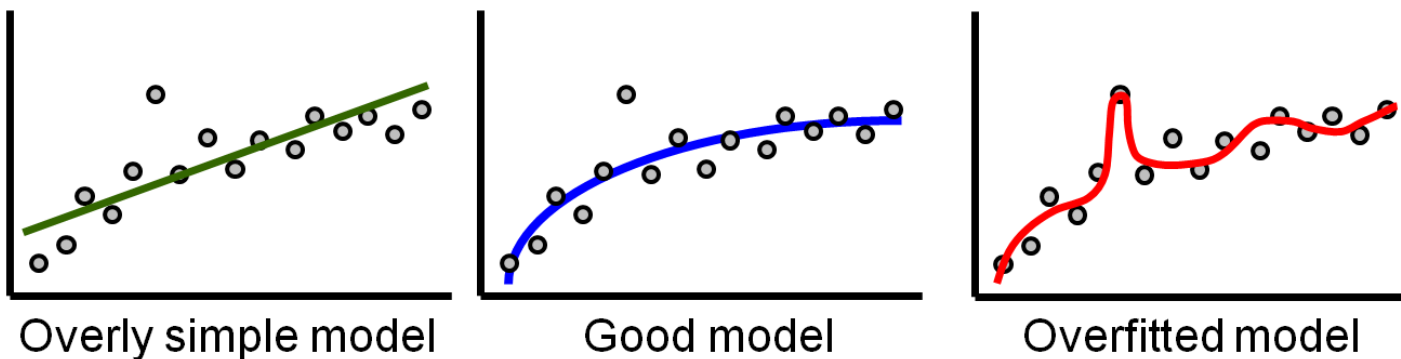


Fig. 1 Graphical illustration of bias and variance.

Source: <http://scott.fortmann-roe.com/docs/BiasVariance.html>

# Indication of Overfitting



# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Trusted AI
- Model Evaluation
- Deep Learning



# Classification – Naïve Bayes (supervised)

- **Two or more outcomes.**
- **Assumes independence among explanatory variables, which is rarely true (thus “naïve”).**
- **Despite its simplicity, often performs very well... widely used.**
- **Significant use cases:**
  - Text categorization (spam vs. legitimate, sports or politics, etc.) using word frequencies as the features
  - Medical diagnosis (e.g., automatic screening)
  - Check a piece of text expressing positive emotions, or negative emotions?
  - Used for face recognition software.
- **Maximum conditional probability**
  - $Prob(Target|Input) = Prob(Input|Target) * \frac{Prob(Target)}{Prob(Input)}$

# Classification – Naïve Bayes

Input →

Outlook	Temp	Humidity	Windy	Play golf
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

← Target

# Classification – Naïve Bayes

Frequencies and probabilities for the weather data:

outlook			temperature			humidity			windy			play	
	yes	no		yes	no		yes	no		yes	no	yes	no
sunny	2	3	hot	2	2	high	3	4	false	6	2	9	5
overcast	4	0	mild	4	2	normal	6	1	true	3	3		
rainy	3	2	cool	3	1								
	yes	no		yes	no		yes	no		yes	no	yes	no
sunny	2/9	3/5	hot	2/9	2/5	high	3/9	4/5	false	6/9	2/5	9/14	5/14
overcast	4/9	0/5	mild	4/9	2/5	normal	6/9	1/5	true	3/9	3/5		
rainy	3/9	2/5	cool	3/9	1/5								

Today's weather prediction sunny, cool, high humidity, windy → play golf ??

**$Prob(\text{sunny}|\text{yes}) * Prob(\text{cool}|\text{yes}) * Prob(\text{high humidity}|\text{yes})Prob(\text{windy}|\text{yes})$**

**$Prob(\text{sunny}|\text{no}) * Prob(\text{cool}|\text{no}) * Prob(\text{high humidity}|\text{no})Prob(\text{windy}|\text{no})$**

## Classification – Naïve Bayes

$$\text{Prob}(\text{Input} \mid \text{yes}) = 2/9 * 3/9 * 3/9 * 3/9 = 0.0082$$

$$\text{Prob}(\text{Input} \mid \text{no}) = 3/5 * 1/5 * 4/5 * 3/5 = 0.0577$$

$$P(\text{yes}) = 9/14$$

$$P(\text{no}) = 5/14$$

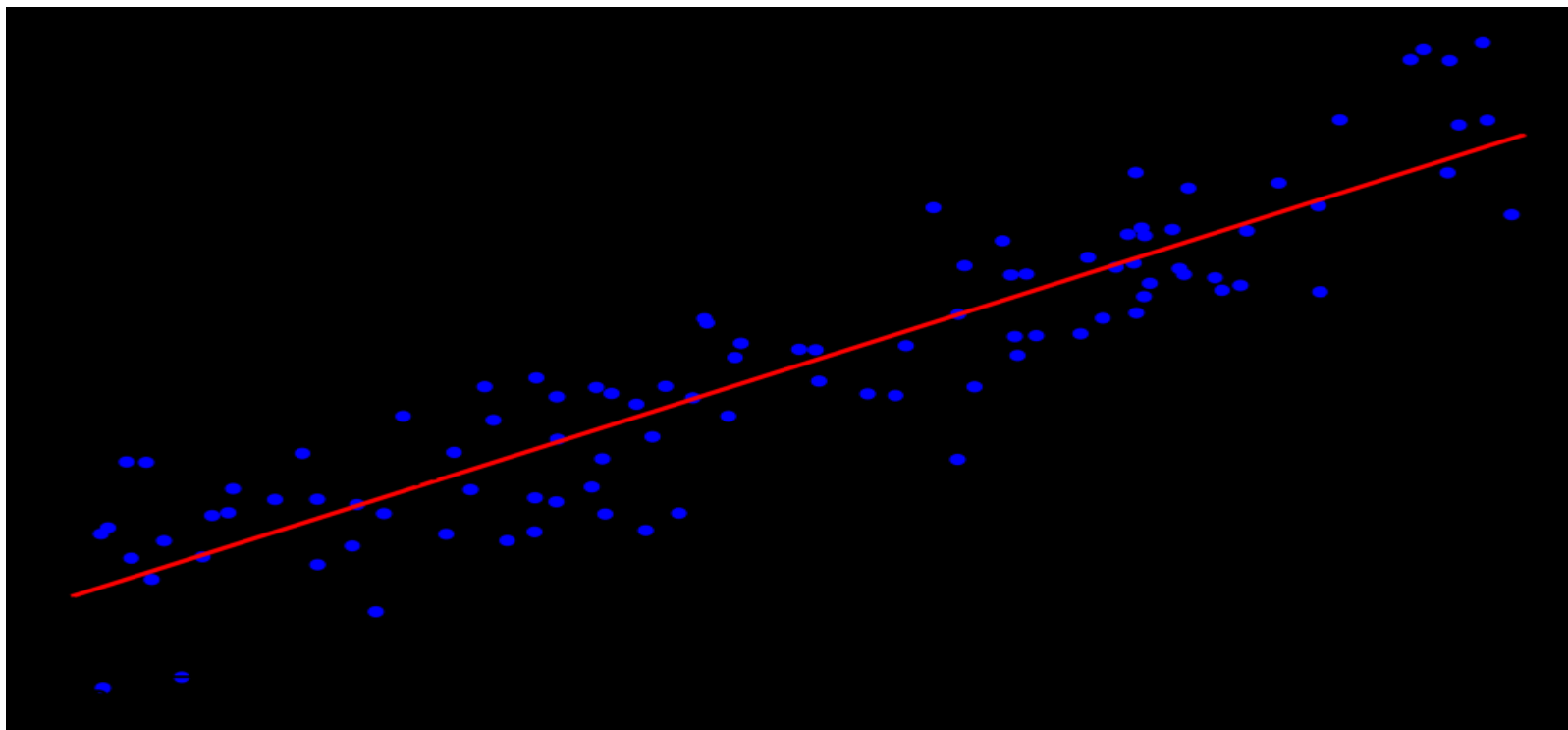
$$\text{Prob}(\text{Input} \mid \text{yes}) * P(\text{yes}) = 0.0082 * (9/14) = 0.0053$$

$$\text{Prob}(\text{Input} \mid \text{no}) * P(\text{no}) = 0.0577 * (5/14) = 0.0206$$

The prediction would be: NO.

## Linear Regression (supervised)

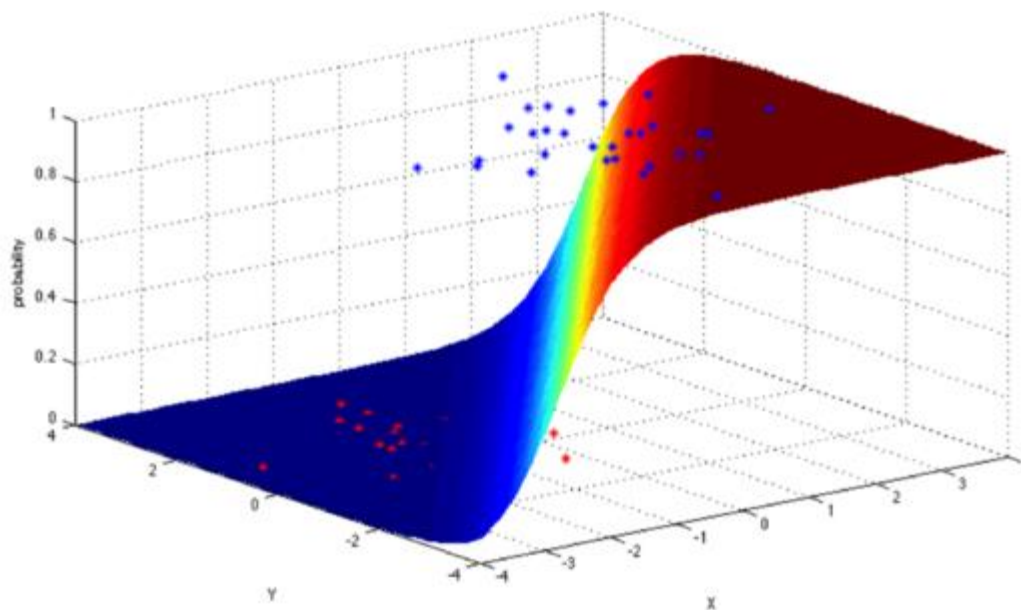
- Draw a line, and then for each of the data points, measure the vertical distance between the point and the line, and add these up; the fitted line would be the one where this sum of distances is as small as possible.
- Use case:
  - Housing prices





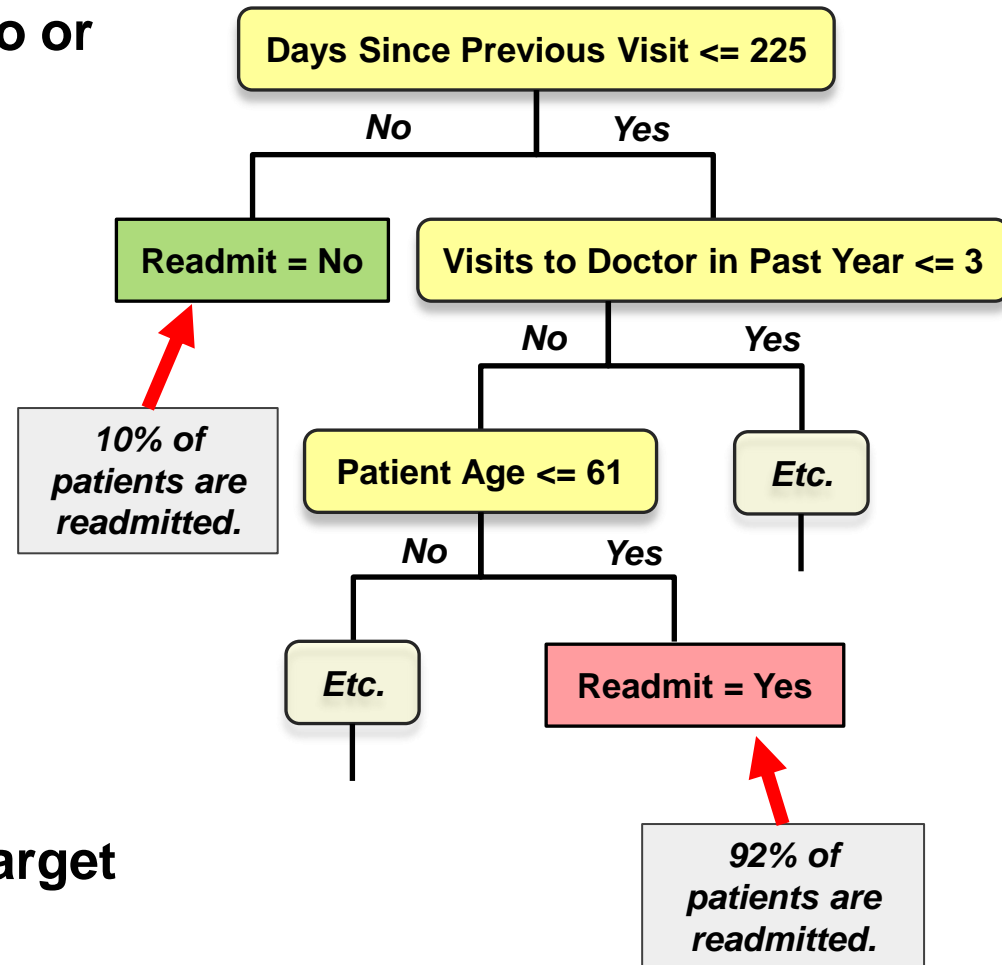
# Logistic Regression (supervised)

- **Logistic regression is a powerful statistical way of modeling a binomial outcome with one or more explanatory variables. It measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution.**

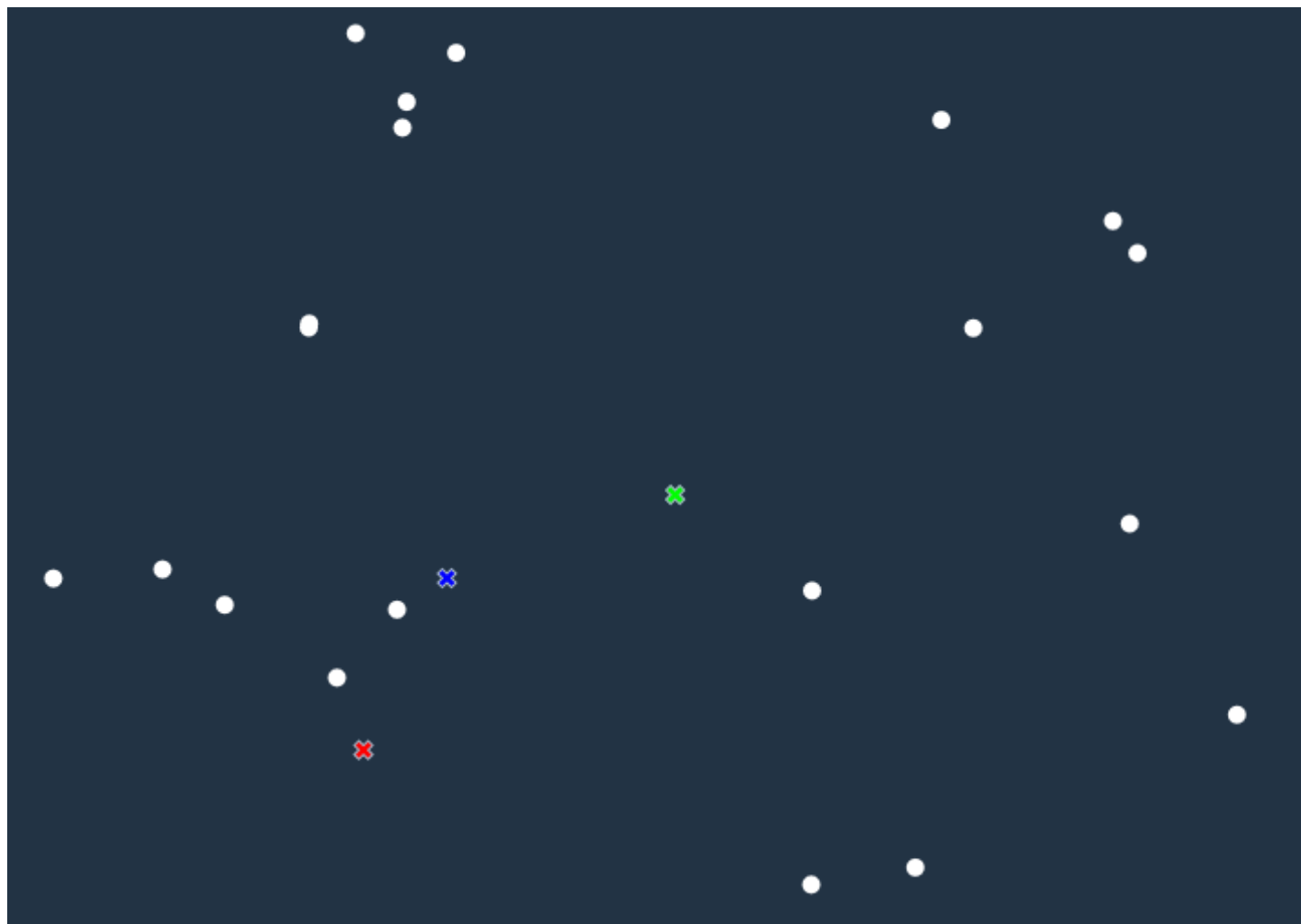


# Classification – Decision tree (supervised)

- **Class variable (target) with two or more outcomes.**
- **Splits records in a tree-like series of nodes along mutually-exclusive paths.**
  - Algorithm decides which variable and threshold value to use at each split
  - New records are predicted (classified) based on the leaf assignment
  - Accurate
  - Explicit decision paths
- **Can also handle continuous target (“regression tree”).**

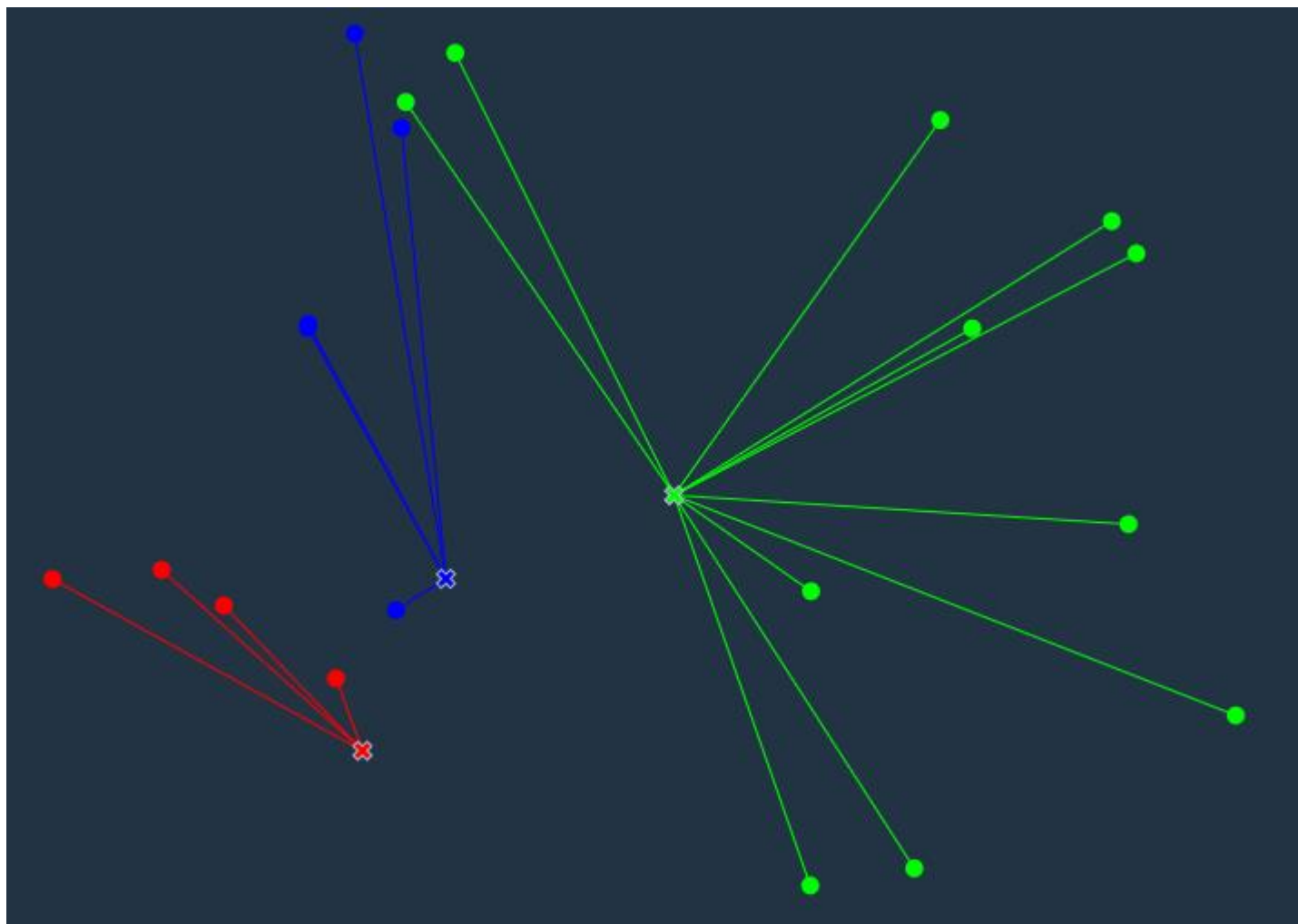


## Clustering – K-means method (unsupervised)



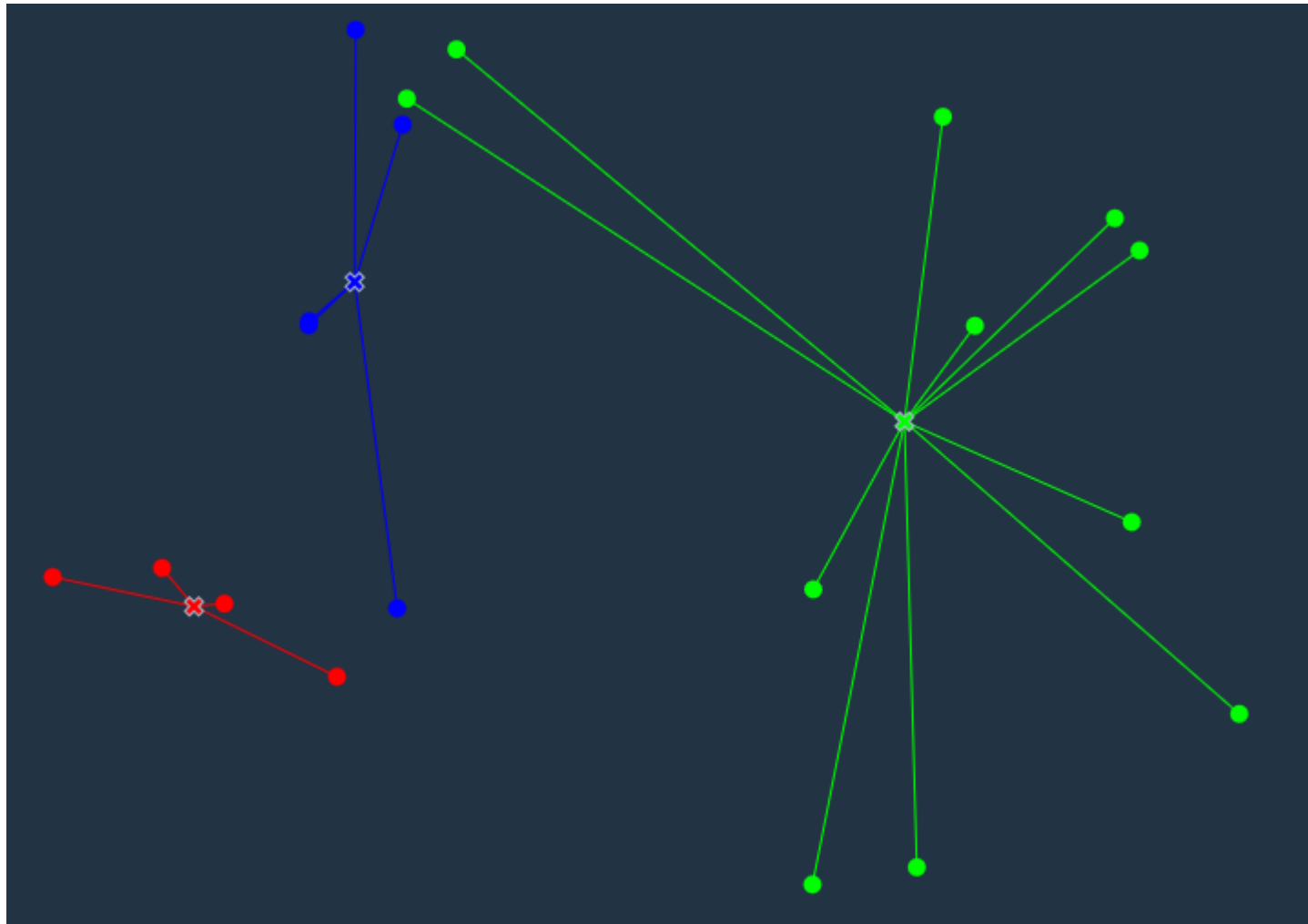
Start with 20 data points and 3 clusters

# Clustering – K-means method



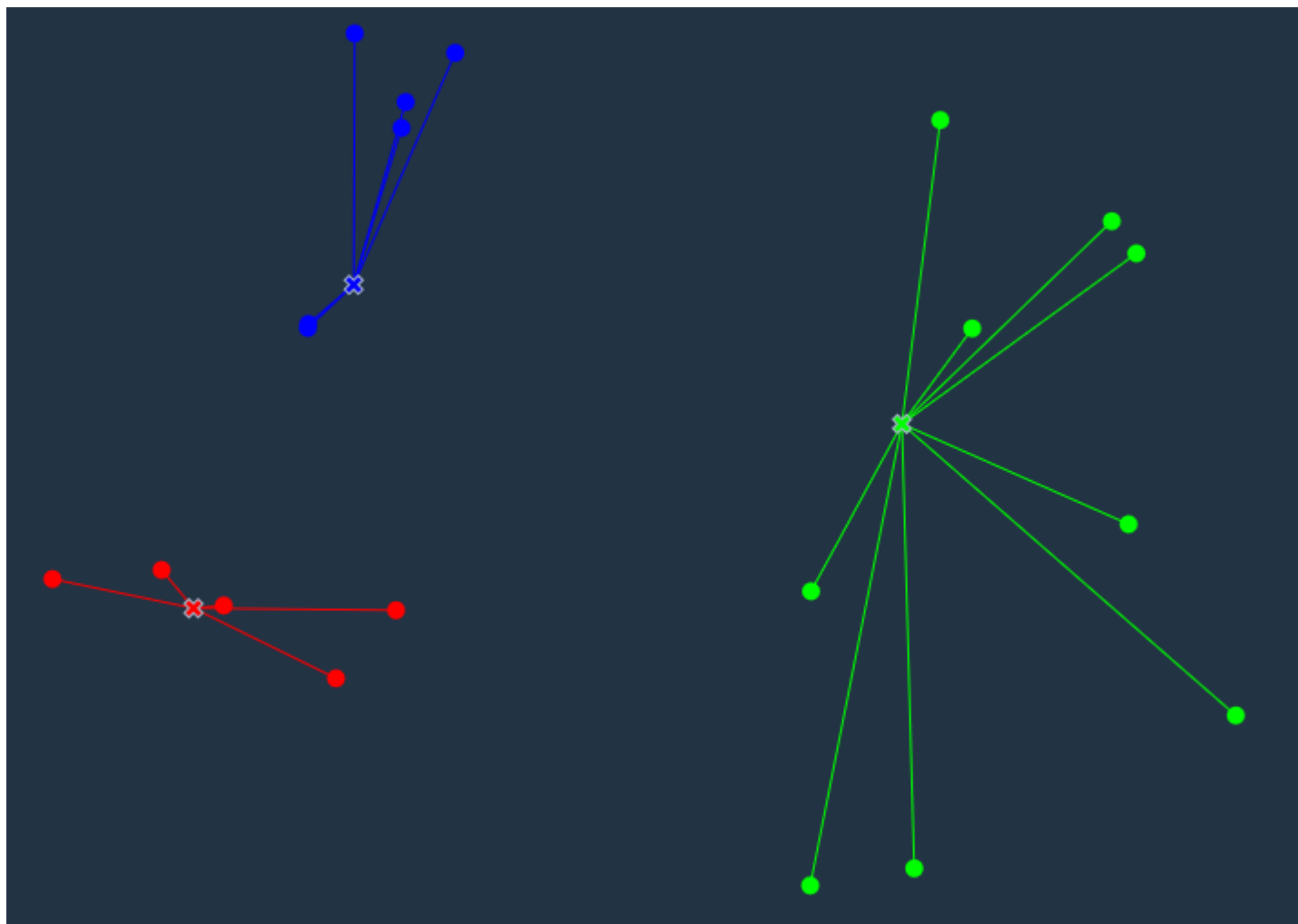
Assign each data point to the nearest cluster

# Clustering – K-means method



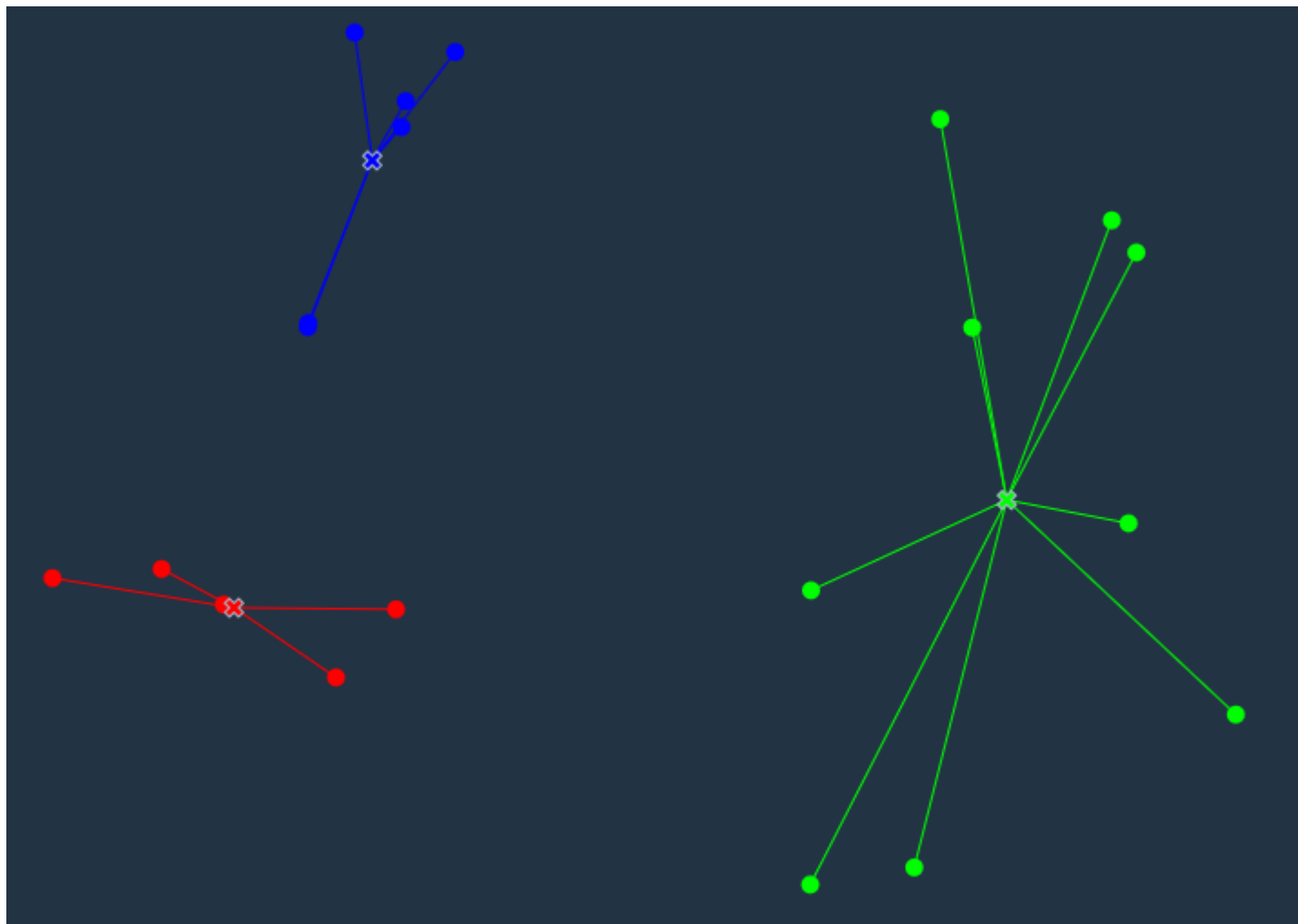
Calculate centroids of new clusters

# Clustering – K-means method



Assign each data point to the nearest cluster

# Clustering – K-means method



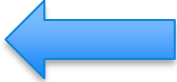
Calculate centroids of new clusters...until convergence

# Ensemble Modeling

- **Use a collection or ensemble of models instead of a single model to create more reliable and accurate predictive models**
  
- **Bagging**
  - New training datasets are generated based on random sampling with replacement of the original data set
  - Models are constructed for each sample and the results are combined
  - Random Forest is bagging applied to Decision Trees
  
- **Boosting**
  - Successive models are built to predict observations misclassified from earlier models.
  - Gradient boosting - train each subsequent model on the residuals (error between predicted value and actual value).



# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation 
- Trusted AI
- Deep Learning

# Training, testing, & validation sets

- During the model development process, supervised learning techniques employ **training** and **testing** sets and sometimes a **validation set**.
  - Historical data with known outcome (*target, class, response, or dependent variable*)
  - Source data randomly split or sampled... mutually exclusive records
- **Why?**
  - Training set → build the model (**iterative**)
  - Validation set → tune the parameters & variables during model building (**iterative**)
    - Assess model quality during training process
    - Avoid overfitting the model to the training set
  - Testing set → estimate accuracy or error rate of model (**once**)
    - Assess model's expected performance when applied to new data



# K-Fold Cross Validation

- Instead of using a separate validation set
- Shuffle Training Samples and sub-divide into “K” folds (groups)
- Train “K” models using K-1 folds as training data and 1 Fold as Test Data
- For example, K=4
  - Model 1 Train on 1,2,3 Test on 4 – calculate and store E1 (Error)
  - Model 2 Train on 2,3,4 Test on 1 – E2
  - Model 3 Train on 3,4,1 Test on 2 - E3
  - Model 4 Train on 4,1,2 Test on 3 - E4
  - $E = (E1+E2+E3+E4)/4$
- A common value for K is 10

# Model Evaluation: Confusion Matrix

**Confusion matrix is more useful measure than simply using prediction accuracy**

- Provides a better visualization of the performance of the algorithm
- Examine the count of each of these boxes

		Predicted	
		Has Disease	No Disease
Actual	Has Disease	true positive (tp) 	false negative (fn) No Treatment
	No Disease	false positive (fp) Unnecessary Treatment	true negative (tn) 



Precision =  $tp / (tp + fp)$       Recall = sensitivity = True Positive Rate  $tp / (tp + fn)$

FPR =  $fp / (fp + tn)$       1 – specificity      ROC = plot of TPR/FPR at different thresholds

# Model Evaluation: Confusion Matrix

**Confusion matrix is more useful measure than simply using prediction accuracy**

- Provides a better visualization of the performance of the algorithm
- Examine the count of each of these boxes

		Predicted	
		Has Disease	No Disease
Actual	Has Disease	true positive (tp) 	false negative (fn) No Treatment
	No Disease	false positive (fp) Unnecessary Treatment	true negative (tn) 



$\text{Precision} = \text{tp} / (\text{tp} + \text{fp})$      
 Recall = sensitivity = True Positive Rate  $\text{tp} / (\text{tp} + \text{fn})$

$\text{FPR} = \text{fp} / (\text{fp} + \text{tn})$      
 1 – specificity     
 ROC = plot of TPR/FPR at different thresholds

# Model Evaluation: Confusion Matrix

**Confusion matrix is more useful measure than simply using prediction accuracy**

- Provides a better visualization of the performance of the algorithm
- Examine the count of each of these boxes

		Predicted	
		Has Disease	No Disease
Actual	Has Disease	true positive (tp) 	false negative (fn) No Treatment
	No Disease	false positive (fp) Unnecessary Treatment	true negative (tn) 

$$\text{Precision} = \text{tp} / (\text{tp} + \text{fp})$$

$$\text{Recall} = \text{sensitivity} = \text{True Positive Rate} = \text{tp} / (\text{tp} + \text{fn})$$



$$\text{FPR} = \text{fp} / (\text{fp} + \text{tn}) \quad 1 - \text{specificity}$$

ROC = plot of TPR/FPR at different thresholds

# Model Evaluation: Confusion Matrix

**Confusion matrix is more useful measure than simply using prediction accuracy**

- Provides a better visualization of the performance of the algorithm
- Examine the count of each of these boxes

		Predicted	
		Has Disease	No Disease
Actual	Has Disease	true positive (tp) 	false negative (fn) No Treatment
	No Disease	false positive (fp) Unnecessary Treatment	true negative (tn) 

$$\text{Precision} = \text{tp} / (\text{tp} + \text{fp})$$

$$\text{Recall} = \text{sensitivity} = \text{True Positive Rate} = \text{tp} / (\text{tp} + \text{fn})$$

$$\text{FPR} = \text{fp} / (\text{fp} + \text{tn}) = 1 - \text{specificity}$$

ROC = plot of TPR/FPR at different thresholds

## Model Evaluation

- When you are building a classifier, it is important to understand the **PREVALANCE** of the condition that you are building a model for,  
i.e. how common or uncommon this condition effectively is...
- Imagine you are working towards building a classifier for some medical condition and your training and testing data sets yield the following model

	Test positive	Test negative
Disease (100)	95 (True Positive)	5 (False Negative)
Normal (100)	5 (False Positive)	95 (True Negative)

**Accuracy = 95%    Recall = 95%    Precision=95%**



## Model Evaluation

- What truly matters to the users of your new model / test (doctors, bankers, practitioners) is the **PREDICTIVE VALUE** of the test:
  - If the test is positive, then what is the actual chance of being sick?
  - Is it 95% ?
- Let's run the test on a population of 1,000,000 where 1% individuals (10,000) are actually suffering from this condition:

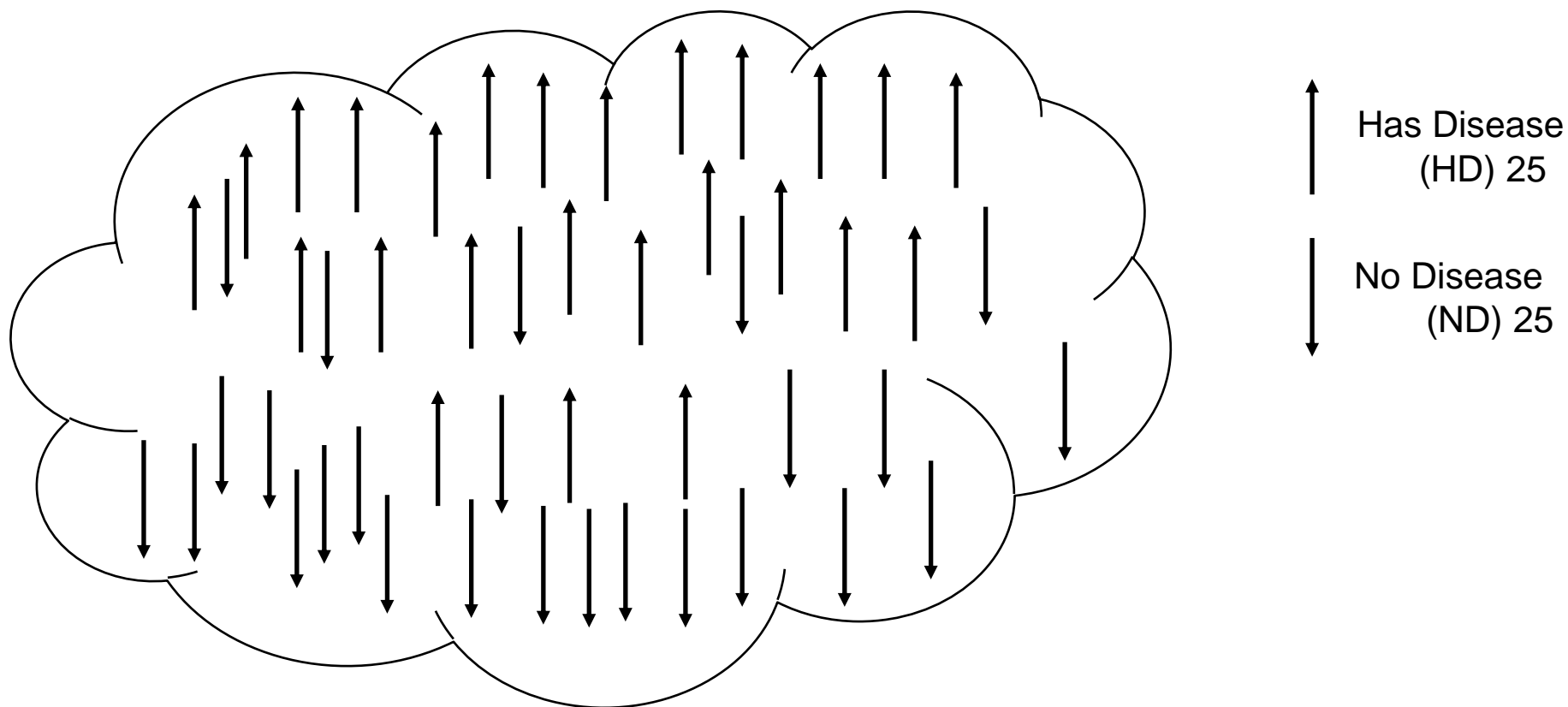
	Test positive	Test negative
Disease (10000)	9500 (95% True Positive)	500 (5% False Negative)
Normal (990000)	49500 (5% False Positive)	940500 (95% True Negative)

**Accuracy = 95%    Precision=16.1%    Recall = 95%**

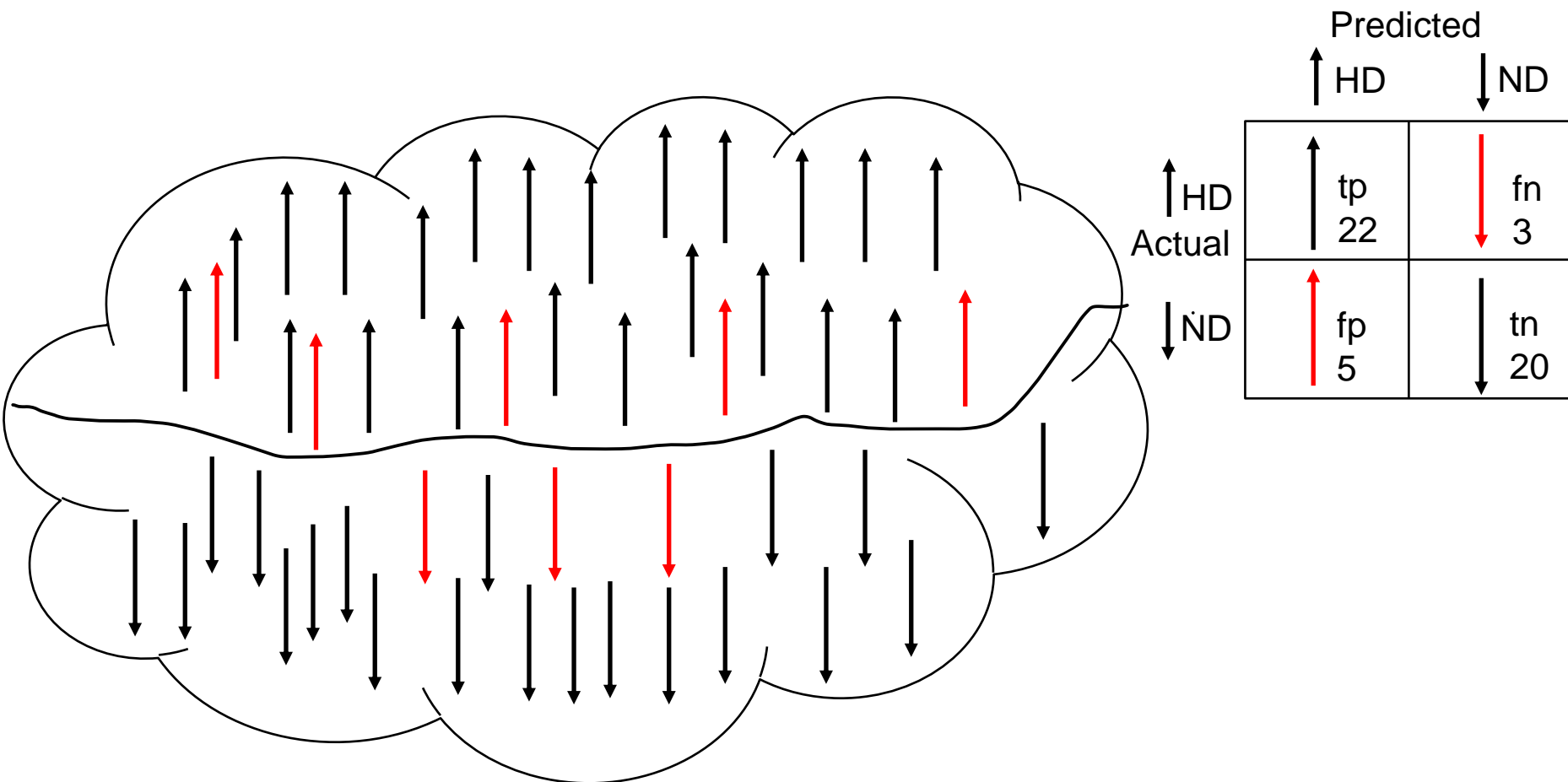
What is happening here:

The condition is RARE and the 5% FALSE POSITIVES are still way higher in numbers than the true positives. Need 99% or higher specificity.

# Model Evaluation – Visual Example



# Evaluation Metrics – Confusion Matrix

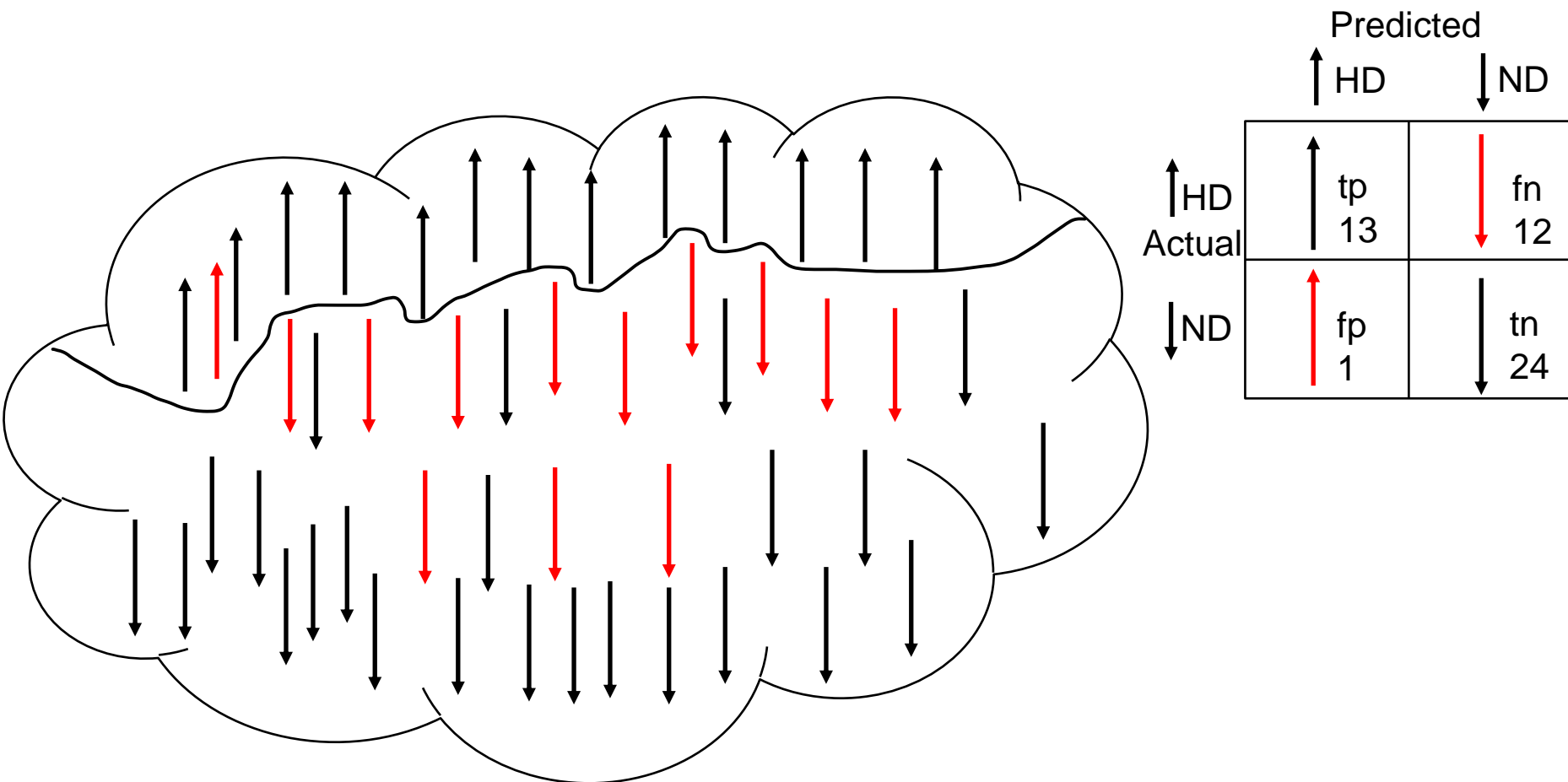


Precision =  $tp / (tp + fp) = 81\%$       Recall = sensitivity = True Positive Rate  $tp / (tp + fn) = 88\%$

FPR =  $fp / (fp + tn) = 20\%$

ROC = plot of TPR/FPR at different thresholds

# Evaluation Metrics – Confusion Matrix



Precision =  $tp / (tp + fp) = 92.3\%$     Recall = sensitivity = True Positive Rate  $tp / (tp + fn) = 52\%$

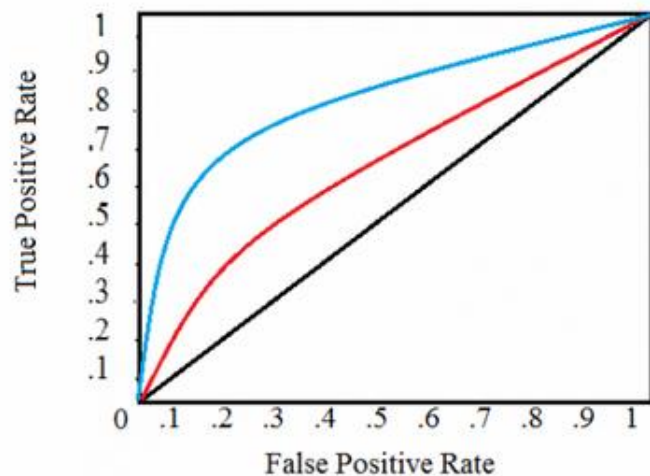
FPR =  $fp / (fp + tn) = 4\%$

ROC = plot of TPR/FPR at different thresholds

## Model Evaluation - Metrics

- **Accuracy** =  $\frac{\text{Correct Predictions}}{\text{Total Predictions}} = \frac{Tp+Tn}{Tp+Tn+Fp+Fn}$
- **F1** =  $2 * \frac{\text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$
- **Area under Receiver Operating Characteristic (ROC)**

[Source](#)



---

# Lab Overview

## Labs 3, 4, 5

## Lab-3: SPSS Modeler

[Build Model](#)[Save and Deploy](#)

### Introduction:

In this lab, you will use the Watson Studio SPSS Modeler capability to explore, prepare, and model trafficking data. The SPSS Modeler is a drag and drop capability to build machine learning pipelines.

### Objectives:

Upon completing this lab, you will have:

- Become familiar with the Watson Studio SPSS Modeler capability
- Profiled the data set
- Explored the data set with visualizations
- Transformed the data
- Trained/Evaluated a machine learning mode.

## Lab-4: AutoAI

[Build Model](#)[Save and Deploy](#)

### Introduction:

In this lab, you will use IBM's Watson Machine Learning GUI to train, evaluate, and deploy a Watson Machine Learning model based on the Titanic dataset.

### Objectives:

Upon completing the lab, you will:

- Become familiar with the AutoAI feature of Watson Studio.
- Train/Evaluate a machine learning model
- Deploy a machine learning model.



## Lab-5: Heart Disease Notebook

[Build Model](#)[Save and Deploy](#)

### Introduction:

In this lab, you will use a Jupyter Notebook to train a model using the XGBoost library to classify whether a person has heart disease or not. In addition to training a model, the notebook also explains how to persist a trained model to the IBM Watson Machine Learning repository, and deploy the model as a REST service.

### Objectives:

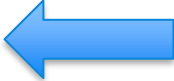
Upon completing the lab, you will know how to:

- Load a CSV file into Pandas DataFrame.
- Prepare data for training and evaluation.
- Create, train, and evaluate a XGBoost model.
- Visualize the importance of features that were used to train the model.
- Use cross validation to select optimal model hyperparameters based on a parameter grid
- Persist best model in Watson Machine Learning repository using Python client library.
- Deploy the model for online scoring using the Watson Machine Learning's REST APIs

Proceed with Lab-3, Lab-4,  
Lab-5

Return for Presentation at  
02:30 PM EST

# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI 
- Deep Learning

# Trusted AI

**Machine Learning is used in many high-stakes decision-making applications**



Credit



Employment



Healthcare



Self-Driving  
Cars

# Our vision for Trusted AI

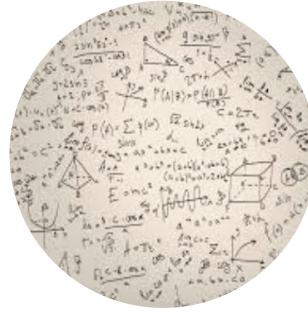
*Pillars of trust, woven into the lifecycle of an AI application*



**Is it  
accurate?**



**Is it  
fair?**



**Is it easy to  
understand  
?**



**Did anyone  
tamper  
with it?**

# Watson OpenScale

## Trust and Transparency

- Intelligently delivers bias mitigation help
- Provides traceability & auditability of AI predictions made in production applications
- Tracks AI accuracy in applications
- Explains an outcome in business terms
- Provides drift detection

## Automation

- Automatically detects and mitigates bias in model output, without affecting currently deployed model or outcomes

## Open by Design

- Monitor models deployed on third party model server engines
- Deploy behind enterprise firewall or on IaaS provider.

## Model build / train frameworks



## Model serving environments



86

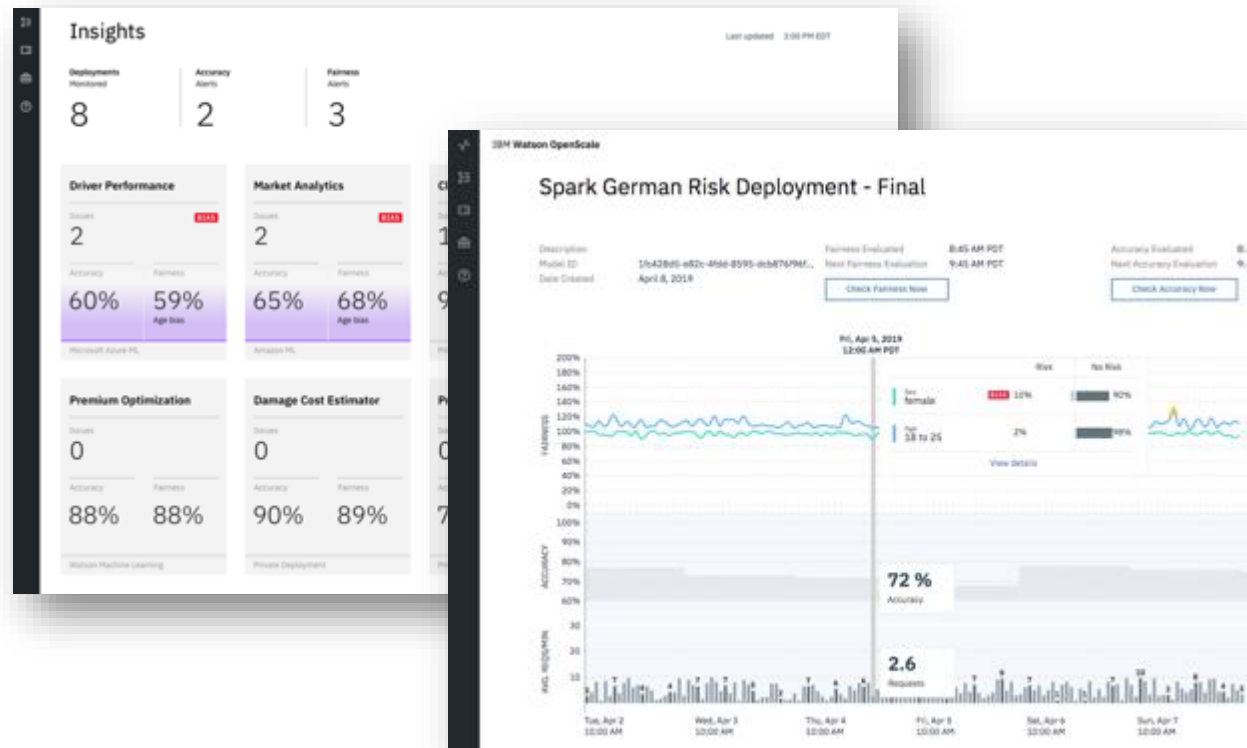
# OpenScale Operations Dashboard

## Description:

Monitor deployed models in a single dashboard that can be filtered by deployment making it easy to manage AI in apps

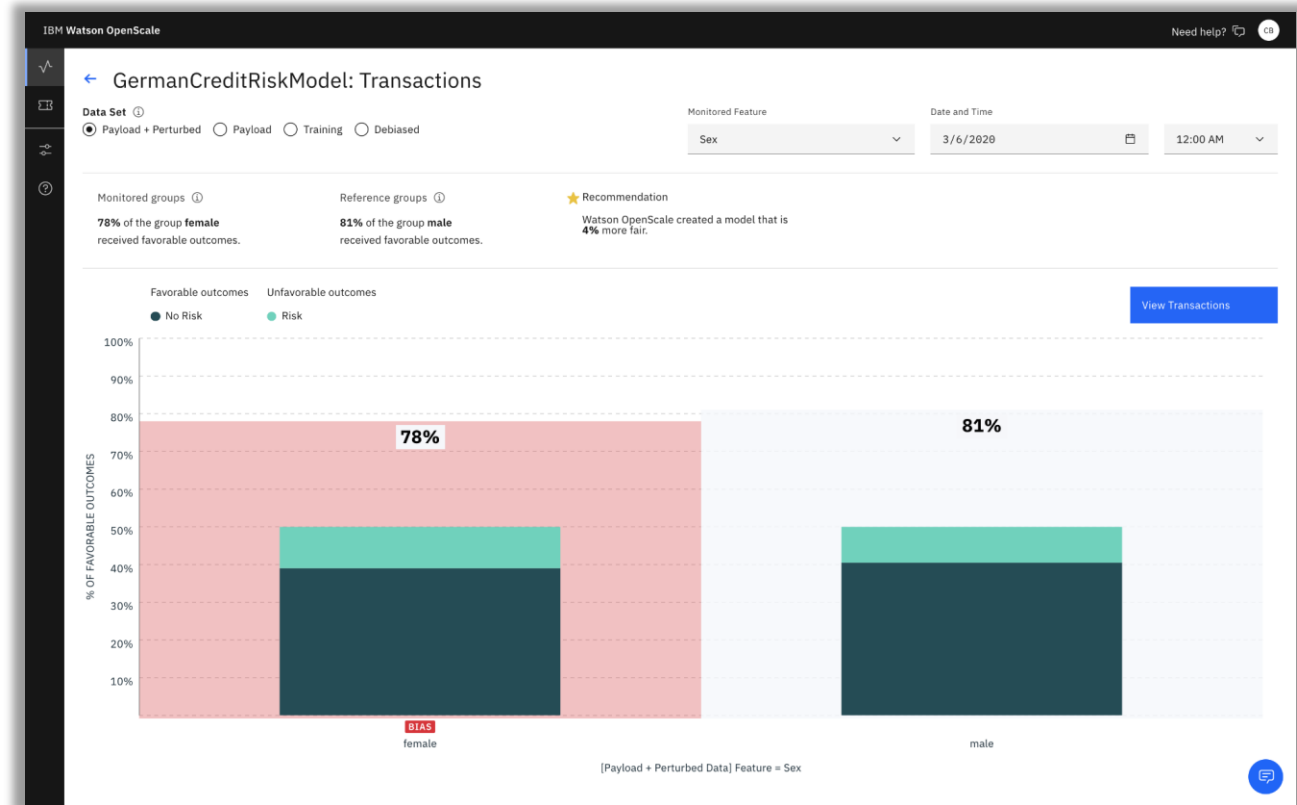
## Value:

- Configure alerts or actions to be triggered when KPIs exceed threshold, ensuring model quality for improve business outcomes
- Measure model accuracy as it pertains to it's ability to deliver outcomes more accurate than knowledge workers
- Provides “continuous evolution” for your models



# Bias Mitigation – Original Model Output

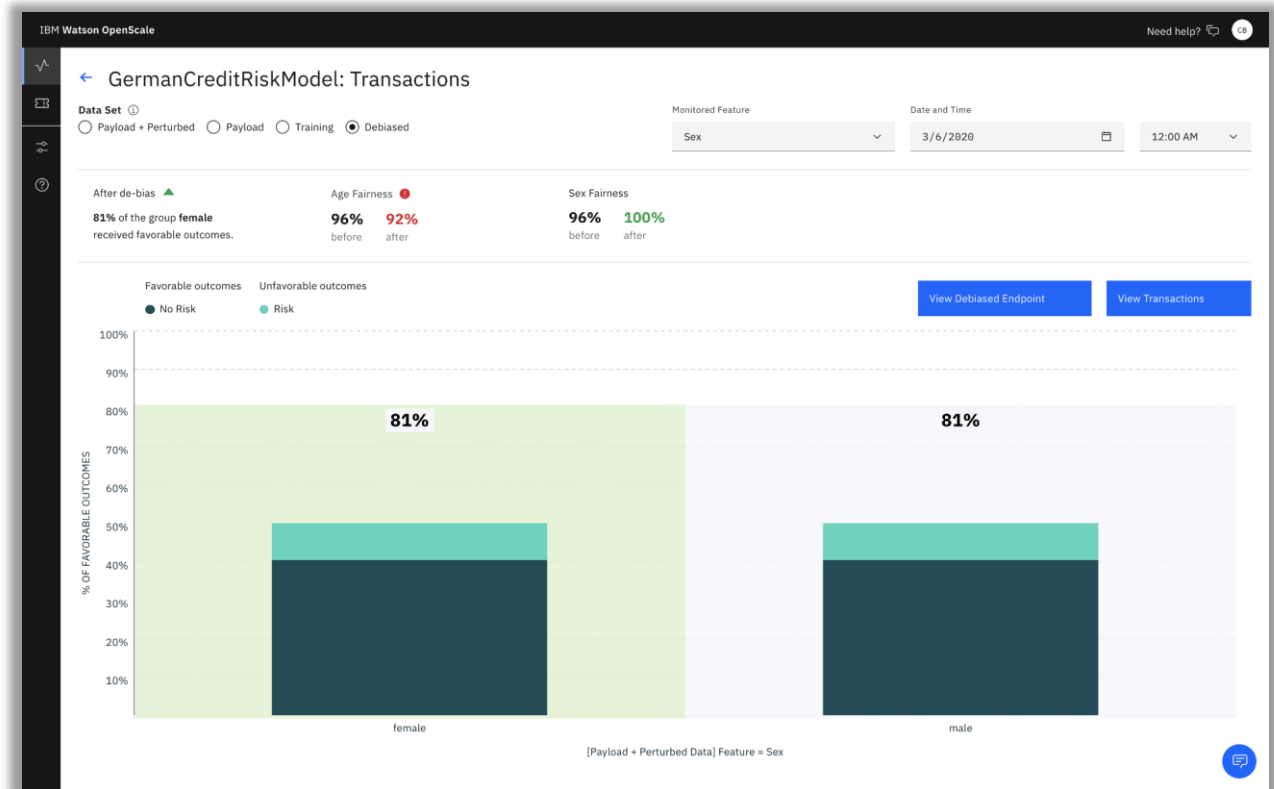
- **Credit Risk Example Model**
- **78% of the protected group (female) have a favorable output**
- **81% of the reference group (male) get a favorable output**
- **Disparate impact Value: 96%**





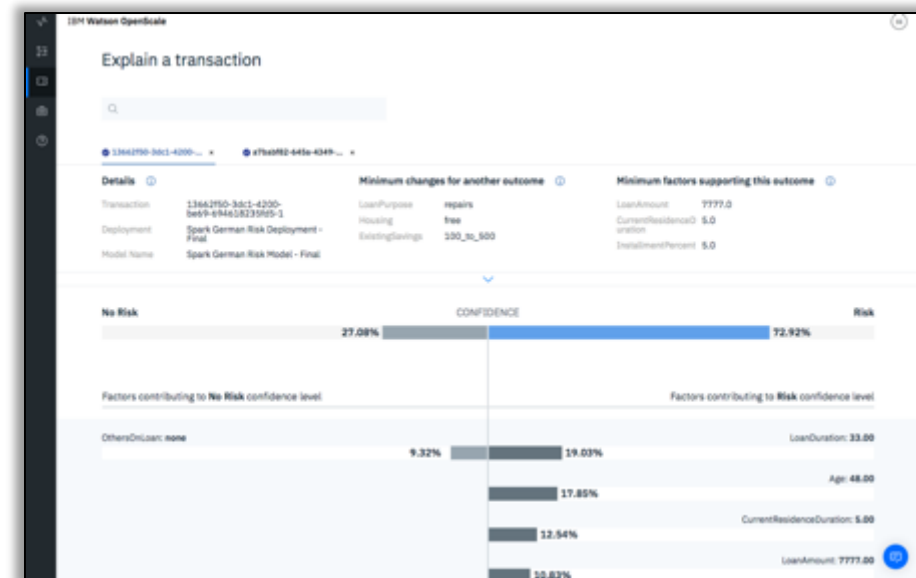
# Bias Mitigation – De-biased Model Output

- After De-biasing algorithms was applied
- Predictions are 4% more fair in this example
- 81% of the protected group (female) and of the reference group (male) get a favorable output
- Disparate impact Value: 100%



# Explainability

- OpenScale records every individual transaction and drills down into its working to explain how the model makes decisions
- It provides a simple explanation that is user friendly and interactive
- **Value:**
  - Explain individual transaction level decisions made by the model in run time, including details about most important attributes and their values in order to assist in compliance and customer care situations
  - Analyze individual transactions in a what-if manner in order to understand how model behavior will change in different business situations



# LIME and Contrastive Explanations

## ■ LIME Output:

- Set of features which played a positive role or negative role in the prediction.
- Also identifies the feature weights which helps to identify the most important or least important features

## ■ Contrastive Explanation:

- Explains the behavior of the model in the vicinity of the data point whose explanation is being generated.
- Assumption:
  - The most common value is the least interesting from an explanation point of view
  - E.g., If median salary is between \$70-90K, then someone who has a salary of \$80K it is not very “interesting” or to say it differently it is “normal”.
  - However, if someone has salary of \$200K, it is very “interesting”

# Drift Detection in OpenScale

- **Drift Monitor in OpenScale measures two types of drifts:**

**Drop in accuracy:** It estimates the drop in accuracy of the model at runtime. The model accuracy could drop if there is an increase in transactions similar to those which the model was unable to evaluate correctly in the training data.

**Drop in data consistency:** It estimates the drop in consistency of the data at runtime as compared to the characteristics of the data at training time.

A drop in model accuracy and data consistency may lead to a negative impact on the business outcomes associated with the model.

OpenScale measures the drift without requiring labelled data. Accuracy computation using labelled data can be expensive and might not be comprehensive

OpenScale does Drift detection on the entire payload data

OpenScale will automatically detect drifted transactions and pinpoint datapoints that contribute to drift

Proceed with Lab-6

Return for Presentation at  
03:45 PM EST

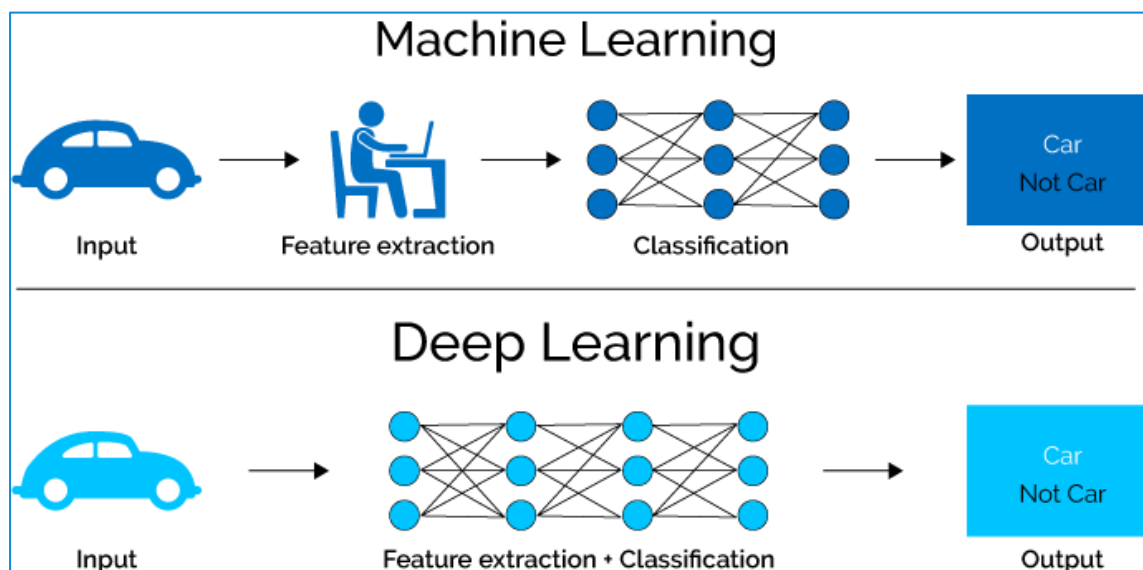
# Introduction to Machine Learning

- Overview
- Data Science Methodology
- Data Understanding
- Data Preparation
- Categories of Machine Learning
- Learning Challenges
- Machine Learning Algorithms
- Model Evaluation
- Trusted AI
- Deep Learning

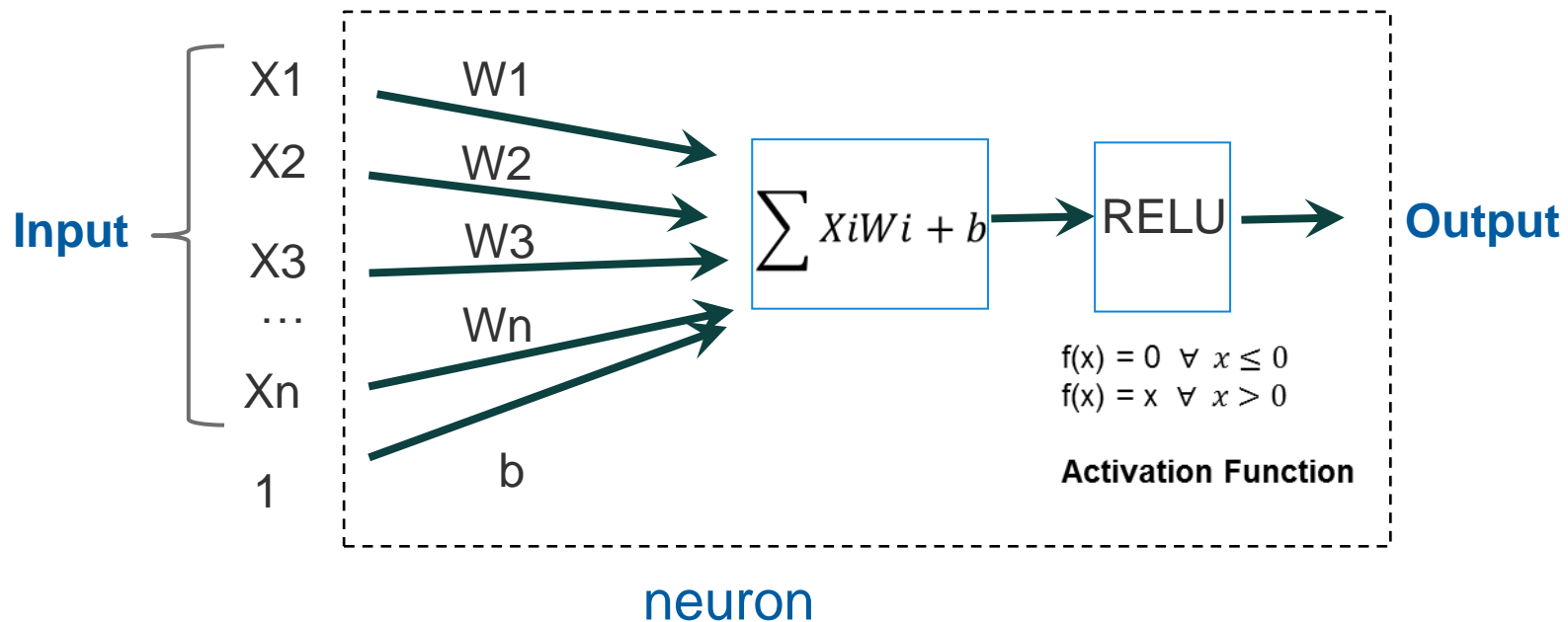
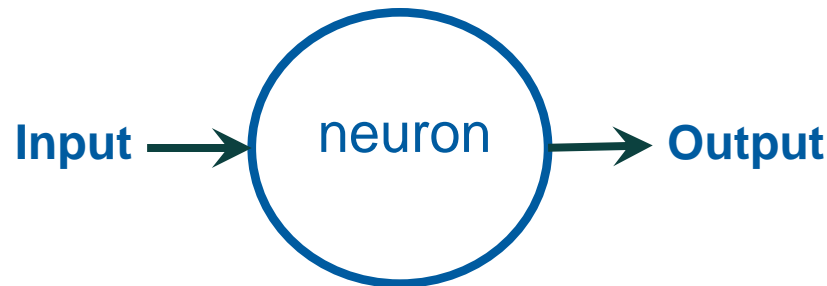


# Deep Learning

- Deep Learning is a machine learning method.
- Could be supervised or unsupervised
- Originated in 1940s
- Became very popular this decade
  - Hardware Improvements/Cost – GPUs, Storage
  - Availability of Large Datasets for Training
  - Better performing algorithms.
- Especially good for human perception type task



# What is an Artificial Neuron?



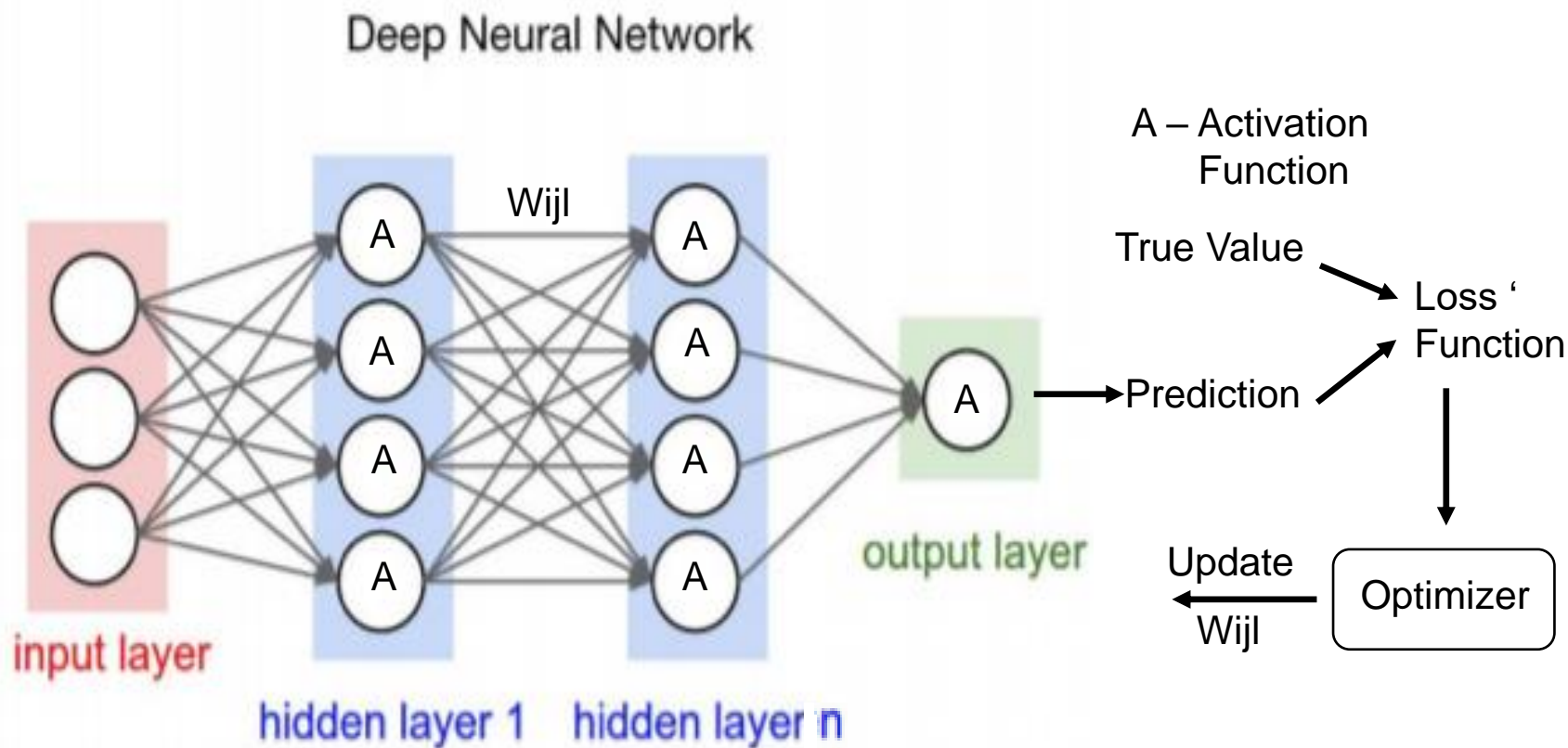


# Neural Network

Modeling

Training the AI is the hardest part of Deep Learning.

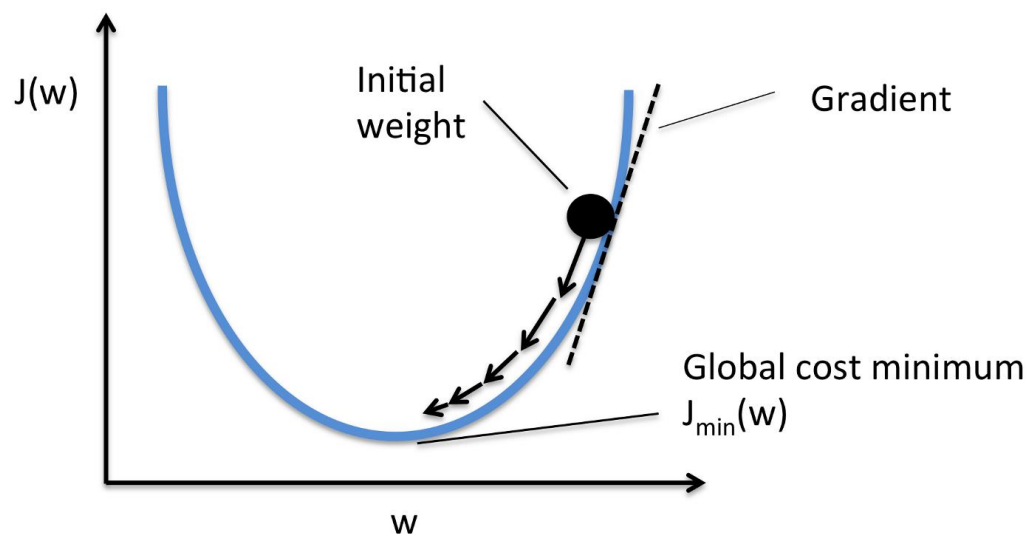
- You need a large data set.
- You need a large amount of computational power

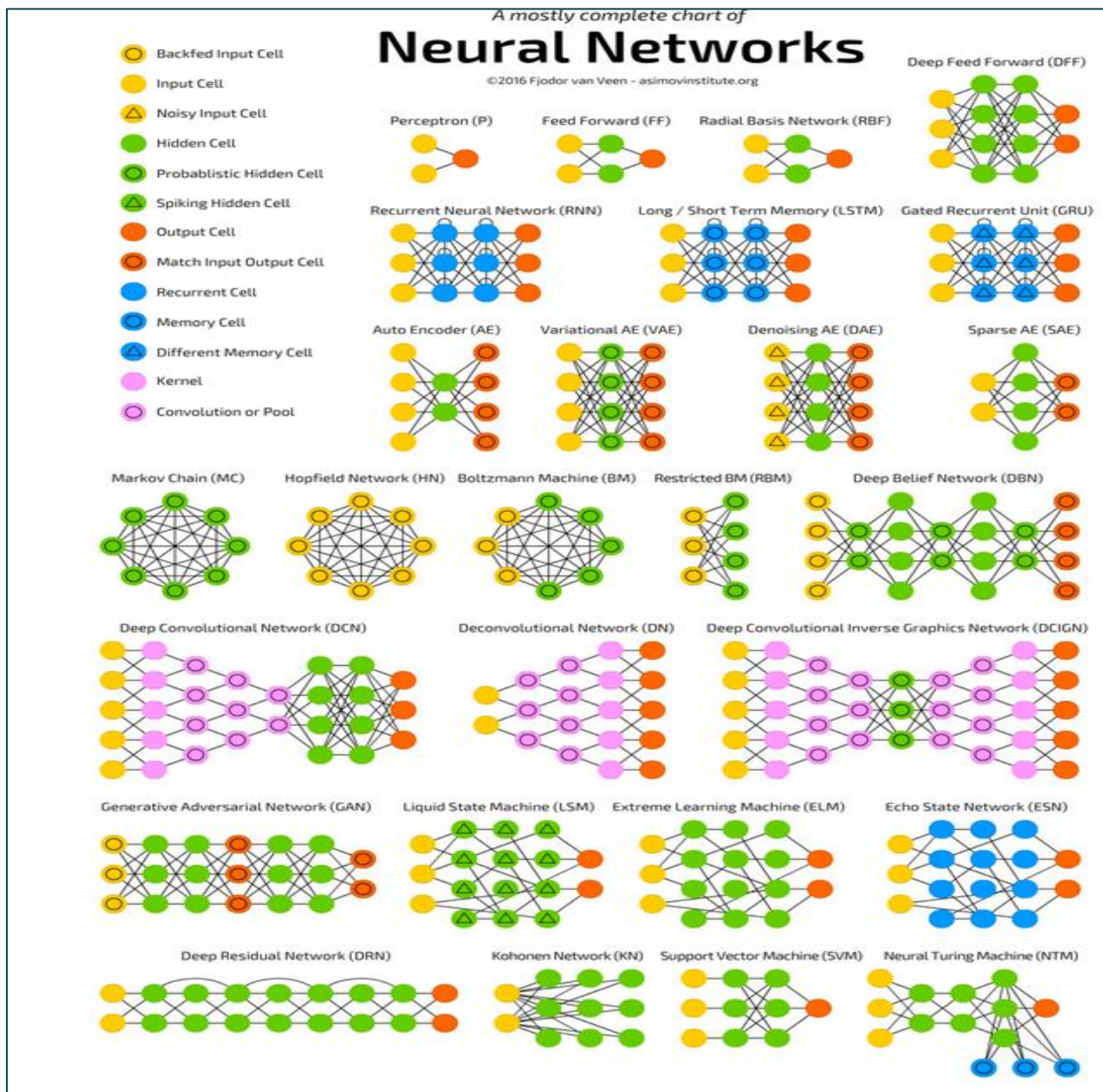


$W_{ijl}$  – weight from neuron (i) in level (l-1) to neuron (j) in level (l)

# Cost Function

- During training we need to know how our DNN is doing!
  - Compare it's predictions to dataset's output (Loss Function)
  - Based on how far it is from actual value → update weights (Optimizer)
- Ideally, we want our loss function to be zero.
  - Does not happen in real world
  - use techniques like “Gradient Descent” → allows us to find the minimum of a function by iterating through dataset and updating the weights





# Common Types of Deep Neural Networks

## Convolutional Neural Networks

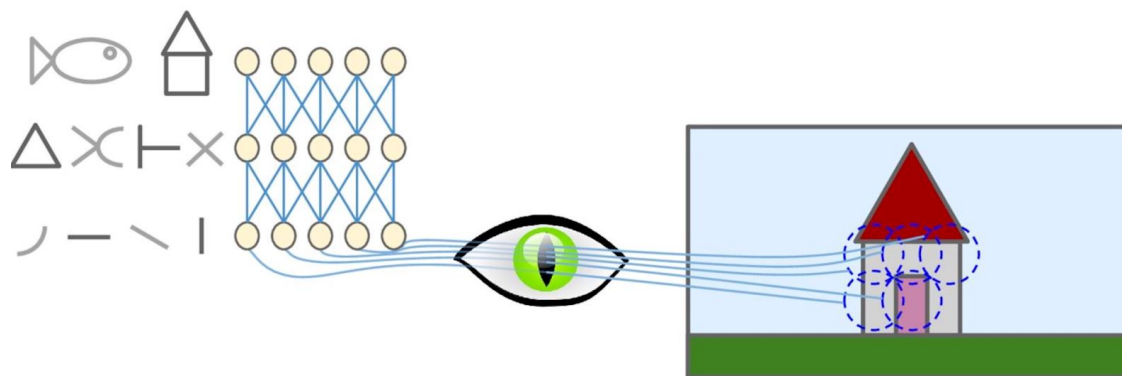
- Image classifications
- Object detection
- Image Segmentation
- Recognizing faces
- Natural language processing
- ..

## Recurrent Neural Networks

- Speech Recognition
- Handwriting Recognition
- Machine Translation
- Sequence prediction
- Natural Language Processing
- ...

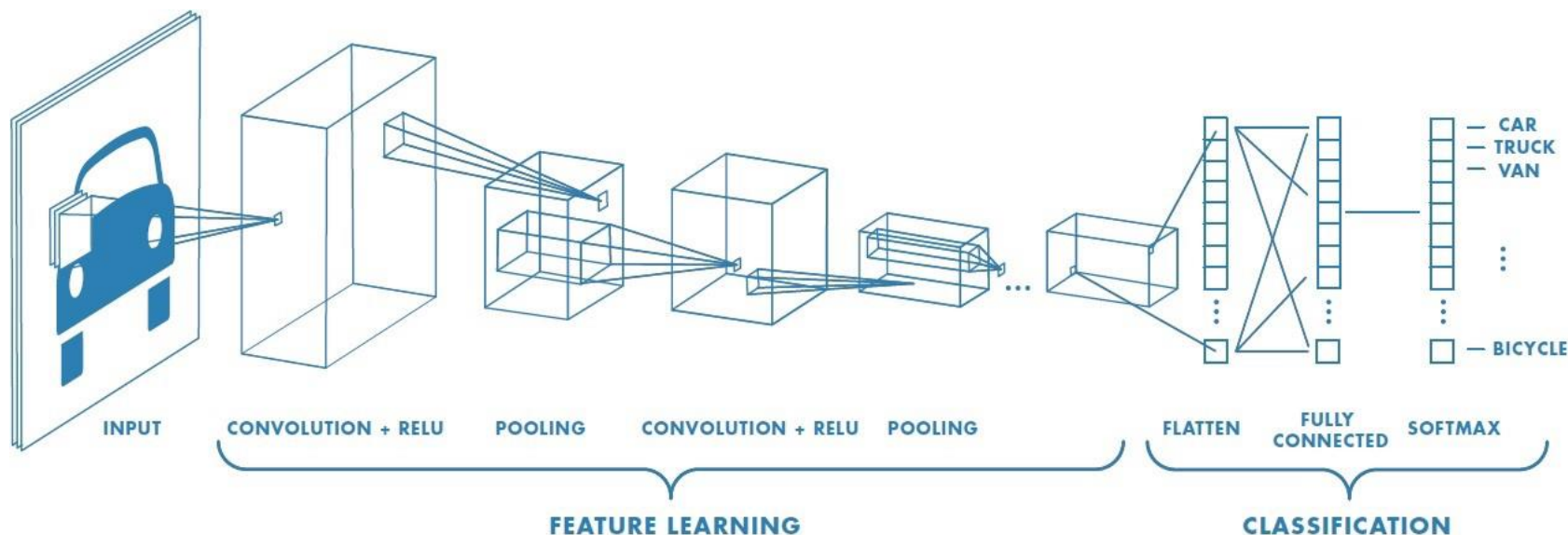
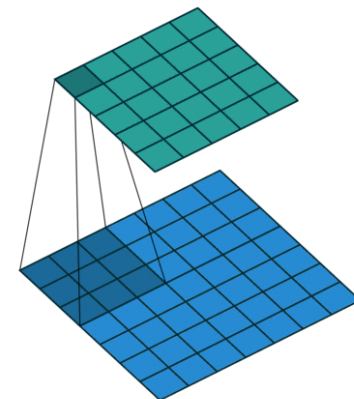
# Convolutional Neural Networks (CNN)

- Inspired by the architecture of the visual cortex
- Some neurons only react to horizontal lines, while others reacts to lines with different orientations
- Some neurons have larger receptive fields, so they react to more complex patterns based on output of lower-level neuron
- Two building blocks: **Convolutional** layers and **Pooling** layers



# Convolutional Layer

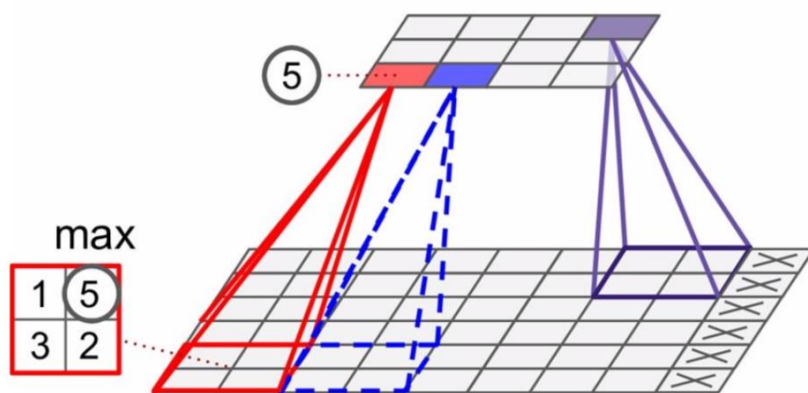
- Convolution is the first layer to extract features from an input image.
- Neurons in the first convolutional layer are not connected to every pixel in the input image, but **only to pixels in their receptive fields**
- Each neuron in the second convolutional layer is connected only to **neurons located within a small rectangle** in the first layer.
- Convolution preserves the relationship between pixels by learning image features using small squares of input data.





# Pooling Layer

- Pooling Layer reduce the number of parameters when the images are too large.
  - Max Pooling
  - Average Pooling
  - Sum Pooling



# CNN Applications

## Classification



## Retrieval

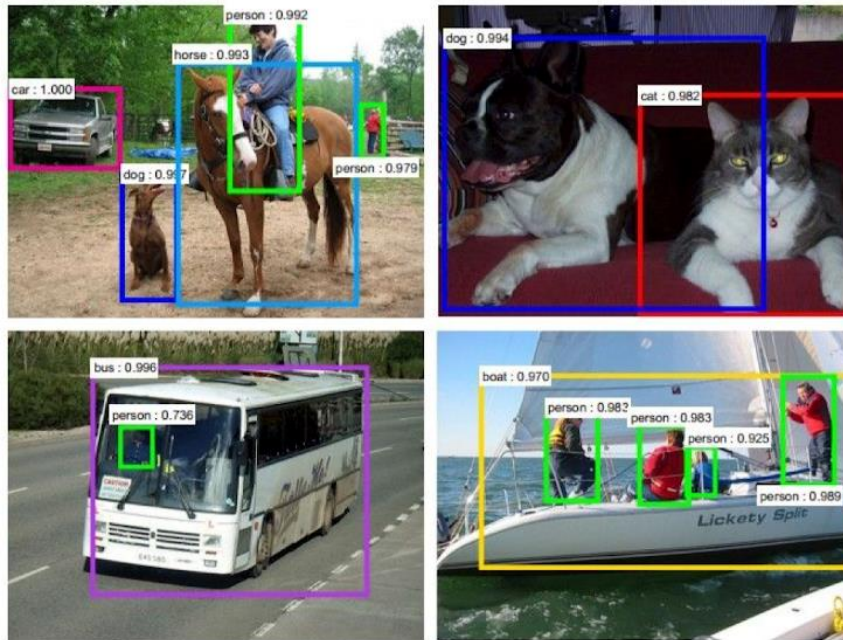


Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.



# CNN Applications

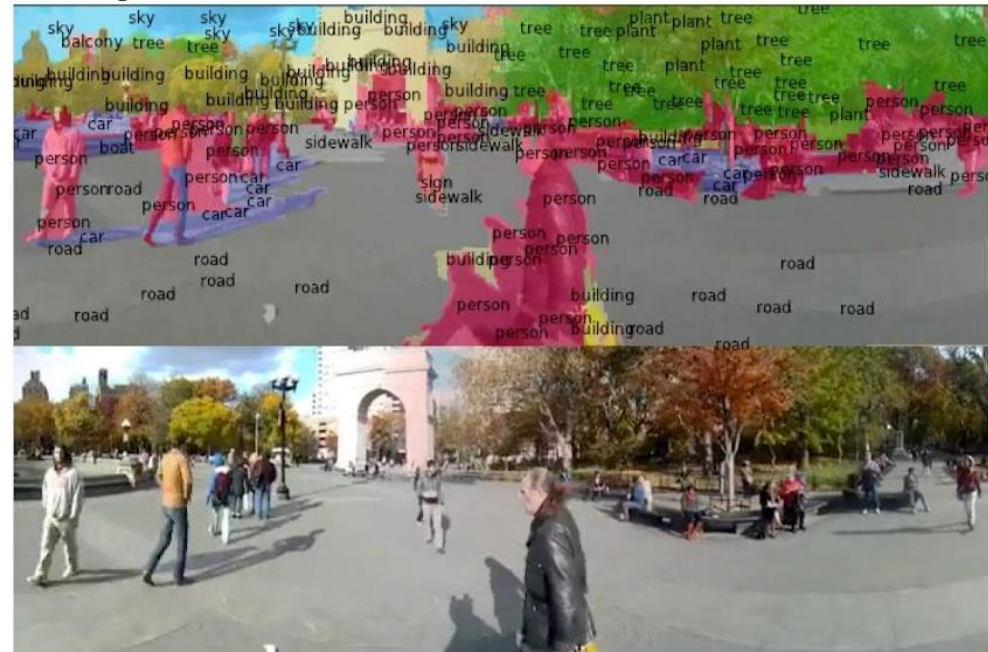
## Detection



Figures copyright Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, 2015. Reproduced with permission.

[Faster R-CNN: Ren, He, Girshick, Sun 2015]

## Segmentation

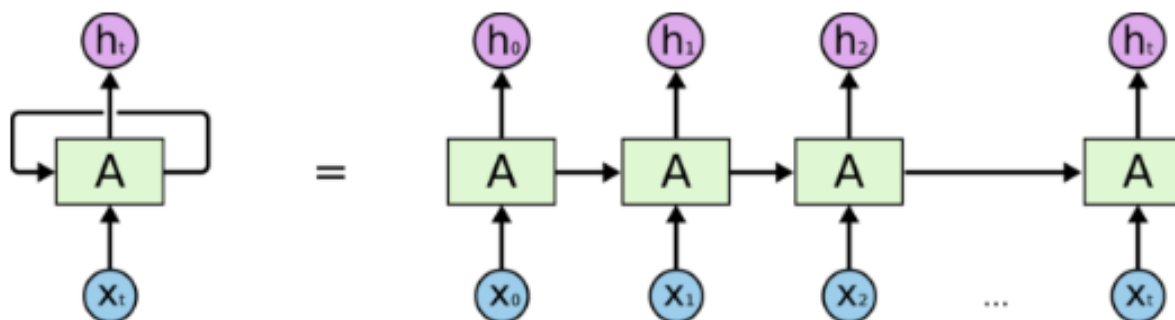


Figures copyright Clement Farabet, 2012.  
Reproduced with permission.

[Farabet et al., 2012]

# Recurrent Neural Networks (RNN)

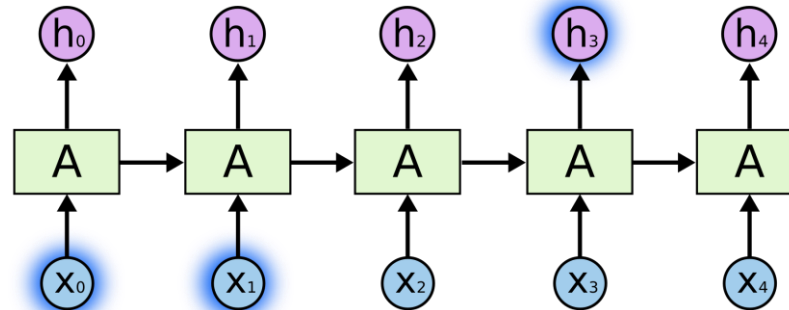
- Humans don't start their thinking from scratch every second. We rely on our memory!
- Traditional Neural Networks **CAN NOT** help → data flows forward only
- Recurrent Neural Networks address this issue.
  - They are networks with loops in them, allowing information to persist.
- RNN Applications are: Speech recognition, Language modeling, Translation



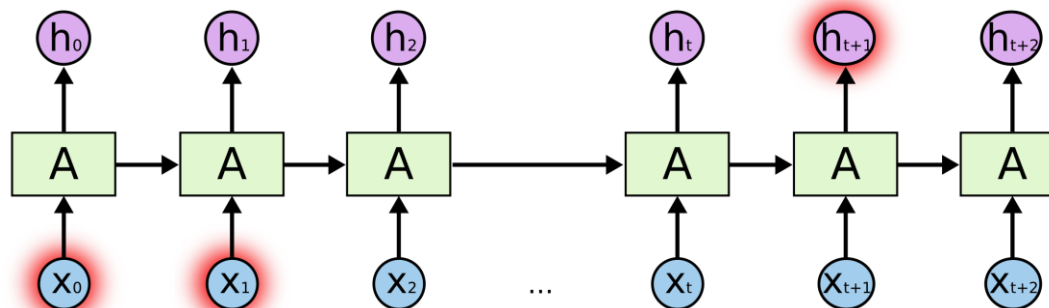
An unrolled recurrent neural network.

# Long-Term Dependencies Problem

- RNNs can look at old information. But how old?
- Successfully uses recent information → Clouds are in the ... [Sky]

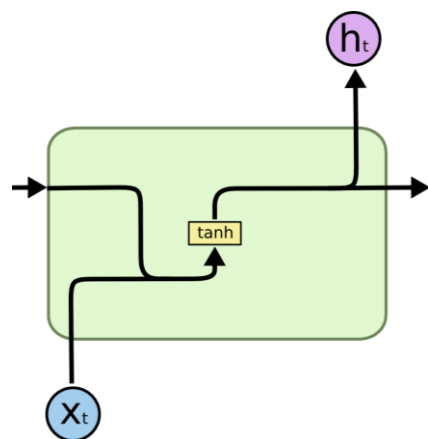


- Can NOT use older information → I grew up in France, in a small city near Paris, so I speak fluent ... [??] **But why?**

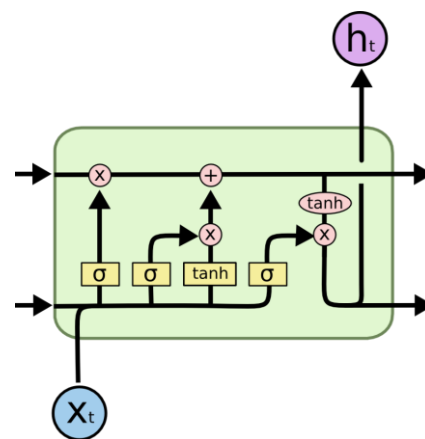


# Long-Term Dependencies Problem

- **Vanishing gradients**
  - $0 < \text{Gradient values} < 1 \rightarrow$  they leave the connection weights **unchanged**
- **Exploding gradients**
  - Calculated gradients are large values  $\rightarrow$  many layers got **insanely large** weight updates, and the network becomes unstable and diverged
- **Long Short Term Memory (LSTM) fixed the gradient problem**
  - by introducing a few more gates that control access to the cell state



The repeating module in RNN contains a single layer.



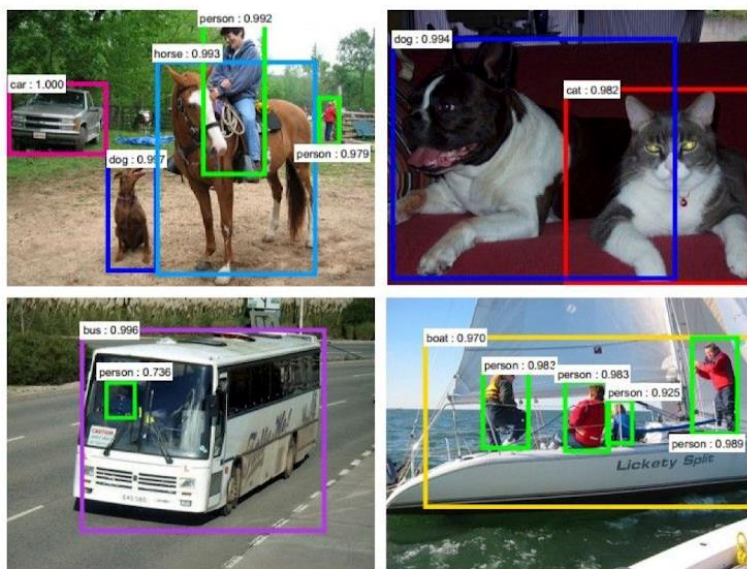
The repeating module in LSTM contains four interacting layers



- We use DNNs for lots of things:

- Facial recognition → iPhone FaceID
- Text recognition → Mobile Check Deposit
- Self driving cars → help detect signs, pedestrians, traffic lights, etc.

## Detection



Figures copyright Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, 2015. Reproduced with permission.

[Faster R-CNN: Ren, He, Girshick, Sun 2015]

## Segmentation

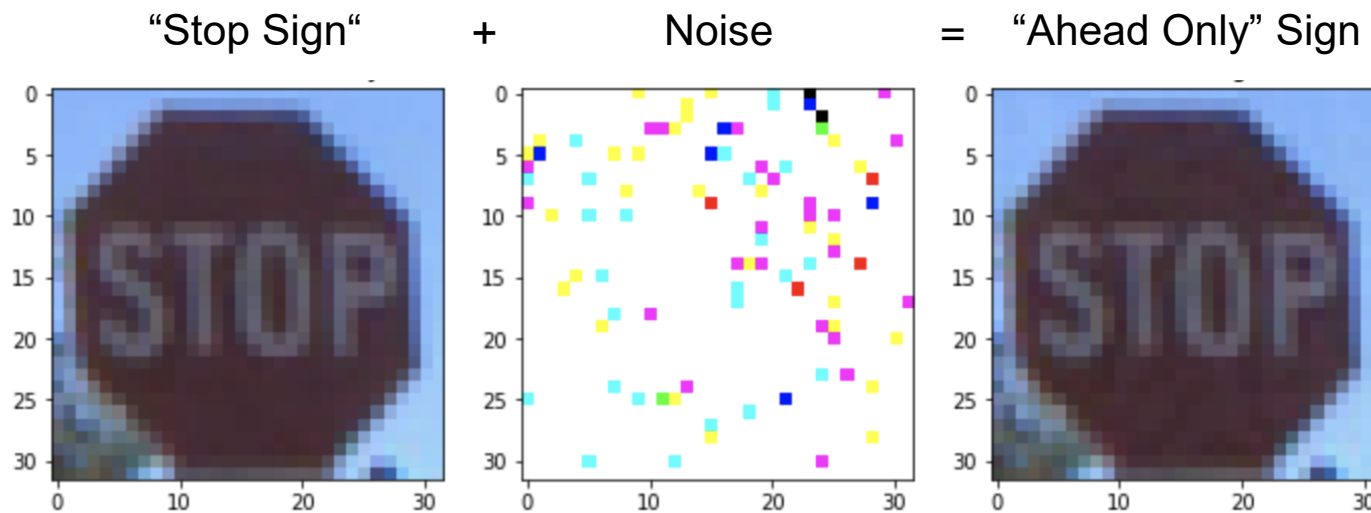


Figures copyright Clement Farabet, 2012.  
Reproduced with permission.

[Farabet et al., 2012]

# What are Adversarial Images?

- Adversarial examples are inputs (say, images) which have deliberately been modified to produce a desired response by a DNN.
- Often, the target of adversarial examples is **misclassification** or a **specific incorrect prediction** which would benefit an attacker.



# Threat Model

## Black Box

- Attackers can only observe the outputs of a model. E.g. Attacking a model via an API
  - The adversary has no knowledge of the training algorithm or hyperparameters.
- Examples:
  - Boundary Attack
  - Substitute Blackbox Attack
  - Etc.

## White Box

- attackers have complete access to the model that they want to attack.
- These are most effective attacks
- Examples:
  - Fast Gradient Sign Method (FGSM)
  - Random + FGSM
  - Projected Gradient Descent
  - Etc.

## Why are they dangerous?

- **Can be crafted even if the attacker doesn't have exact knowledge of the architecture of the DNN**
- **Adversarial attacks can be launched in the physical world**
  - adversaries could evade face recognition systems by wearing specially designed glasses
  - defeat visual recognition systems in autonomous vehicles by sticking patches to traffic signs

Subtle Poster



Camouflage Sticker













\* Pictures from paper: Kevin Eykholt, et al. "Robust Physical-World Attacks on Deep Learning Visual Classification"



# Are they effective?

- Researchers proved that these attacks are successful! [1]

Perturbation	Attack Success	A Subset of Sampled Frames $k = 10$				
Subtle poster	100%					
Camouflage abstract art	84.8%					

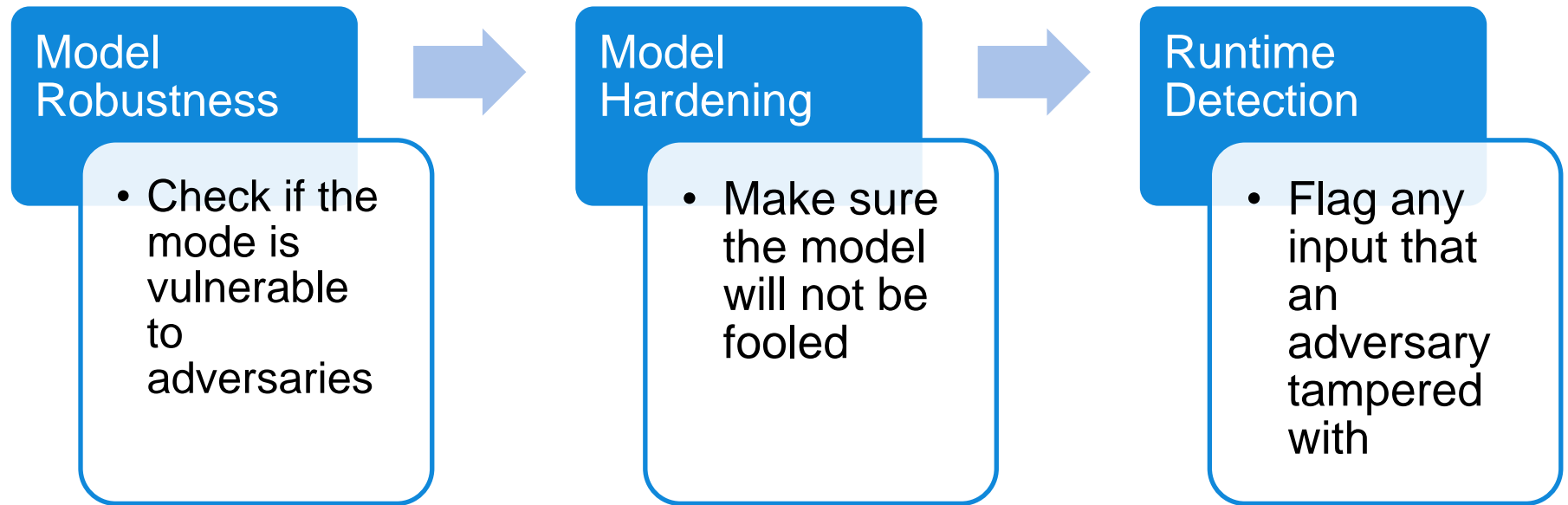
[1] Kevin Eykholt, et al. "Robust Physical-World Attacks on Deep Learning Visual Classification"

# Adversarial Robustness Toolbox (ART)

- IBM Research team developed the toolkit to help defend DNNs against adversarial attacks
- Open-source software library
- Written in python
- Supports most deep learning frameworks : TensorFlow, Keras, PyTorch, etc.
- It creates adversarial examples AND provides methods for defending DNNs against those.



# How can ART help?



# Conclusions

- **Adversarial attacks are real threats**
  - Self-driving cars
  - Healthcare
  - Financial institutions
  - Insurance companies
  - ...
- **It's important to**
  - Realize there are vulnerabilities
  - Have means to protect ourselves

Proceed with Lab 7 and Lab 8

Return for Presentation at  
05:30 PM EST

## Some items to think about ....

### ▪ **Business**

- What are your goals?
- What are the criteria for success?
- How are you going to measure it?

### ▪ **Data**

- Do you need labeled (\$\$) data?
- What is the quality of your data?
- What features are pertinent?
- Do you have enough data?
- How are you going to obtain the data?

### ▪ **Models**

- What algorithms to use?
- What metrics to evaluate the algorithms?

### ▪ **Implementation**

- What tooling will you use?
- Do you need to scale?
- What resources do you have? Memory?, GPUs?, Compute?
- How are you going to get feedback?