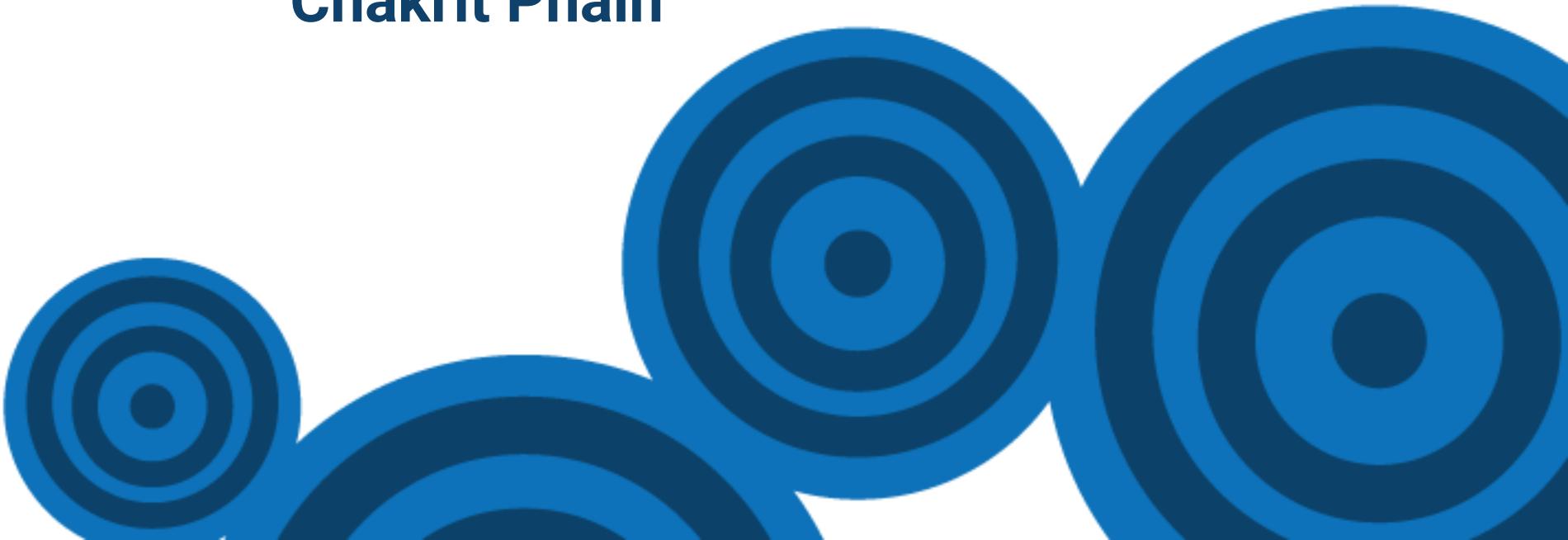




Anomaly Detection

Softnix Technology
Chakrit Phain





Agenda Day 2

- Anomaly Detection and Supervised Learning
- Neural Network & Deep Learning
- Anomaly Detection with LSTM & GRU
- Anomaly Detection with Autoencoder
- Anomaly Detection Workshop



Anomaly Detection

Day 2

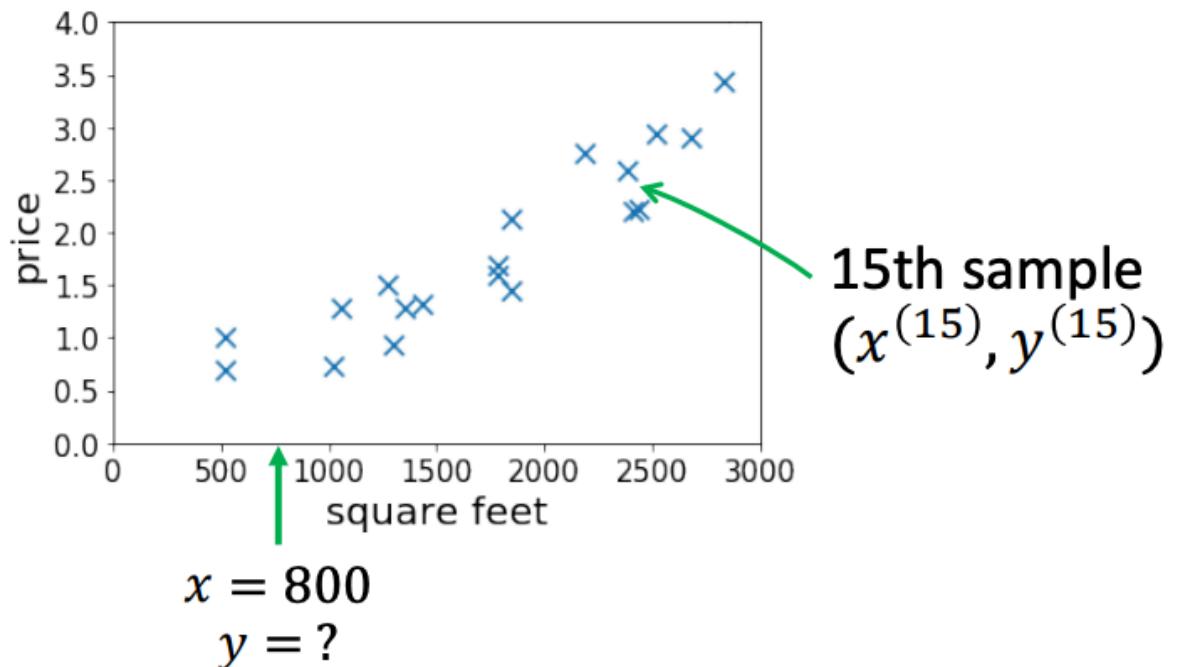




Anomaly Detection and Supervised Learning



- Given: a dataset that contains n samples
 $(x^{(1)}, y^{(1)}), \dots (x^{(n)}, y^{(n)})$
- Task: if a residence has x square feet, predict its price?



Housing Price Prediction

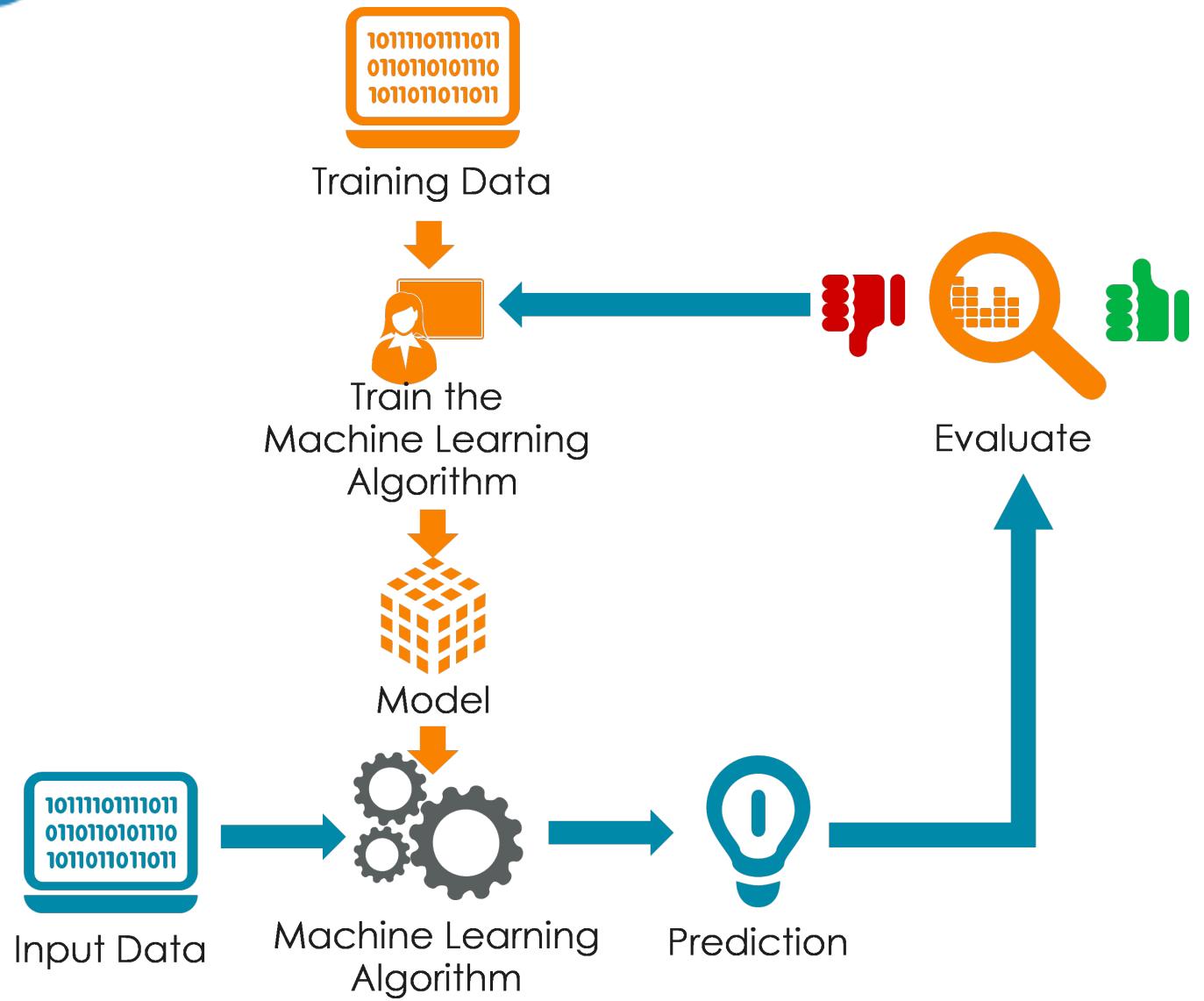
- $x \in \mathbb{R}^d$ for large d
- E.g.,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ x_d \end{bmatrix} \begin{array}{l} \text{--- living size} \\ \text{--- lot size} \\ \text{--- \# floors} \\ \text{--- condition} \\ \text{--- zip code} \\ \vdots \end{array} \longrightarrow y \text{ --- price}$$





Supervised Learning

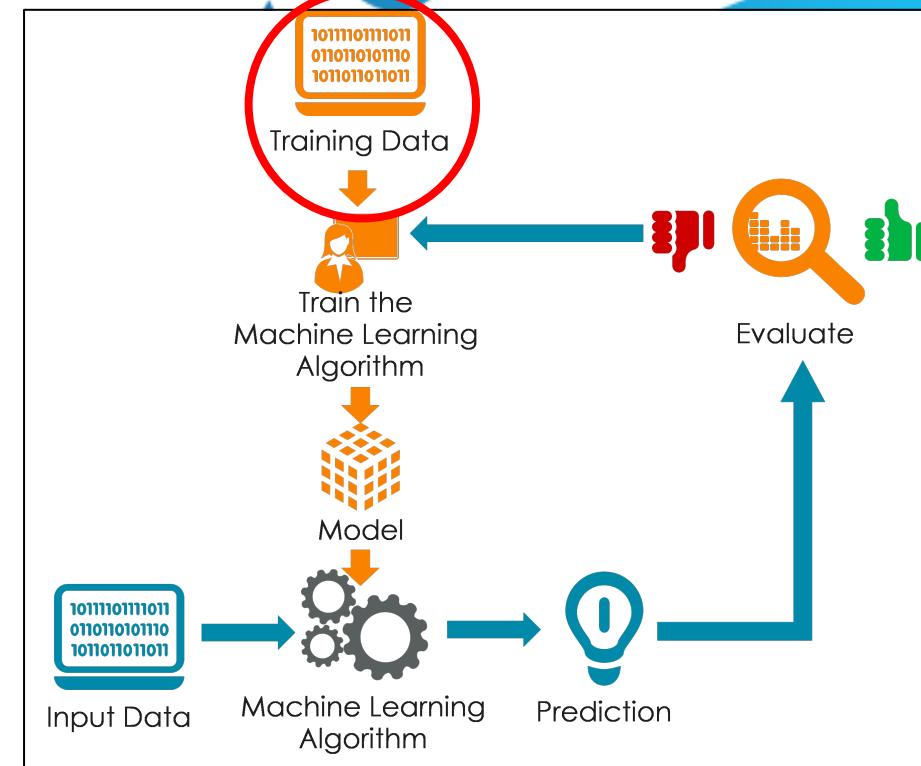


Supervised Learning

Dimension,Factor

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...
57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

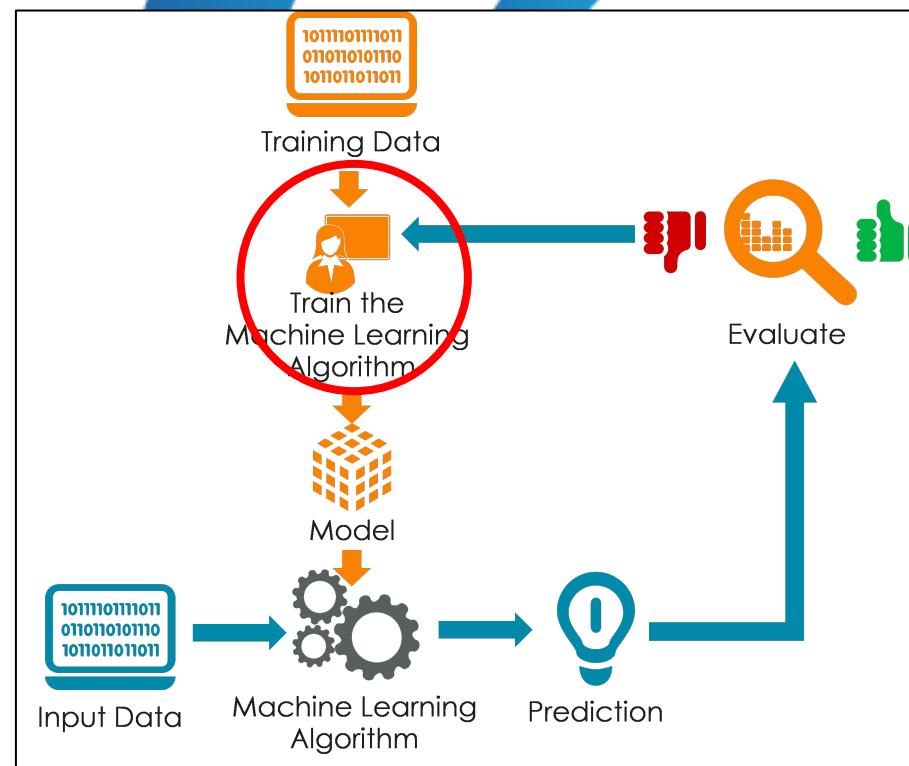
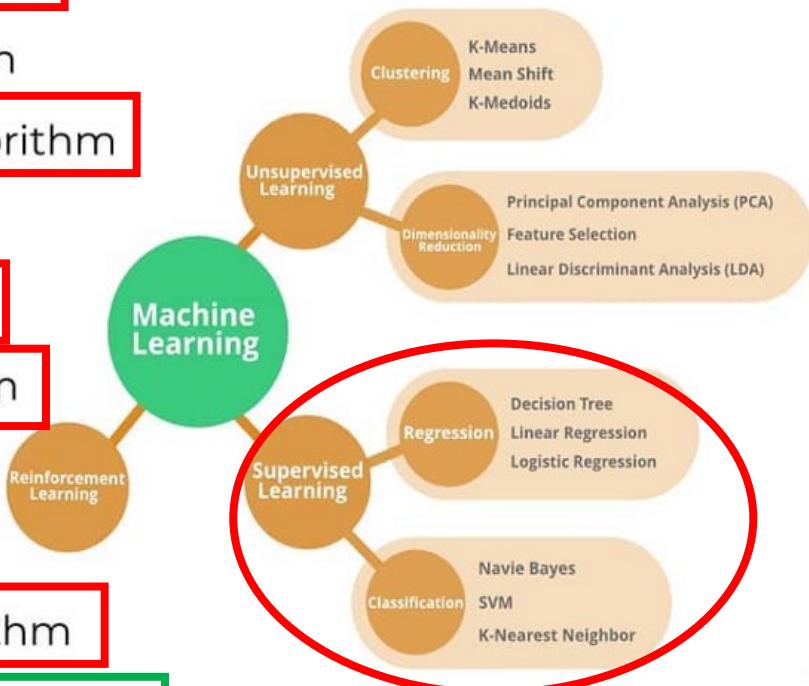
Label



Supervised Learning

Top 10 Algorithms every Machine Learning Engineer should know

- 1. Naïve Bayes Classifier Algorithm**
- 2. K Means Clustering Algorithm**
- 3. Support Vector Machine Algorithm**
- 4. Apriori Algorithm**
- 5. Linear Regression Algorithm**
- 6. Logistic Regression Algorithm**
- 7. Decision Trees Algorithm**
- 8. Random Forests Algorithm**
- 9. K Nearest Neighbours Algorithm**
- 10. Artificial Neural Networks Algorithm**

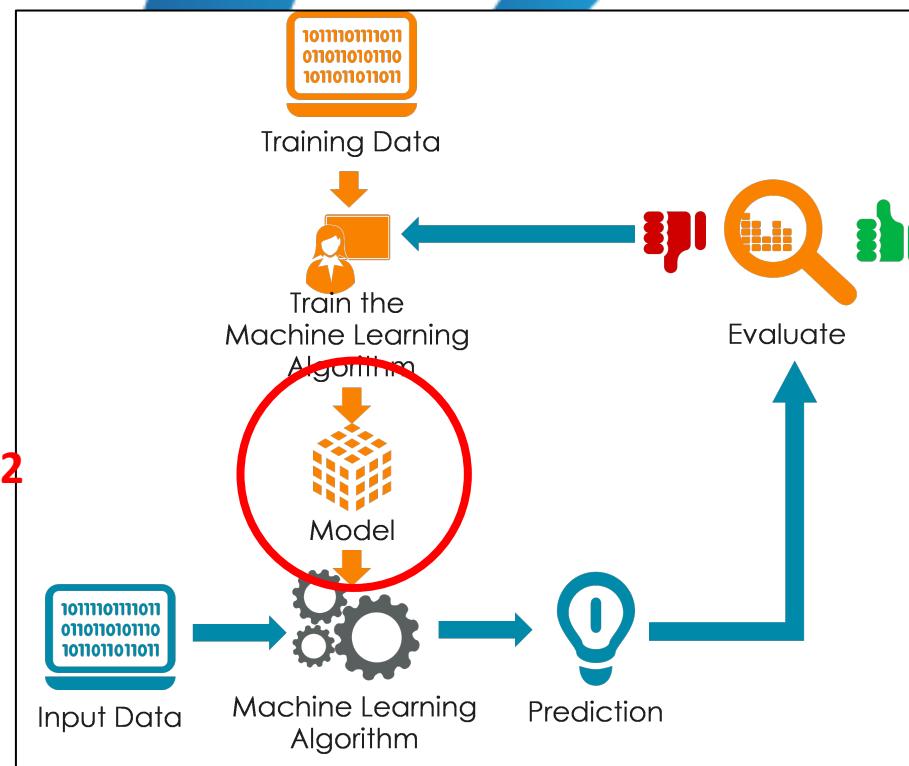
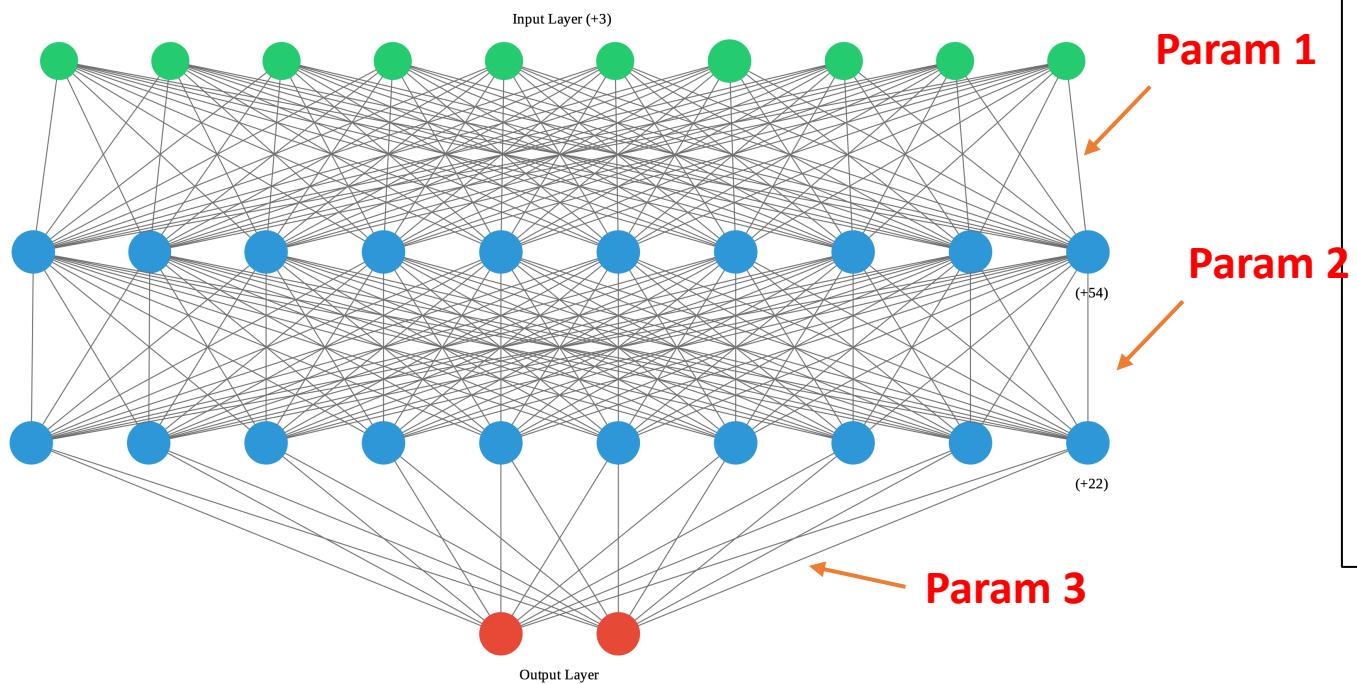




Supervised Learning

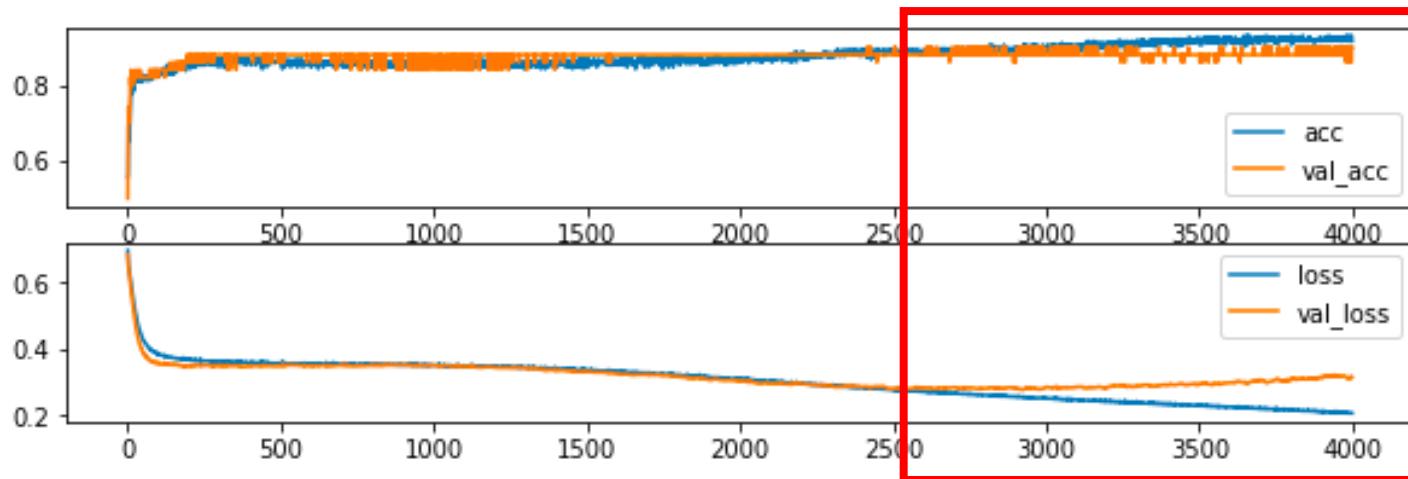
Model: "sequential_9"

Layer (type)	Output Shape	Param #
dense_20 (Dense)	(None, 64)	896
dense_21 (Dense)	(None, 32)	2080
dense_22 (Dense)	(None, 2)	66
Total params:	3,042	
Trainable params:	3,042	
Non-trainable params:	0	

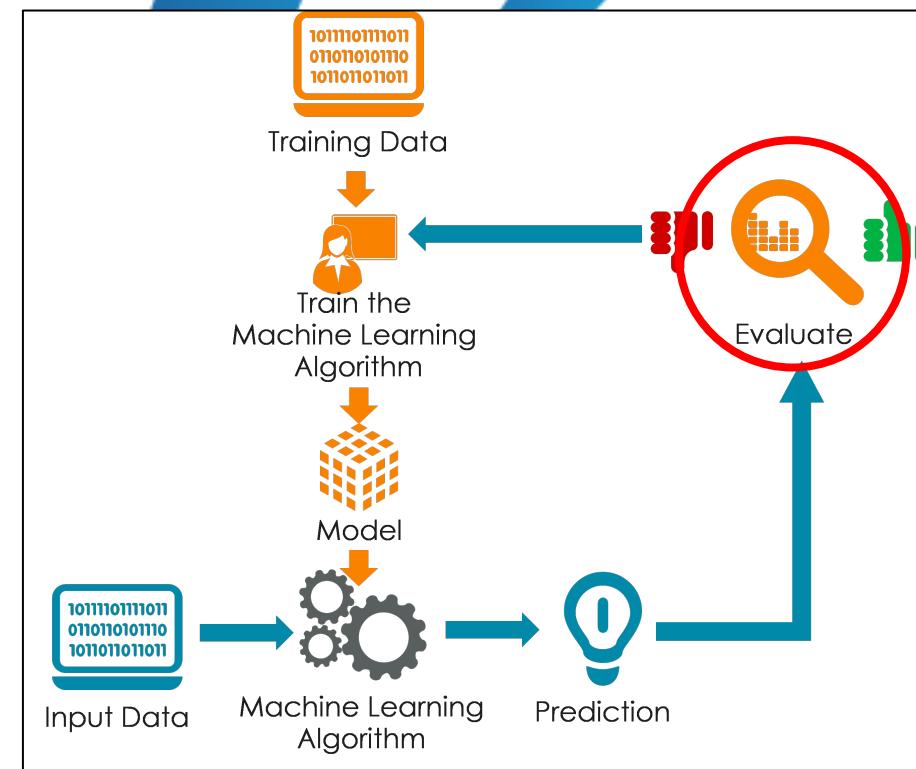




Supervised Learning

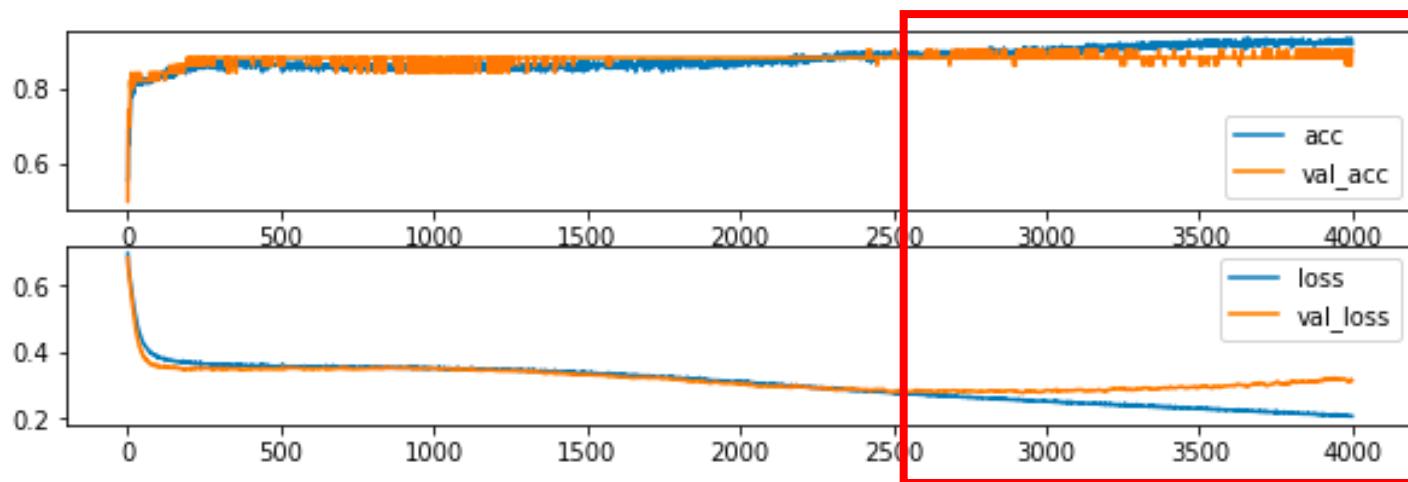


	precision	recall	f1-score	support
0	0.90	0.78	0.84	23
1	0.83	0.93	0.88	27
micro avg	0.86	0.86	0.86	50
macro avg	0.87	0.85	0.86	50
weighted avg	0.86	0.86	0.86	50
samples avg	0.86	0.86	0.86	50

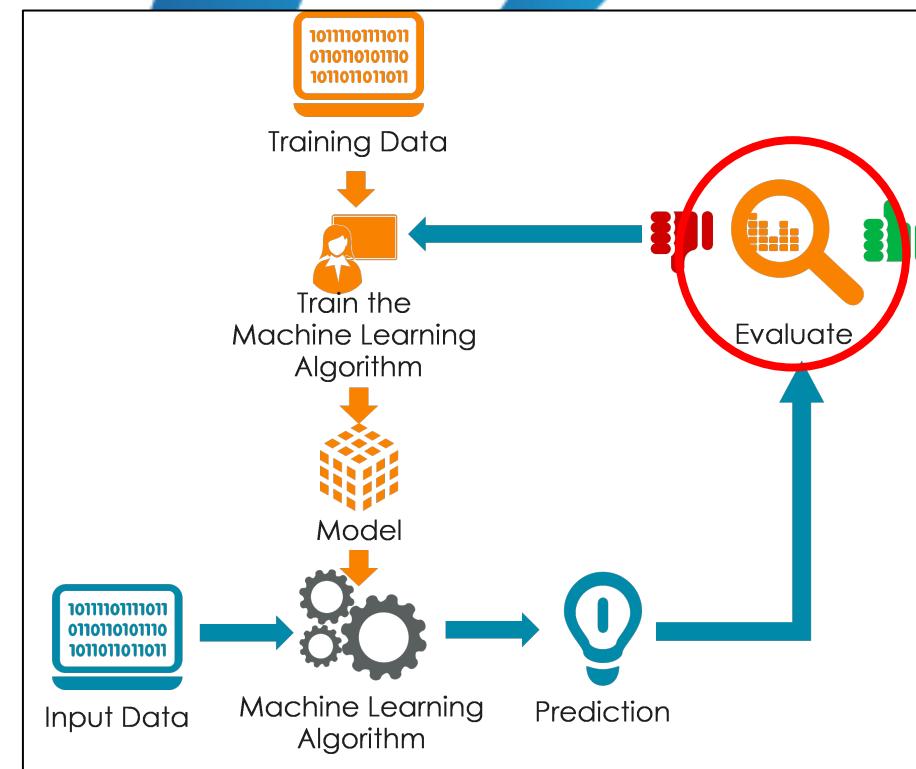




Supervised Learning



	precision	recall	f1-score	support
0	0.90	0.78	0.84	23
1	0.83	0.93	0.88	27
micro avg	0.86	0.86	0.86	50
macro avg	0.87	0.85	0.86	50
weighted avg	0.86	0.86	0.86	50
samples avg	0.86	0.86	0.86	50





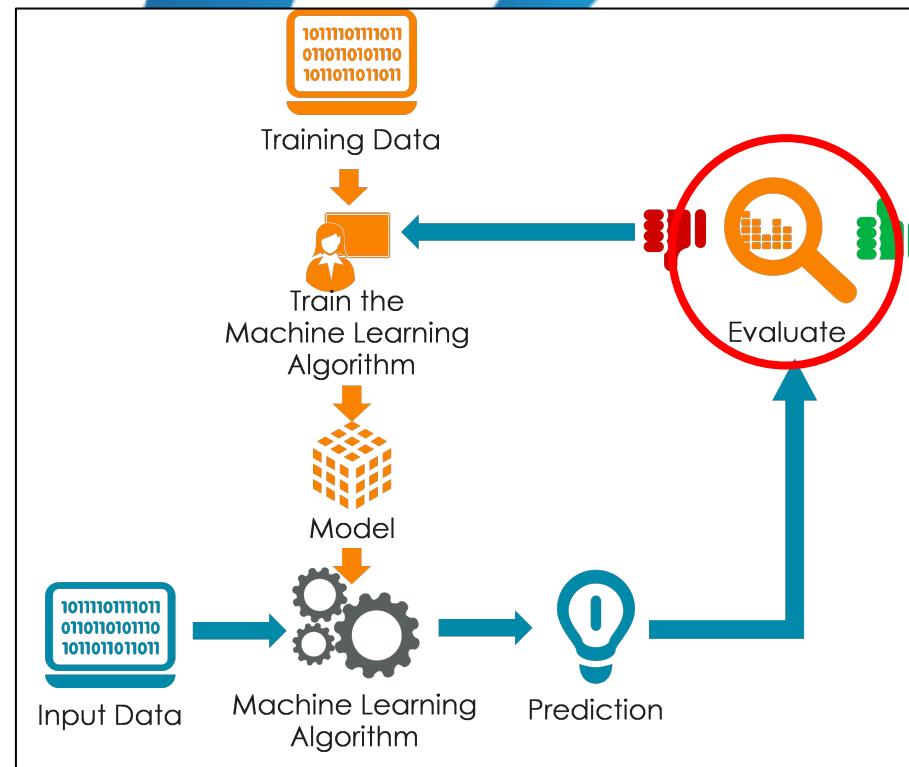
Performance metrics



Accuracy

ຕົວລະຄອບໃນມັກກຮຍກ	ຕອບ	ຈົງ	ດຸກ
ເອີ້ກ້ວຍ	Yes	Yes	True
ວຶ້ງຢັ້ງ	Yes	Yes	True
ເສີວເລັ່ງນິ້ນ	Yes	Yes	True
ຈົວປັບກົງ	Yes	Yes	True
ກໍວຍເຈັ່ງ	Yes	Yes	True
ແອຣ໌ ພອຕາຕອර໌	Yes	No	False

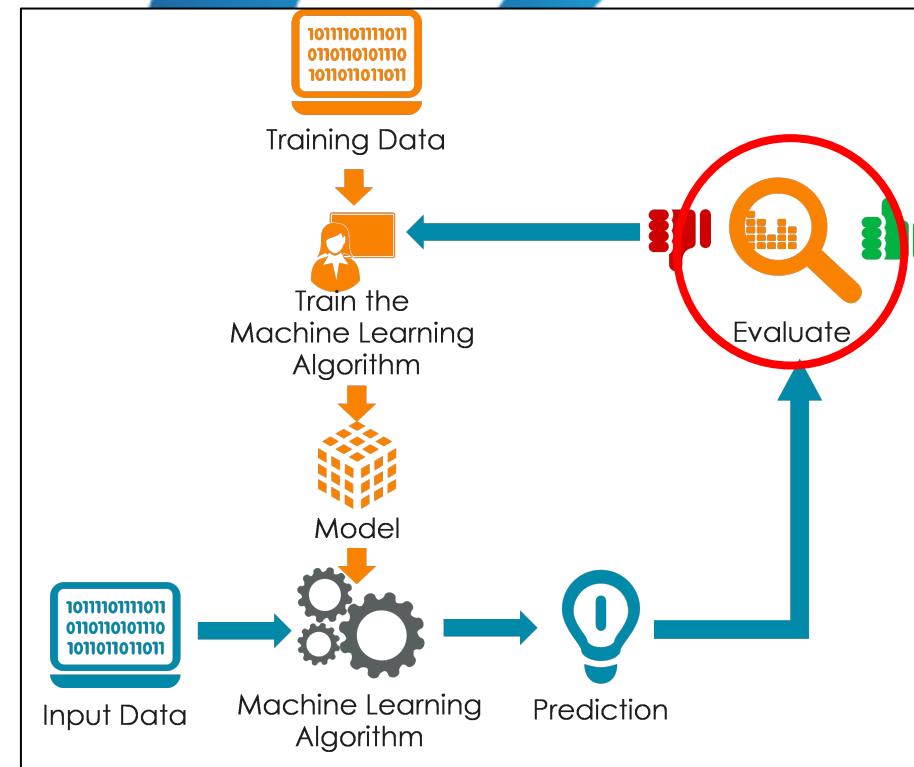
$$\text{Accuracy} = 5/6 = 0.833$$



Accuracy

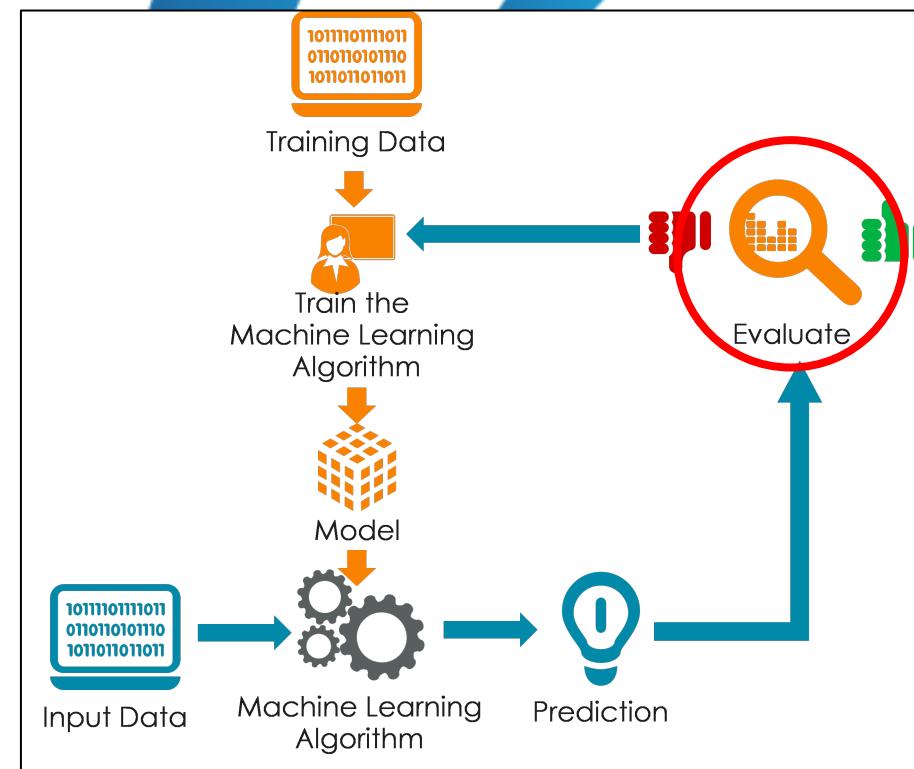
იკრთიდი კოვიდ-19	თობ	ჯრებ	გუკ
ერეკვა	ჟრებ	ჟრებ	True
ვარდა	ჟრებ	ჟრებ	True
შეივალებული	ჟრებ	ჟრებ	True
ვარდა	ჟრებ	ჟრებ	True
გარეული	ჟრებ	ჟრებ	True
ვარდა	ჟრებ	ჟრებ	False

Accuracy = 5/6 = 0.833



Confusion Matrix

		Predicted Class		Sensitivity $\frac{TP}{(TP + FN)}$
		Positive	Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	Specificity $\frac{TN}{(TN + FP)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	
	Precision $\frac{TP}{(TP + FP)}$	Negative Predictive Value $\frac{TN}{(TN + FN)}$		Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$



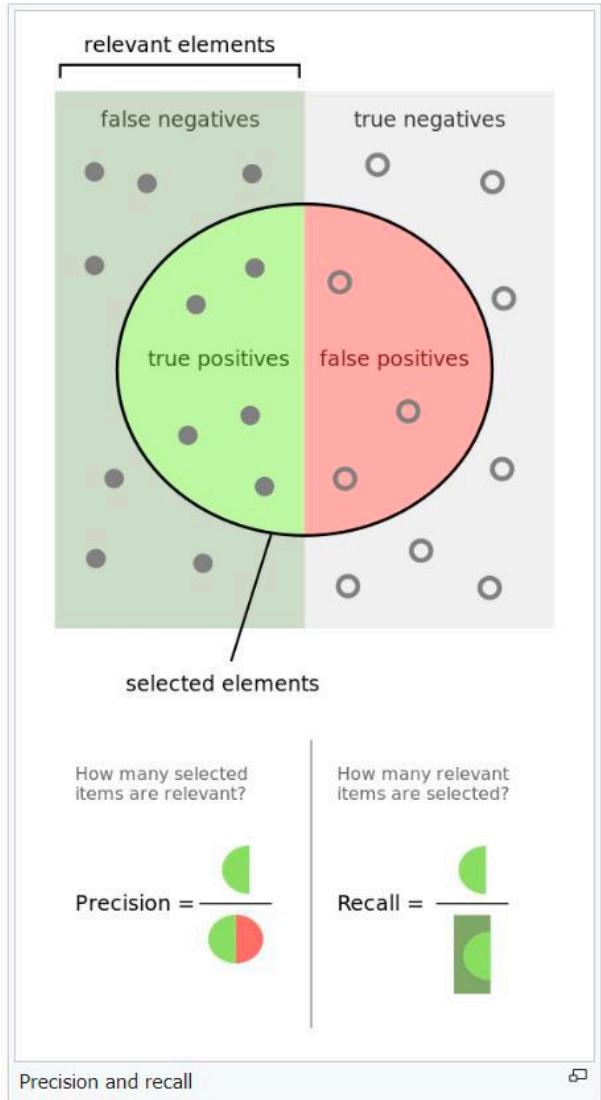
Accuracy

იკრთიდოვანი	თობ	ჯრები	გუკი
იმუნუსი	ჟრები	ჟრები	True
ვარდი	ჟრები	ჟრები	True
ვარდი	ჟრები	ჟრები	True
ვარდი	ჟრები	ჟრები	True
ვარდი	ჟრები	ჟრები	True
ვარდი	ჟრები	ჟრები	False

Actual Class	Predicted Class		Sensitivity $\frac{TP}{(TP + FN)}$	Specificity $\frac{TN}{(TN + FP)}$	Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$
	Positive	Negative			
Positive	True Positive (TP)	False Negative (FN) Type II Error			
Negative	False Positive (FP) Type I Error	True Negative (TN)			
	Precision $\frac{TP}{(TP + FP)}$	Negative Predictive Value $\frac{TN}{(TN + FN)}$			
	5	0			
	1	0			

$$\text{Accuracy} = \frac{5}{6} = 0.833$$

Precision & Recall



Precision & Recall

ไครติคโควิด-19	ตอบ	จริง	ถูก
เอี้ยก้วย	ไม่ติด	ไม่ติด	True
อึ้งยัง	ไม่ติด	ไม่ติด	True
เซียวเล่นนี่ง	ไม่ติด	ไม่ติด	True
จิวแปะกง	ไม่ติด	ไม่ติด	True
กวยเจঁ	ไม่ติด	ไม่ติด	True
แօร্জ	ไม่ติด	ติด	False

	Precision	Recall
ไม่ติด	$5/6 = 0.833$	$5/5 = 1$
ติด	$0/1 = 0$	$0/1 = 0$

Precision & Recall

ไครติดโควิด-19	ตอบ	จริง	ถูก
เอี้ยก้วย	ไม่ติด	ไม่ติด	True
อึ้งยัง	ไม่ติด	ไม่ติด	True
เซียวเล่นนี่	ไม่ติด	ไม่ติด	True
จิวแปะกง	ไม่ติด	ไม่ติด	True
กวยเจঁ	ติด	ไม่ติด	False
แօร্স	ติด	ติด	True

	Precision	Recall
ไม่ติด	$4/4 = 1$	$4/5 = 0.8$
ติด	$1/2 = 0.5$	$1/1 = 1$



Supervised Learning

1066 lines (1066 sloc) | 90.3 KB

[Open in Colab](#)

Data set <https://www.kaggle.com/ronitf/heart-disease-uci?select=heart.csv>

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow.keras as keras
import numpy as np

/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
    import pandas.util.testing as tm
```

```
In [2]: df = pd.read_csv("https://raw.githubusercontent.com/krmonline/AnomalyDetection/master/datasets_33180_43520_heart.csv")
df
```

Out[2]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

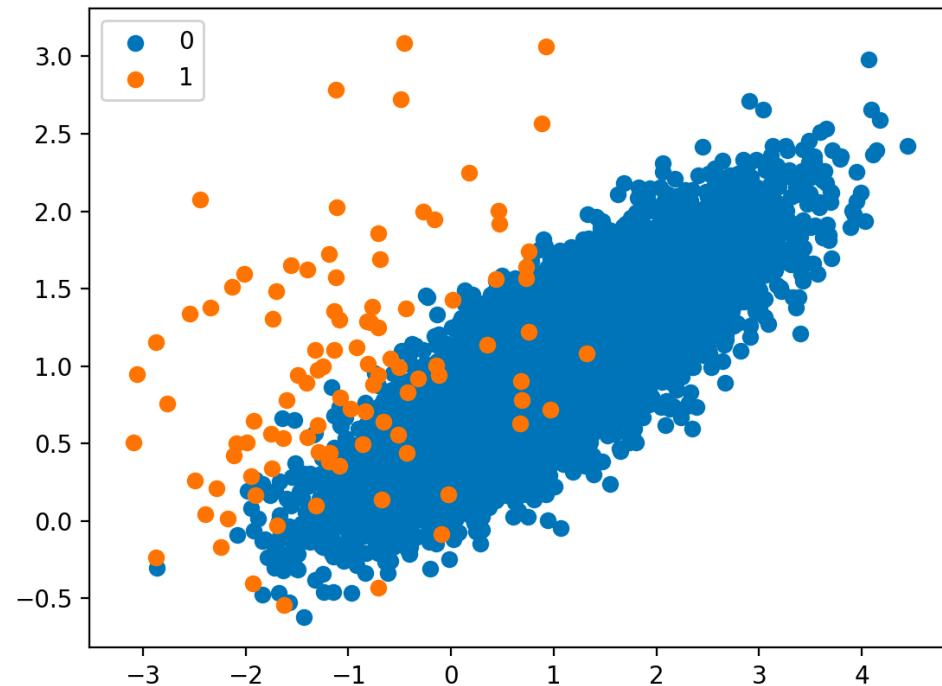
303 rows × 14 columns

```
In [3]: from sklearn.preprocessing import minmax_scale
```

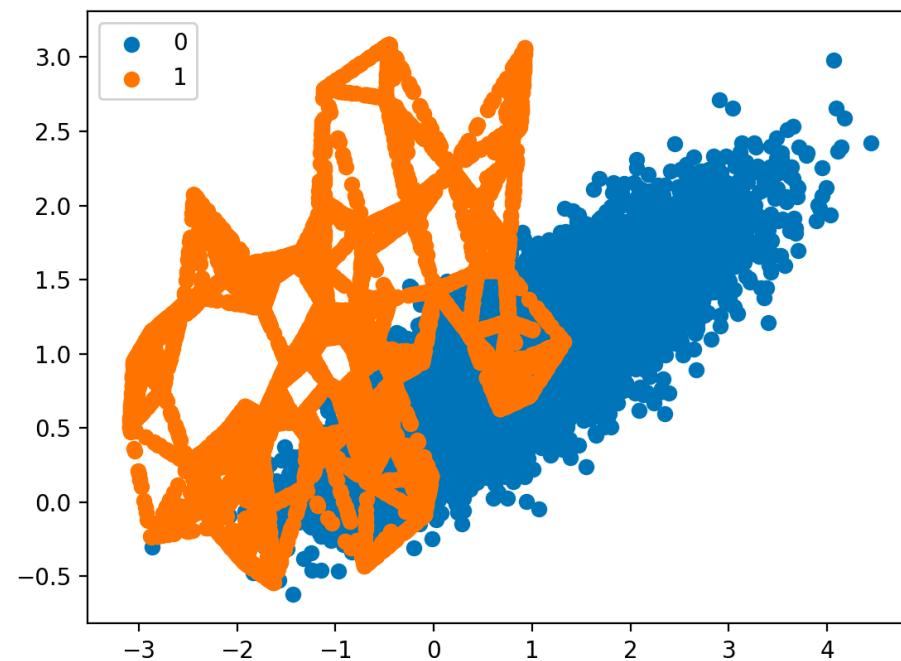
```
In [4]: n = minmax_scale(df)
df2 = pd.DataFrame(n)
```

Supervised Learning and Imbalance data

Oversampling



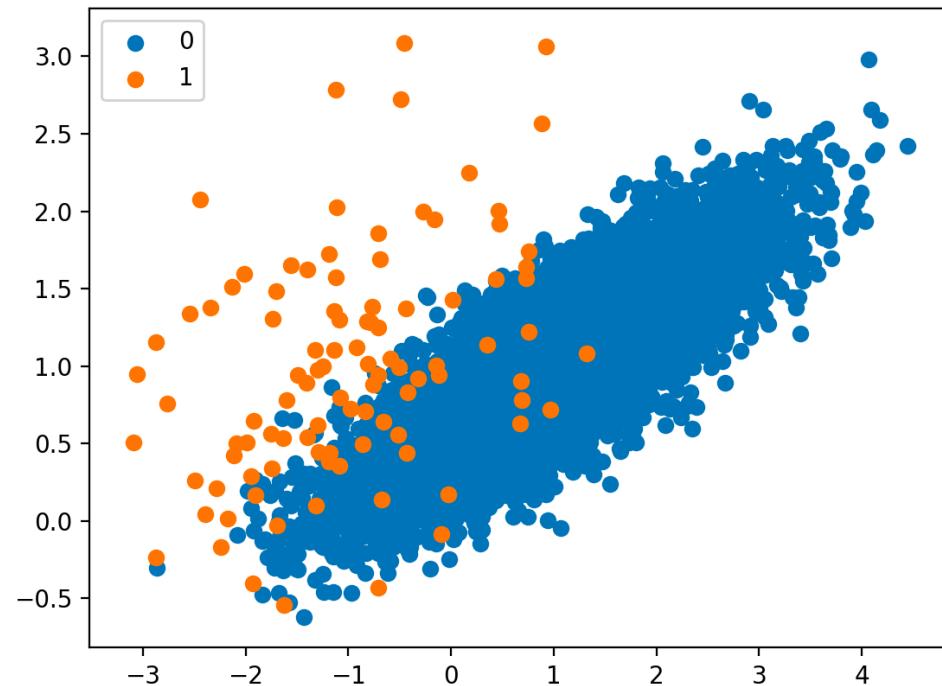
Original dataset



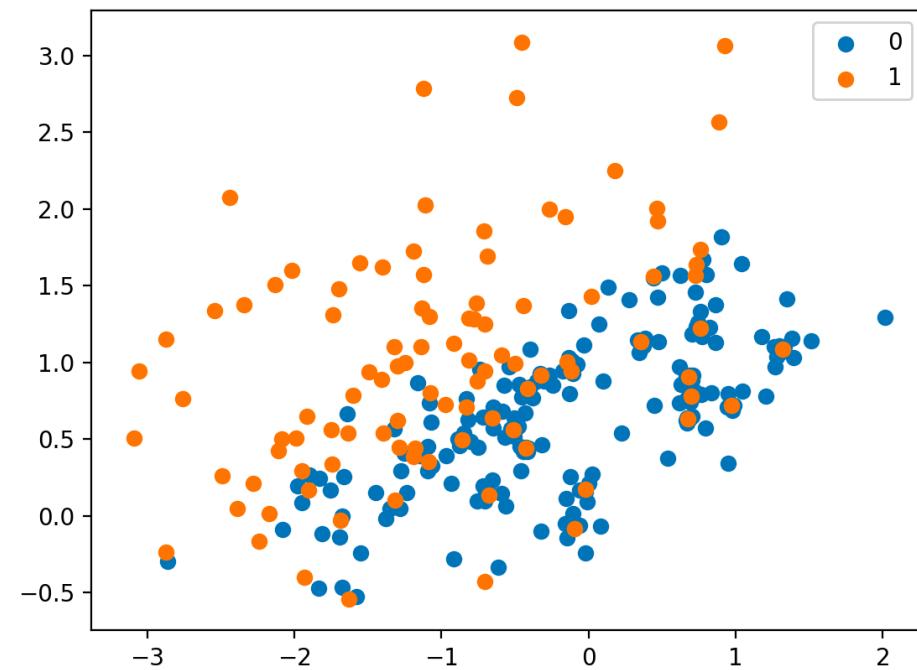
Over sampling with smote

Supervised Learning and Imbalance data

Undersampling



Original dataset



Undersampling

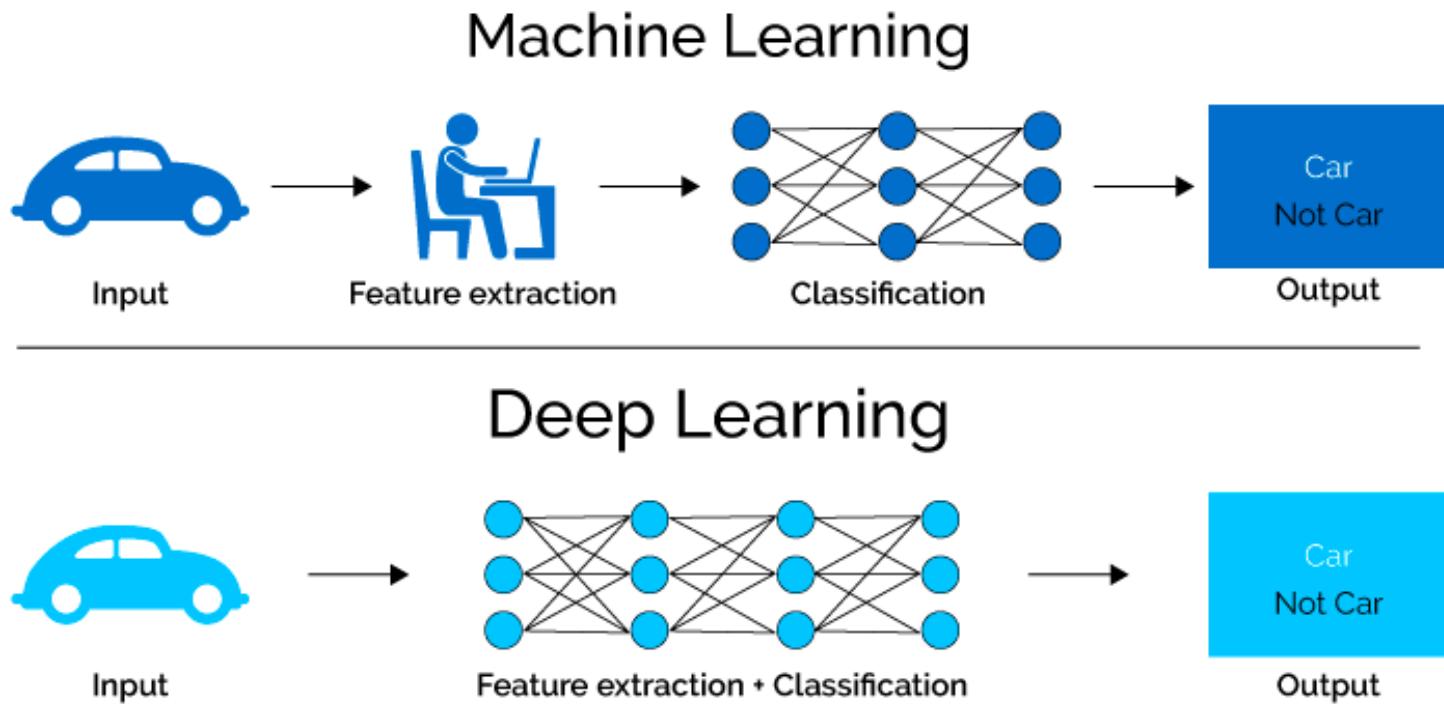


Neural Network & Deep Learning



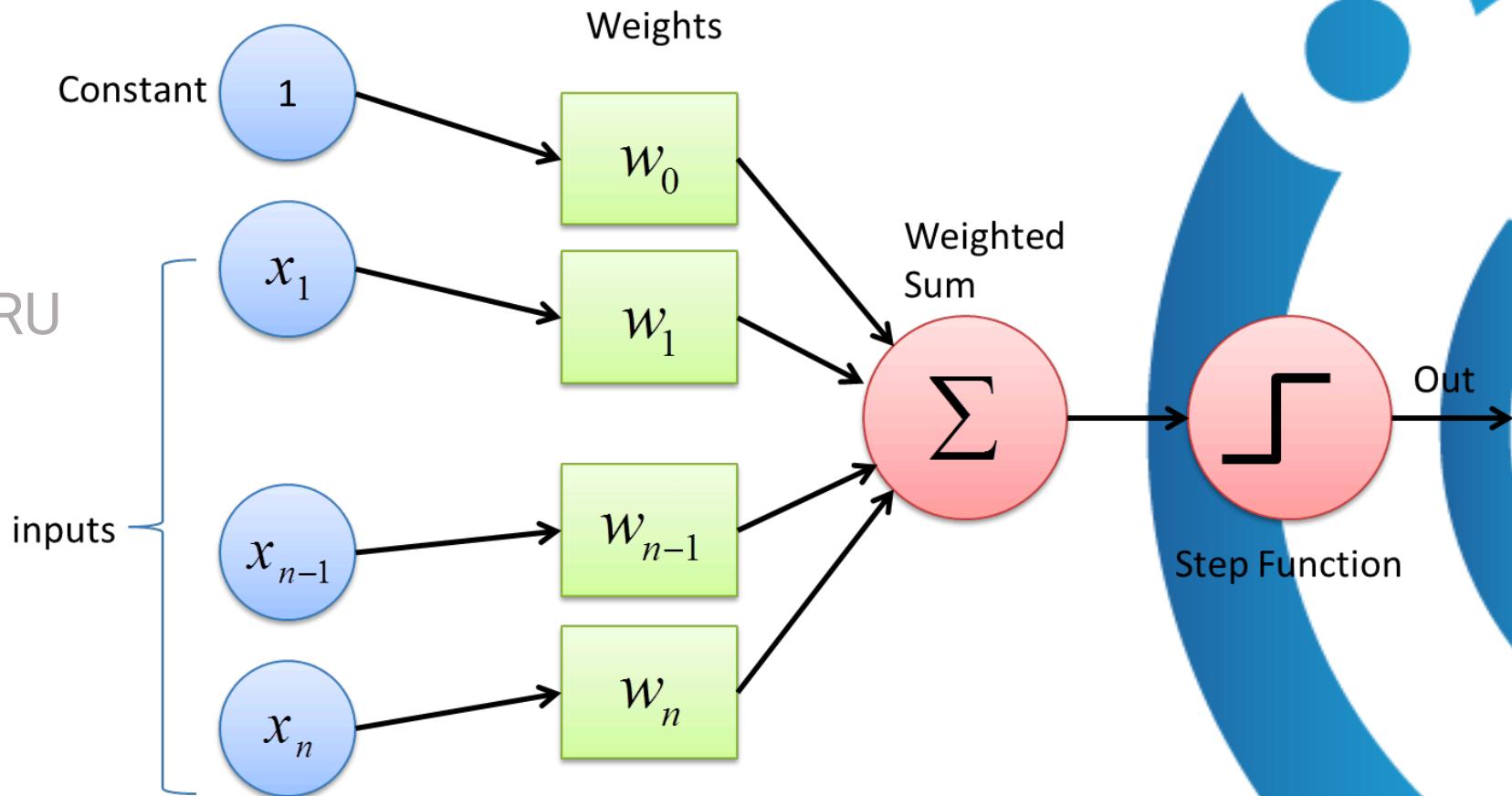
Deep Learning

- Part of machine learning field of learning representations of data. Exceptional effective at learning patterns.
- Utilizes learning algorithms that derive meaning out of data by using a hierarchy of multiple layers that mimic the neural networks of our brain.



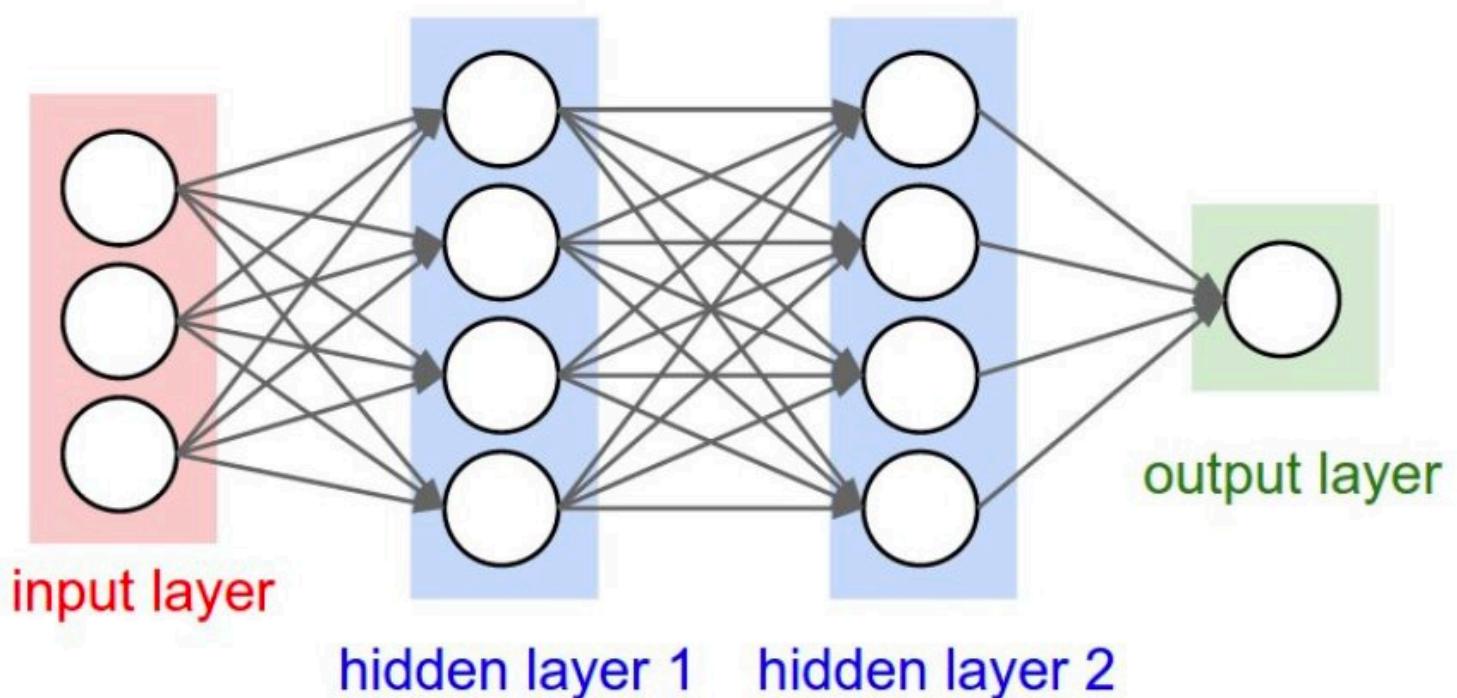
Neural Network

- NN
- DNN
- RNN
- LSTM & GRU



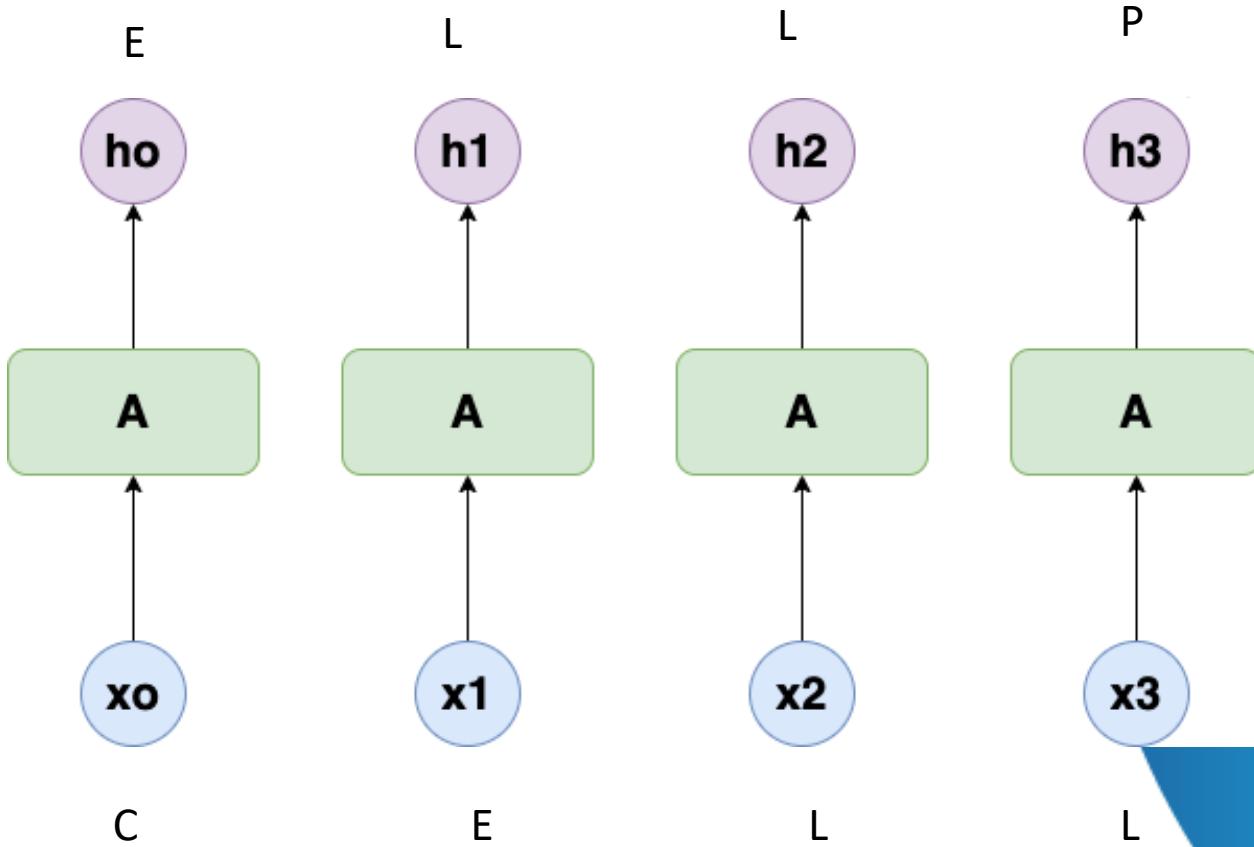
Deep Learning

- NN
- DNN
- RNN
- LSTM & GRU



Deep Learning

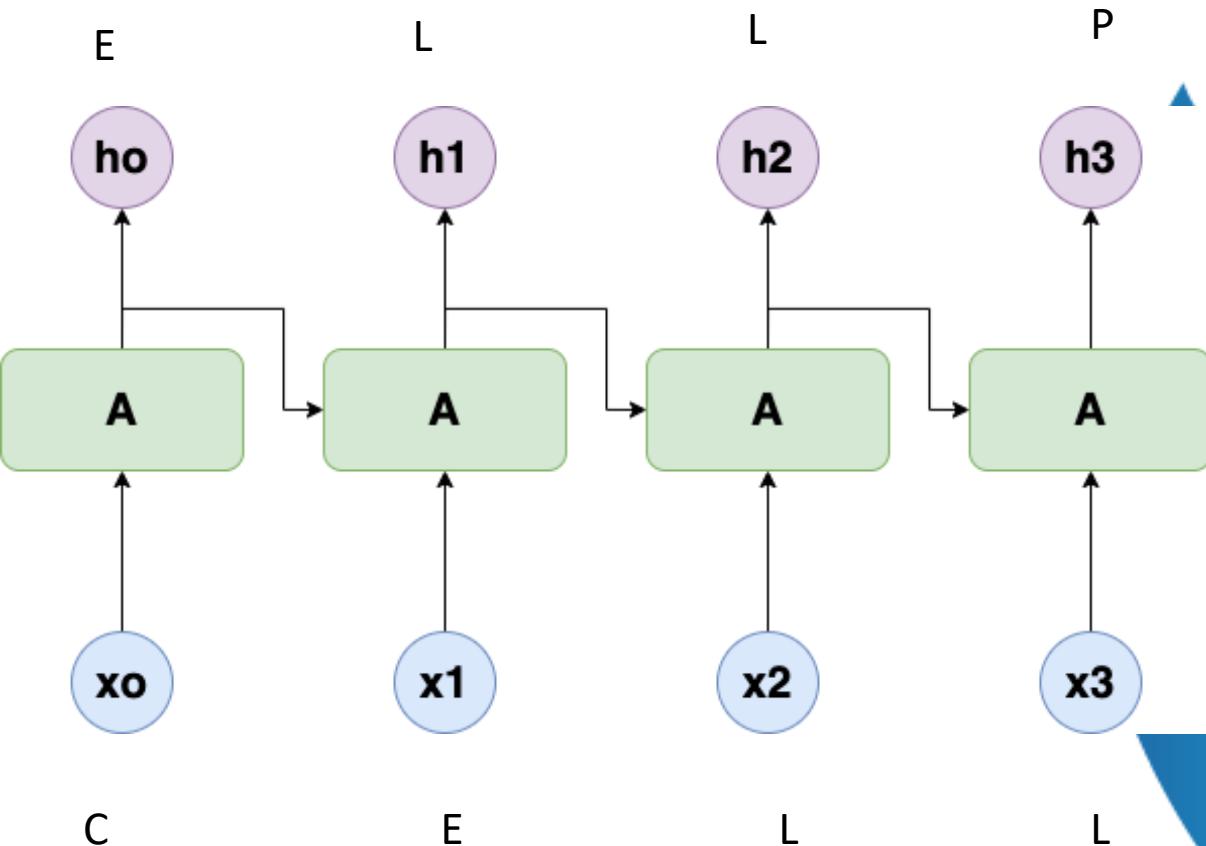
- NN
- DNN
- RNN
- LSTM & GRU



Cellphone

Deep Learning

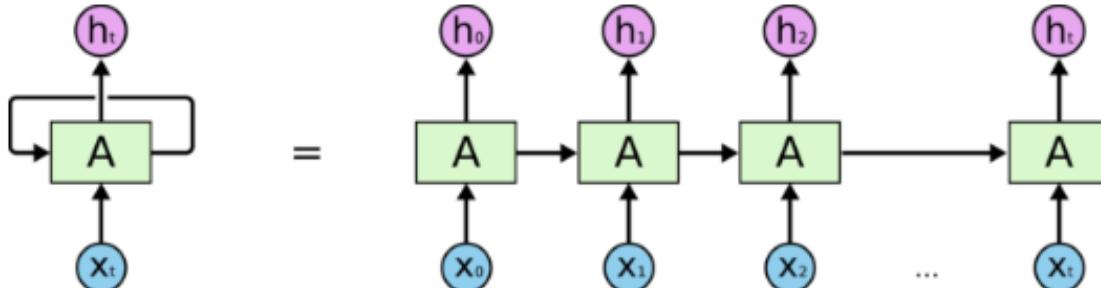
- NN
- DNN
- RNN
- LSTM & GRU



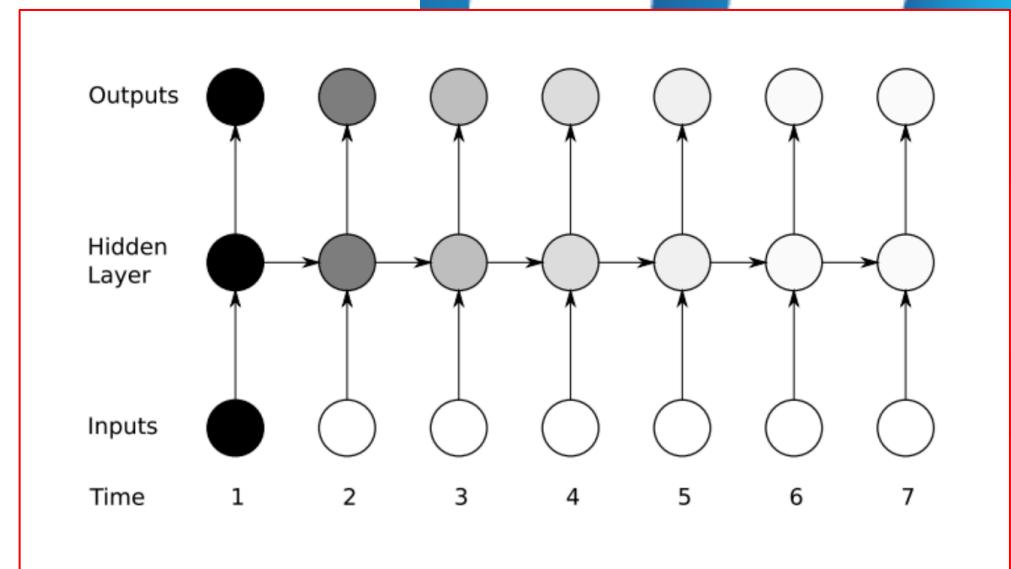
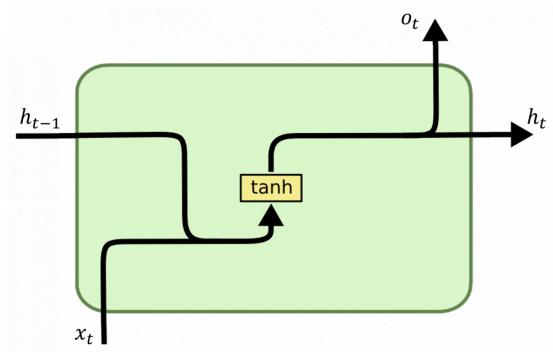
Cellphone

Deep Learning

- NN
- DNN
- RNN
- LSTM & GRU



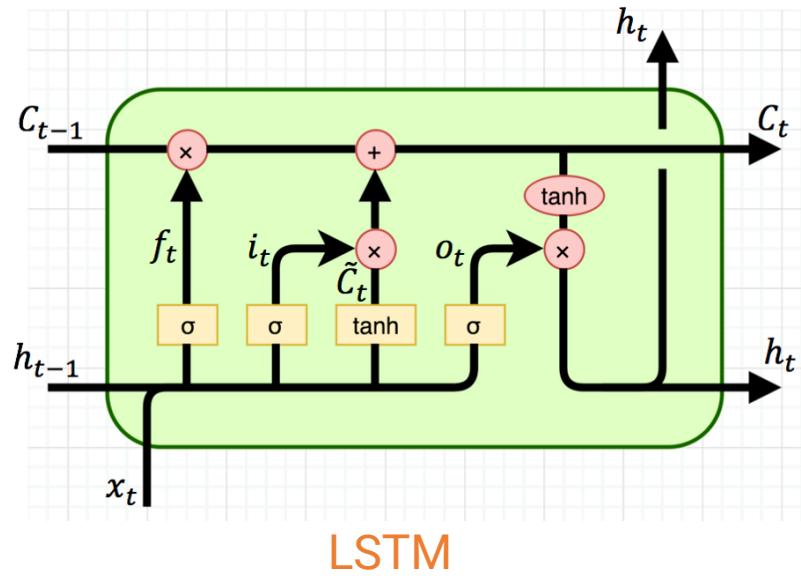
An unrolled recurrent neural network.



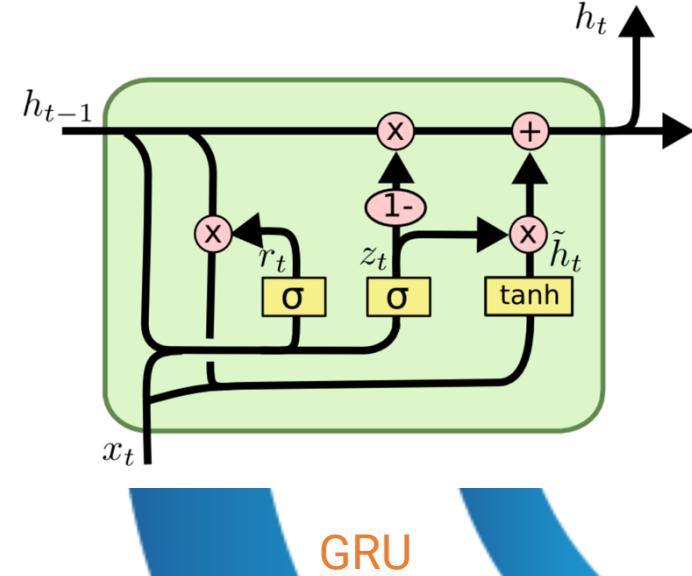
vanishing gradient

Deep Learning

- NN
- DNN
- RNN
- LSTM & GRU



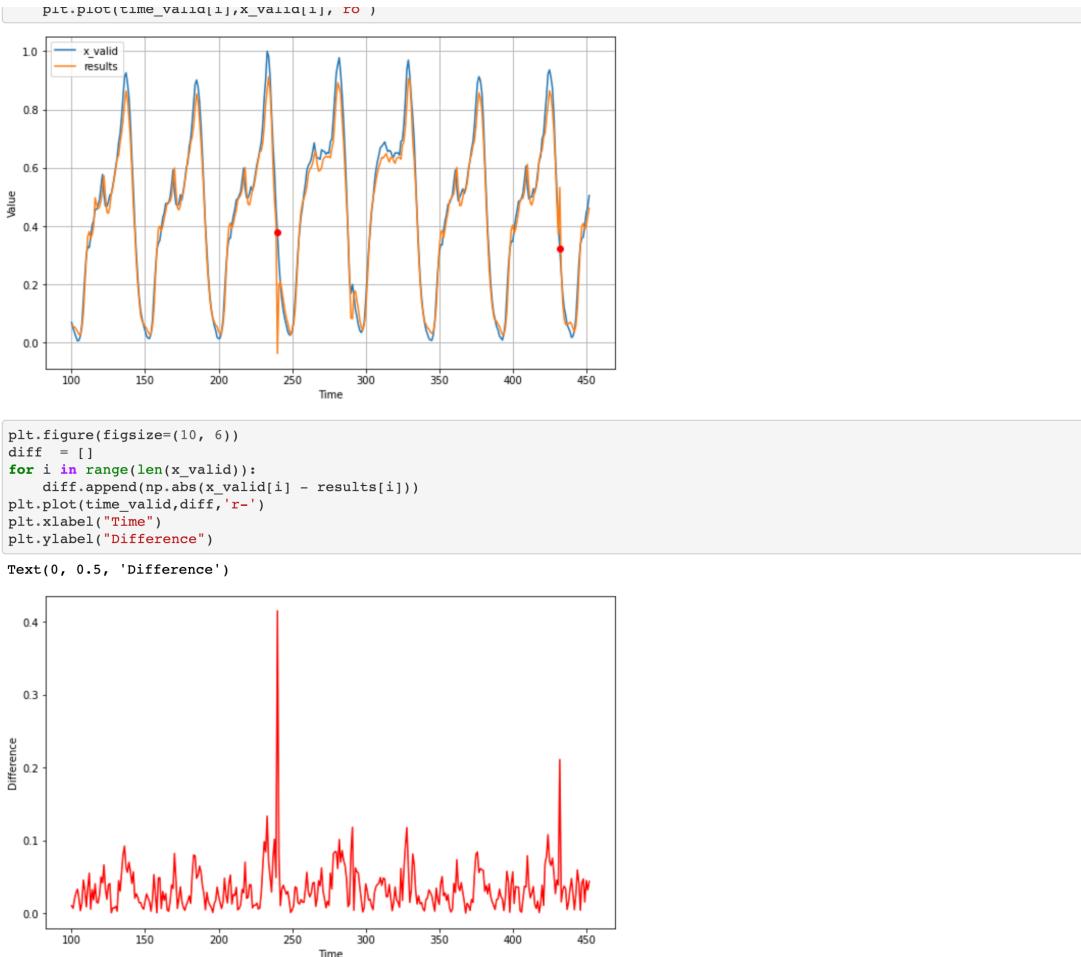
LSTM



GRU



Deep Learning



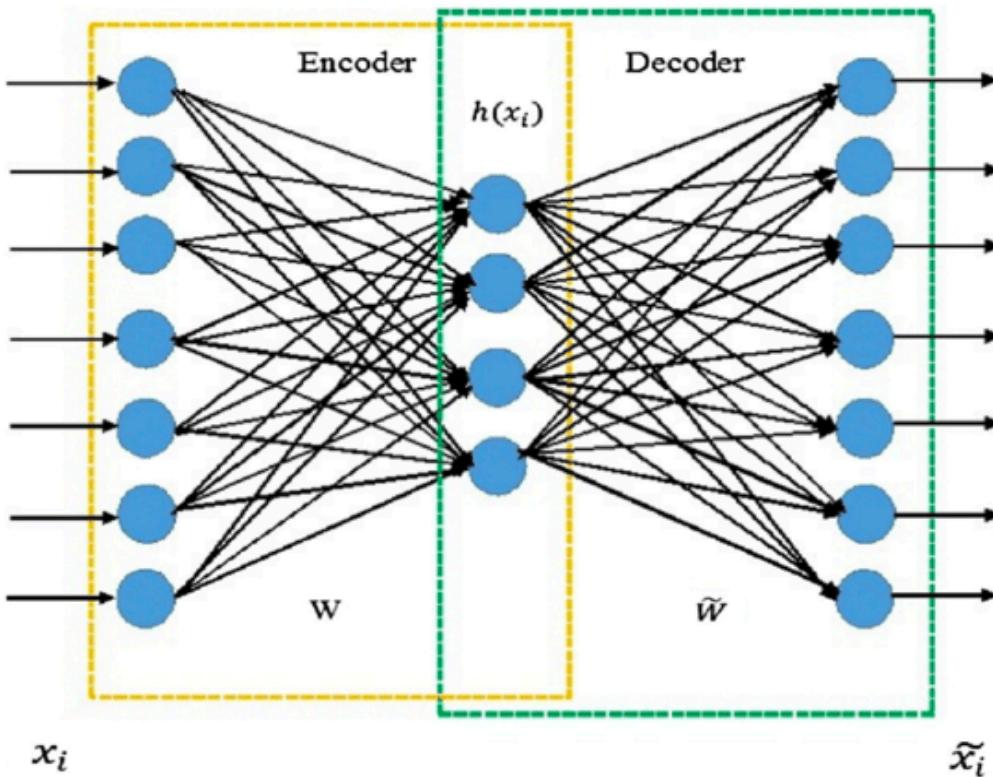


Anomaly Detection with Autoencoder

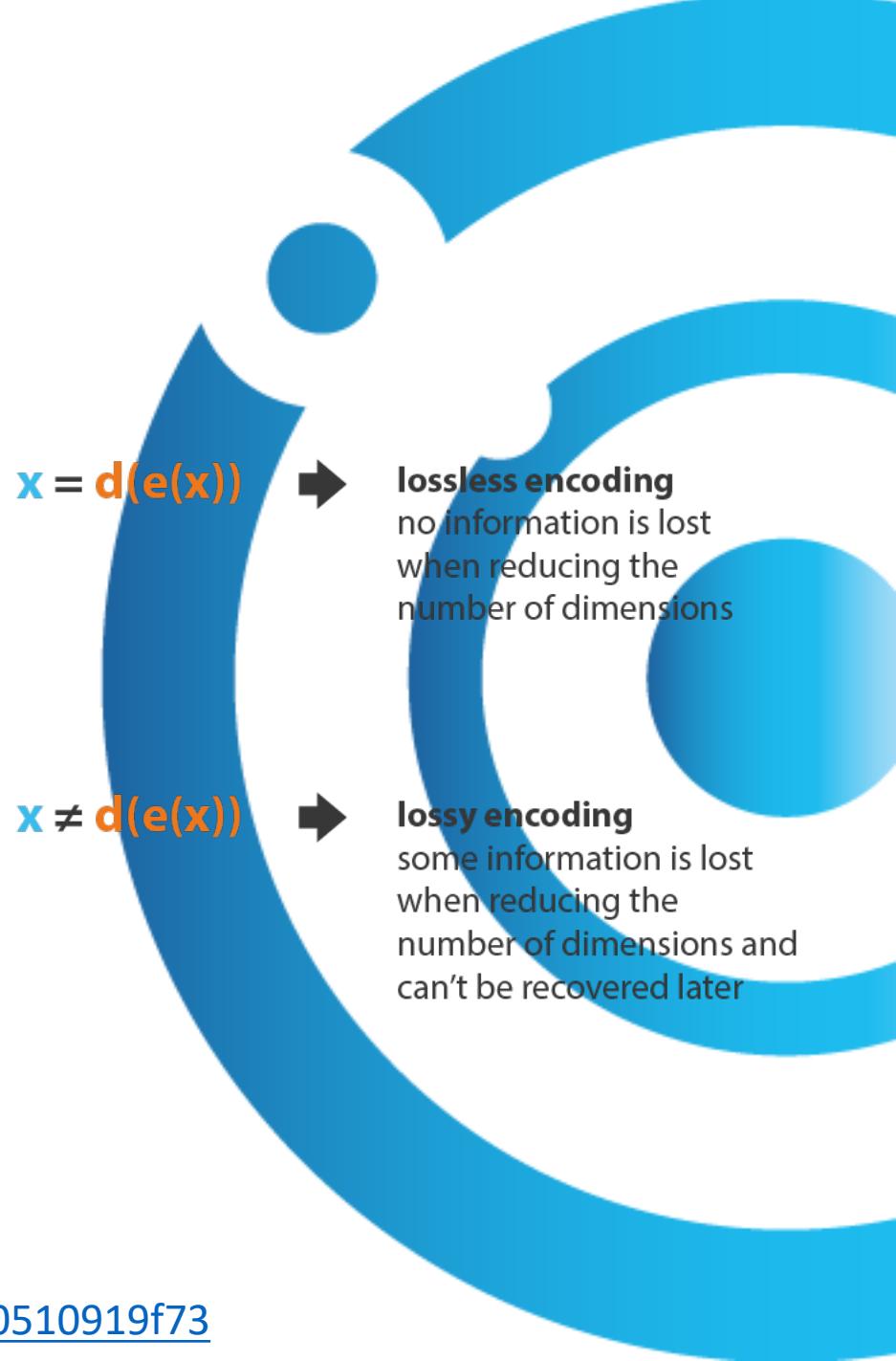
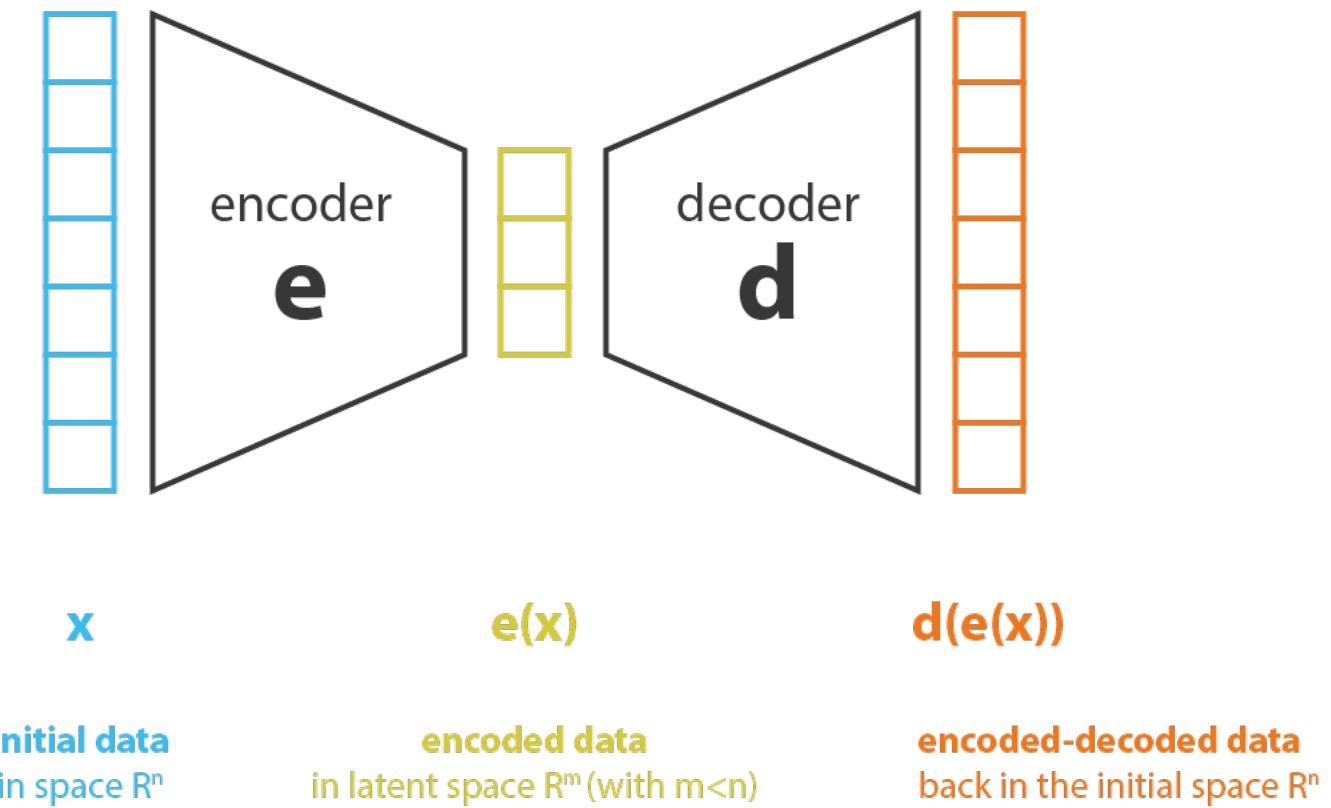


Autoencoder

Autoencoders are a specific type of feedforward neural networks where the input is the same as the output.

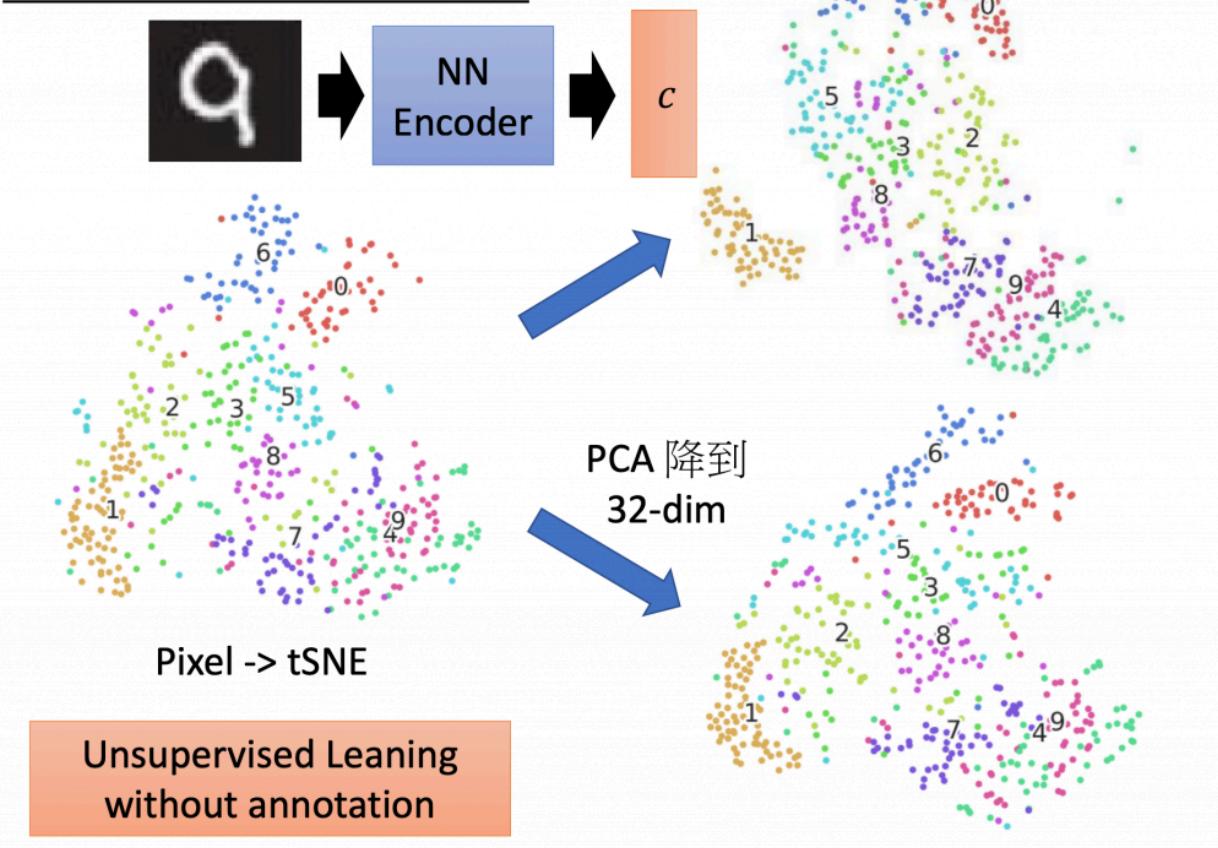


การทำงานของ Autoencoder



ความสามารถในการทำ Dimension Reduction

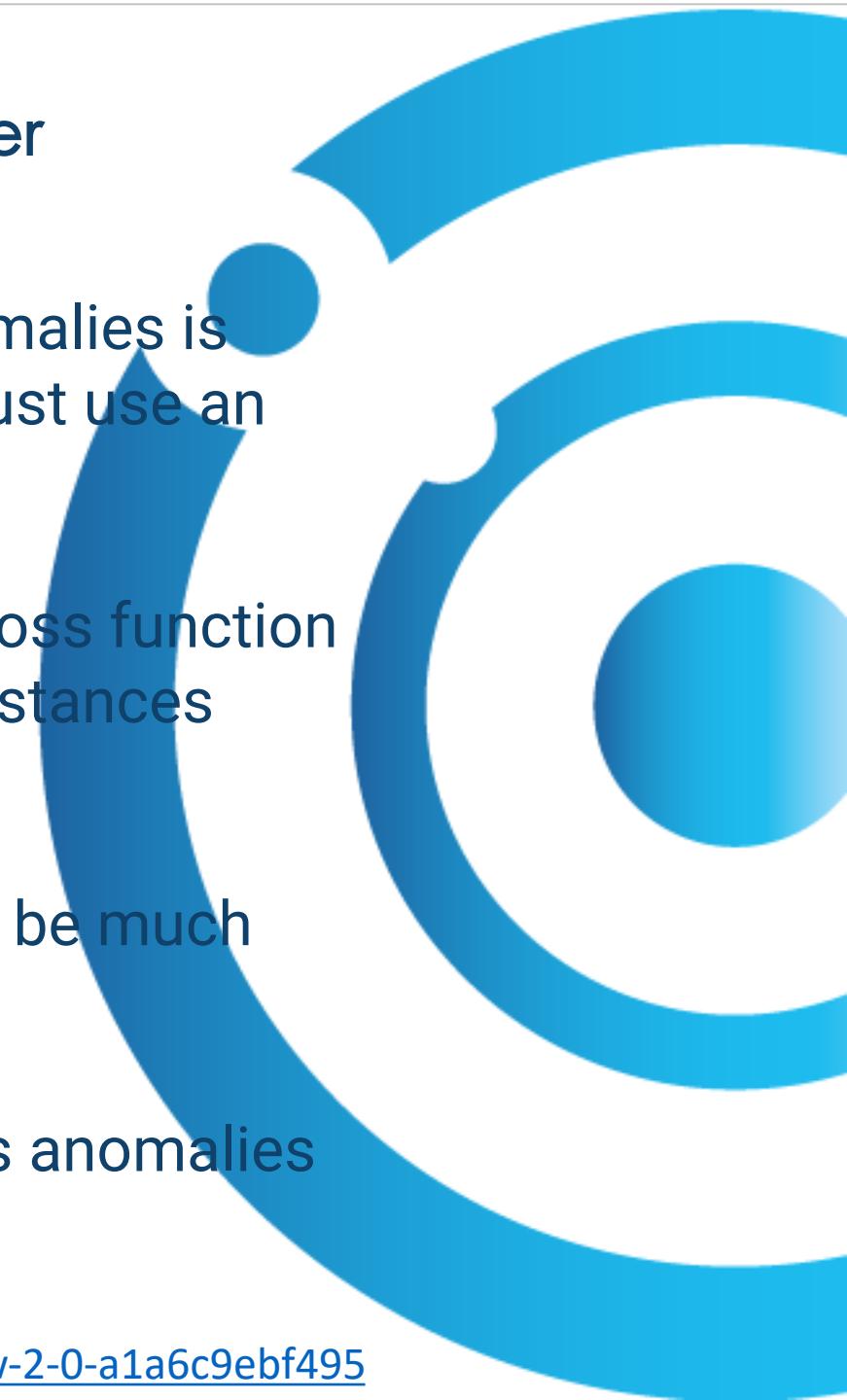
Auto-encoder - Example





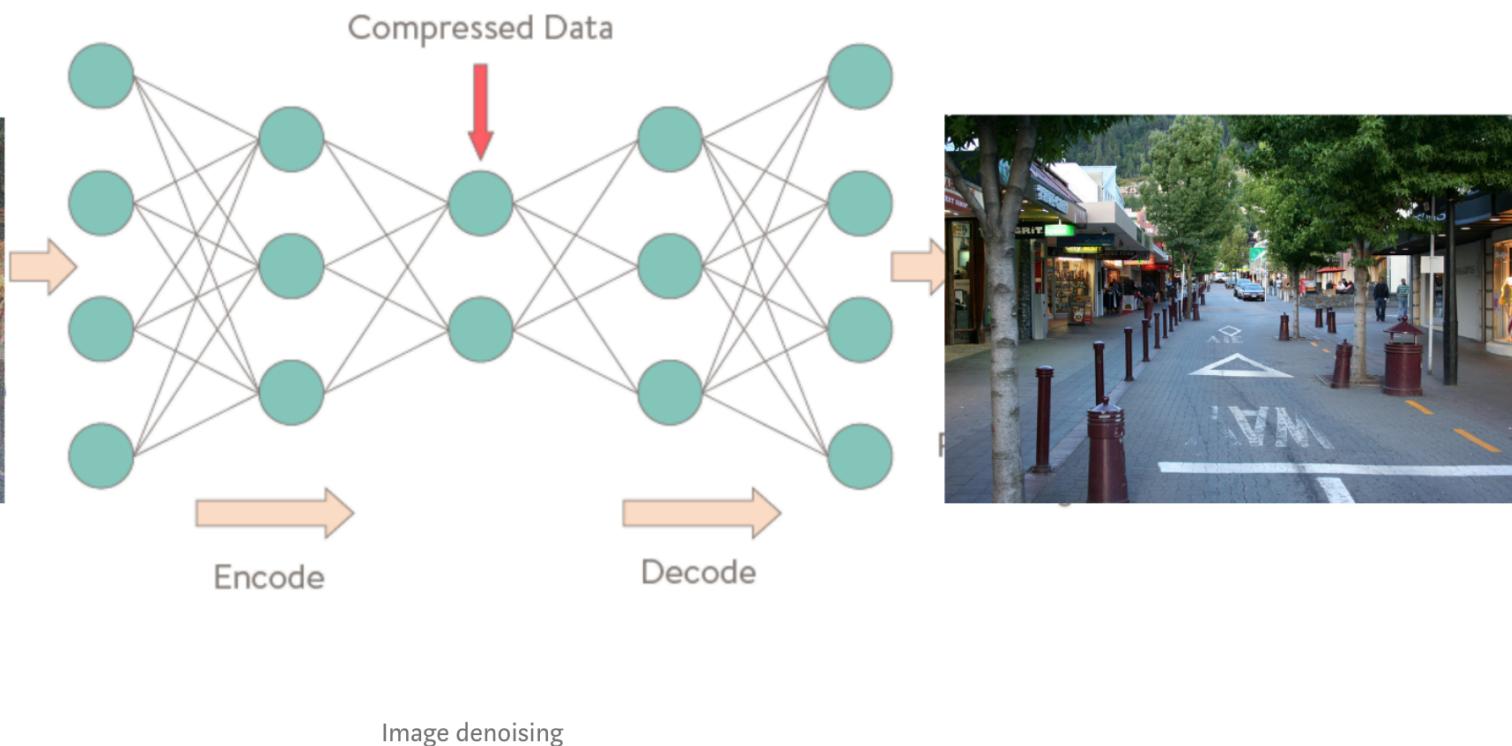
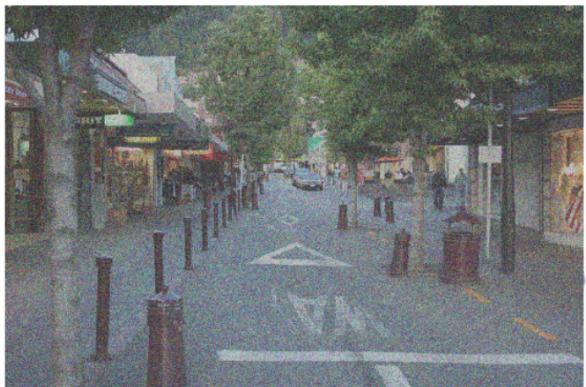
Anomaly detection step with Autoencoder

- First, if your dataset that should be examined for anomalies is imbalanced (which is almost always the case) you must use an autoencoder
- Train the autoencoder using the mean squared error loss function only on normal data, don't use any anomalous data instances during training
- After the training, the loss value for anomalies should be much higher than for normal data instances
- Find out a loss value threshold that best distinguishes anomalies from normal data.





Autoencoder application





Autoencoder Example

AnomalyDetection / Autoencoder.ipynb

krmonline Created using Colaboratory

File display

Latest commit 0fb09aa 1 hour ago History

1 contributor

341 lines (341 sloc) | 9.19 KB

In [1]:

```
import tensorflow as tf
import numpy as np
from sklearn.metrics import mean_squared_error
```

In [15]:

```
x = [0,1,2,3,4,5,6,7]
X = tf.keras.utils.to_categorical(x)
X
```

Out[15]:

```
array([[1., 0., 0., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0., 0., 0.],
       [0., 0., 0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 0., 0., 0., 1.]], dtype=float32)
```

In [16]:

```
inputs = tf.keras.Input(shape=X.shape[1])
encoder = tf.keras.layers.Dense(3, activation='sigmoid')
decoder = tf.keras.layers.Dense(X.shape[1], activation='sigmoid')
outputs = decoder(encoder(inputs))
model = tf.keras.Model(inputs,outputs)
```

In [17]:

```
model.compile(loss=tf.keras.losses.MeanSquaredError(),
              optimizer=tf.keras.optimizers.SGD(learning_rate=3.5))
```

In [70]:

```
model.fit(X,X,epochs=2000,verbose=0)
```

Out[70]:

```
<tensorflow.python.keras.callbacks.History at 0x7f7ceff4ae10>
```

In [71]:

```
pred = model.predict(np.array([[0., 1., 0., 0., 0., 0., 0., 0.]]))
print(pred)
print(np.round(pred))
```

rr1 0.024011e-03 8.8290018e-01 9. 2.455745e-04 8. 0.137193e-02 2. 5.001941e-06





Autoencoder Step

In [13]:

```
import numpy as np
import tensorflow as tf
#from keras.models import Sequential, Model
#from keras.layers import LSTM ,RepeatVector ,TimeDistributed , Dense , GRU
#from keras.utils import plot_model
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.metrics import mean_squared_error
```

In [15]:

```
df = pd.read_csv('https://raw.githubusercontent.com/krmonline/AnomalyDetection/master/data/timeseries1.csv',names=[ 'eps'])
len(df.eps)
series = df.eps.values
```

In [16]:

```
from sklearn.preprocessing import minmax_scale
```

In [64]:

```
#Prepare
def split_sequence(sequence, n_steps):
    X, y = list(), list()
    for i in range(len(sequence)):
        # find the end of this pattern
        end_ix = i + n_steps
        # check if we are beyond the sequence
        #print("end_ix="+str(end_ix))
        if end_ix > len(sequence)-1:
            break
        seq_x = [tmp for tmp in sequence[i:end_ix]]
        X.append(seq_x)
    return np.array(X)

seq_step = 5
```

In [4]:

```
plt.plot(df)
```

Out[4]:

```
[<matplotlib.lines.Line2D at 0x7fa2809ac240>]
```



Workshop



- ใช้ Technique ต่าง ๆ จากที่ได้เรียนรู้เพื่อหา Anomaly จาก File timeseries5_anomaly.csv

- ใช้ Technique ต่าง ๆ จากที่เรียนรู้ หาค่า Anomaly จาก ODDS Data Set



Thank You

