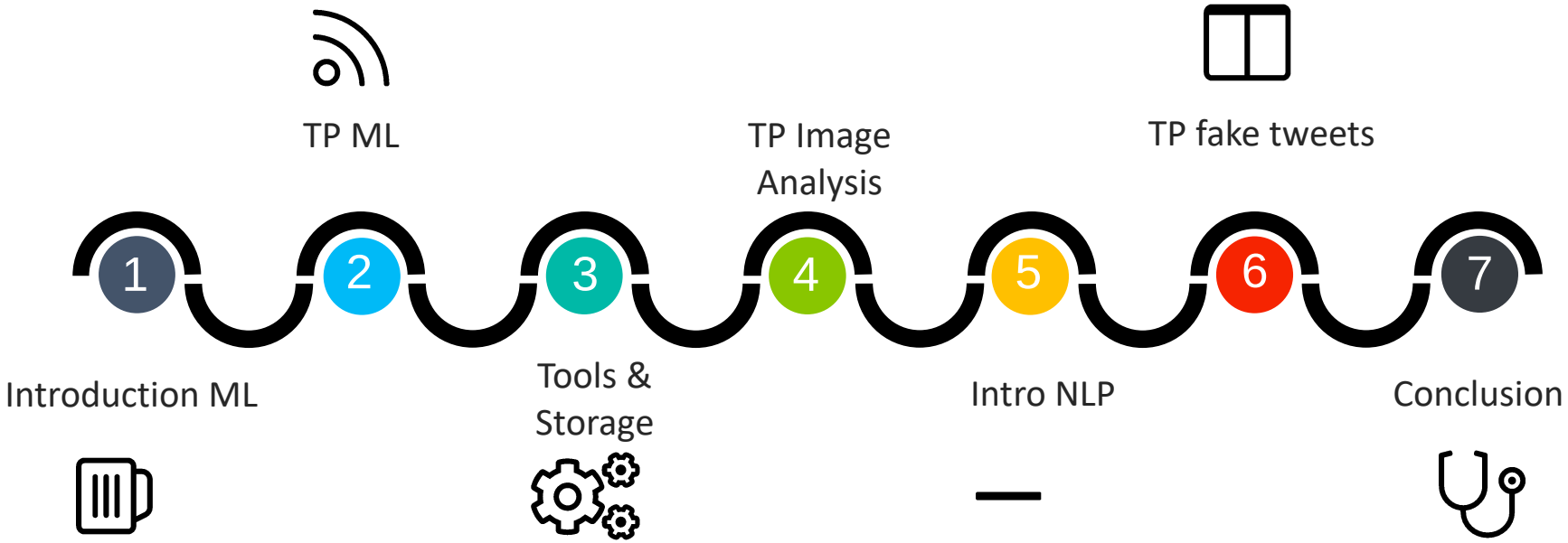
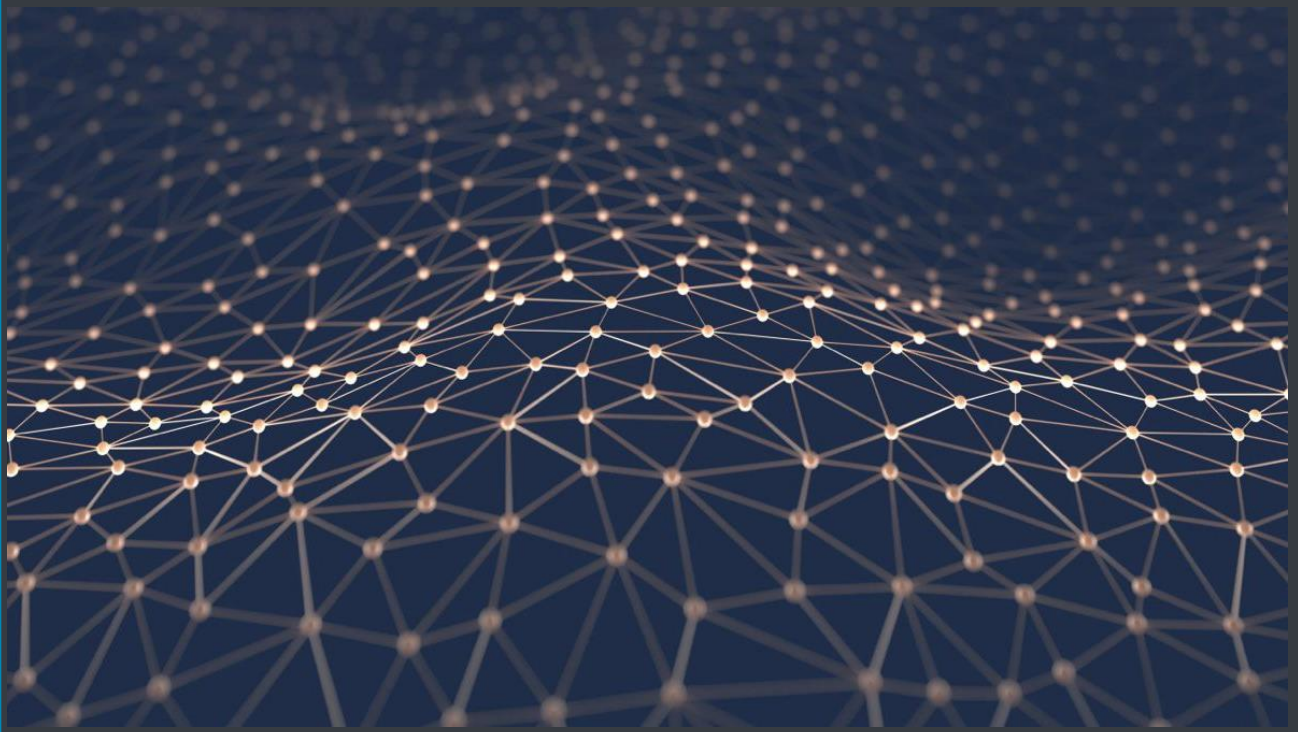

Introduction to IA
With Python
Mickael BOLNET – Python Instructor

Introduction to IA

Table of contents

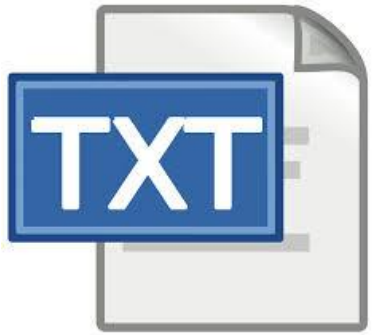


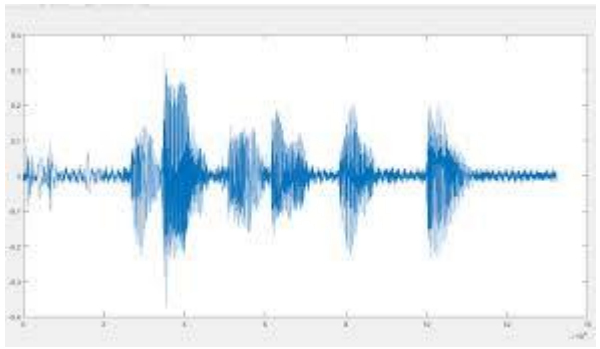


Intro to Machine Learning

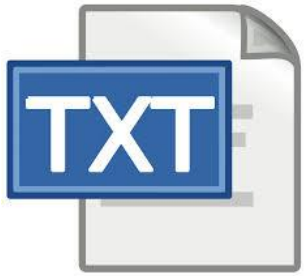
50s - Walter Pitts & Warren McCulloch
60s – Bayesian Methods
1969 - Marvin Minsky : non linearity & conexity issue for perceptrons
80s – Multilayer perceptron (still not enough power)
90s – SVM
2000s – Kernel Methods : SVC
2009 – ImageNet : The importance of DBs
2014 XGBoost
2017 Transformer Nets : « All you need is Attention »

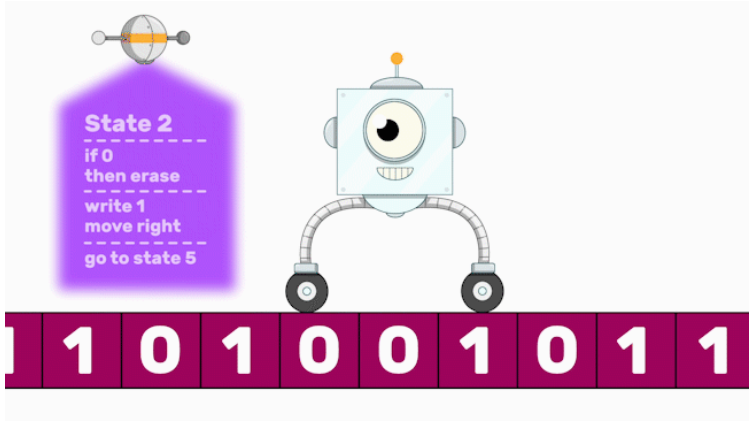
Introduction to Machine Learning

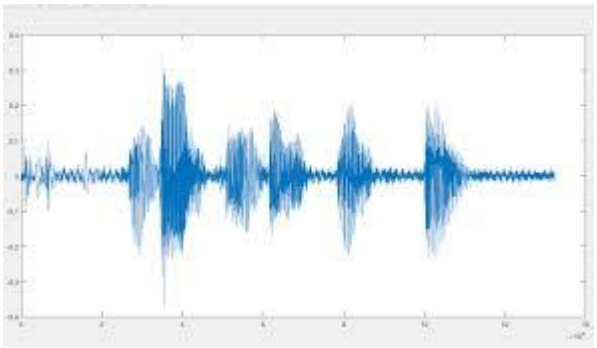




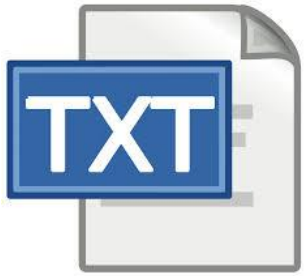
Introduction to Machine Learning

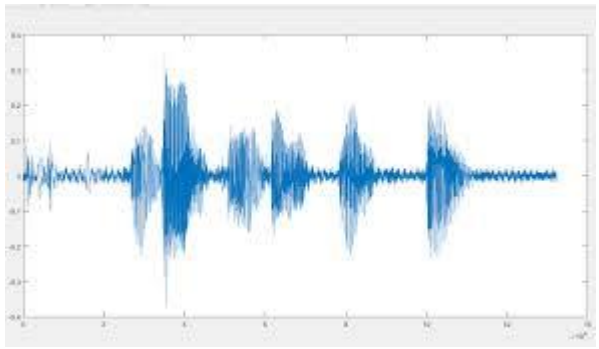






Algorithm



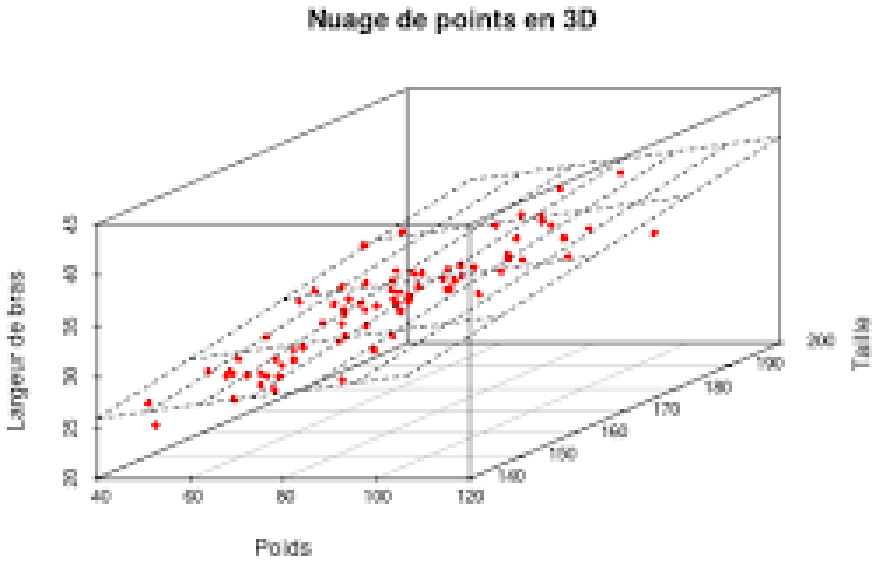


Y = Fp(x)

Decision : Yes (1) or No (0)
Labels : Dog, Cat, Person…
Product : Another Image

Simple Example

$$Y = F_{a,b}(X) = a * x_1 + b * x_2$$
$$avec\ X = \begin{matrix} x1 \\ x2 \end{matrix}$$



Simple Example

$$Y = F_{a,b}(X) = a * x_1 + b * x_2$$
$$avec\ X = \begin{matrix} x1 \\ x2 \end{matrix}$$

Suite d'exemples :

$(Y_1^*, X_1^*), (Y_2^*, X_2^*), (Y_3^*, X_3^*), (Y_4^*, X_4^*), (Y_5^*, X_5^*)$ …

Simple Example

$$Y = F_{a,b}(X) = a * x_1 + b * x_2$$
$$avec\ X = \begin{matrix} x1 \\ x2 \end{matrix}$$

Suite d'exemples :

$(Y_1^*, X_1^*), (Y_2^*, X_2^*), (Y_3^*, X_3^*), (Y_4^*, X_4^*), (Y_5^*, X_5^*)$ …

Erreurs:

$$Y_1^* - F_{a,b}(X_1)$$
$$Y_2^* - F_{a,b}(X_2)$$
…

Simple Example

$$Y = F_{a,b}(X) = a * x_1 + b * x_2$$
$$avec\ X = \begin{matrix} x1 \\ x2 \end{matrix}$$

Suite d'exemples :

$(Y_1^*, X_1^*), (Y_2^*, X_2^*), (Y_3^*, X_3^*), (Y_4^*, X_4^*), (Y_5^*, X_5^*)$ …

Erreurs:

$$Y_1^* - F_{a,b}(X_1)$$
$$Y_2^* - F_{a,b}(X_2)$$
$$\dots$$

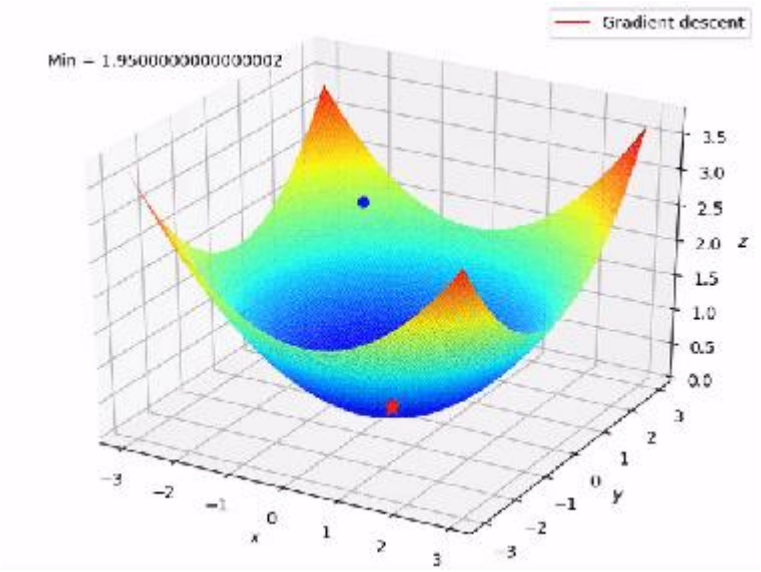
Minimiser :

$$J(a,b) = \sum_i \left(E_i(a,b)\right)^2$$

Simple Example

Fonction modèle simple -> Solution algébrique

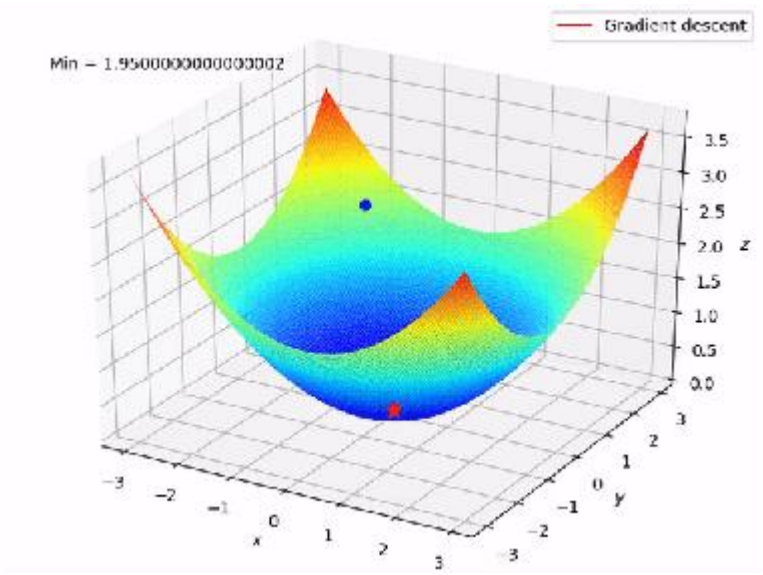
Fonction modèle complexe et prise en compte du bruit -> Autres algos : i.e. Descente de Gradient

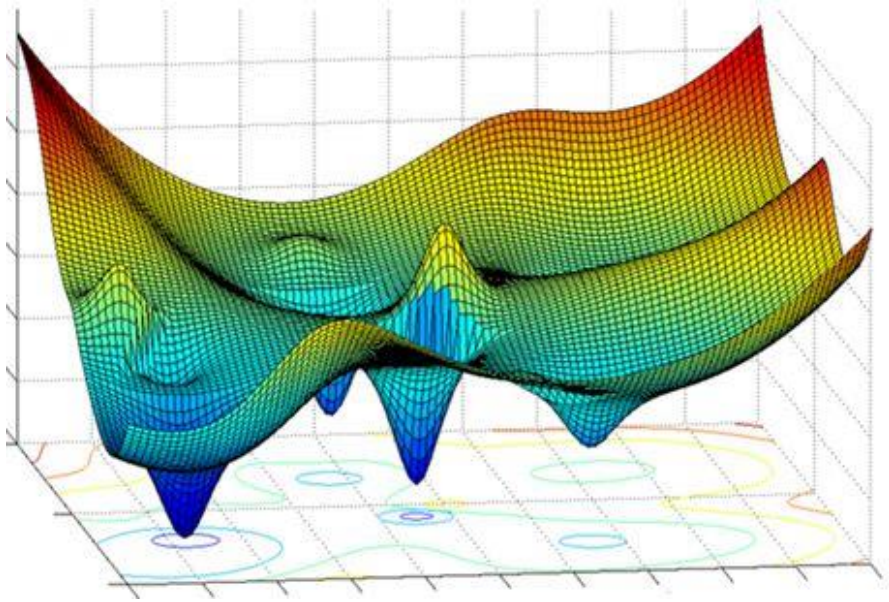


Simple Example

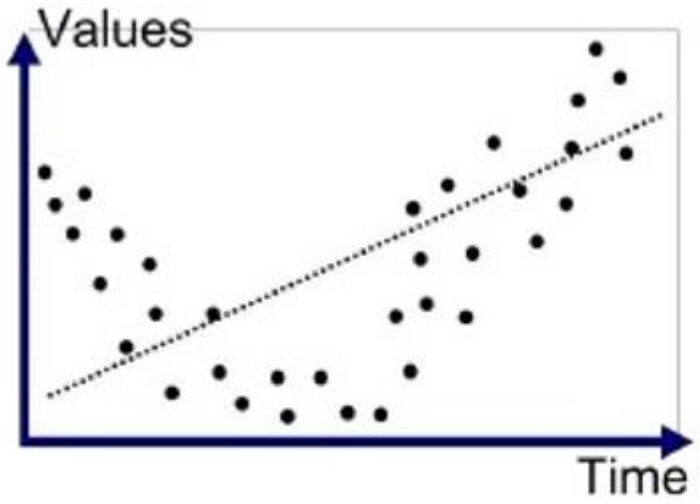
Fonction modèle simple -> Solution algébrique

Fonction modèle complexe et prise en compte du bruit -> Autres algos : i.e. Descente de Gradient

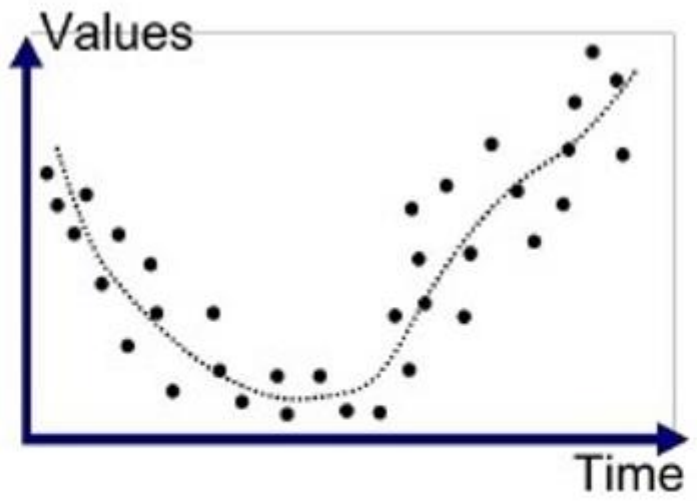




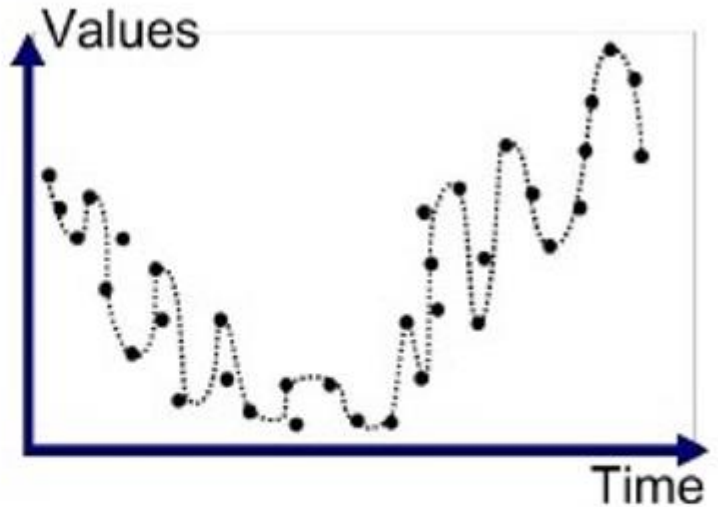
Problème du coût



Underfitted

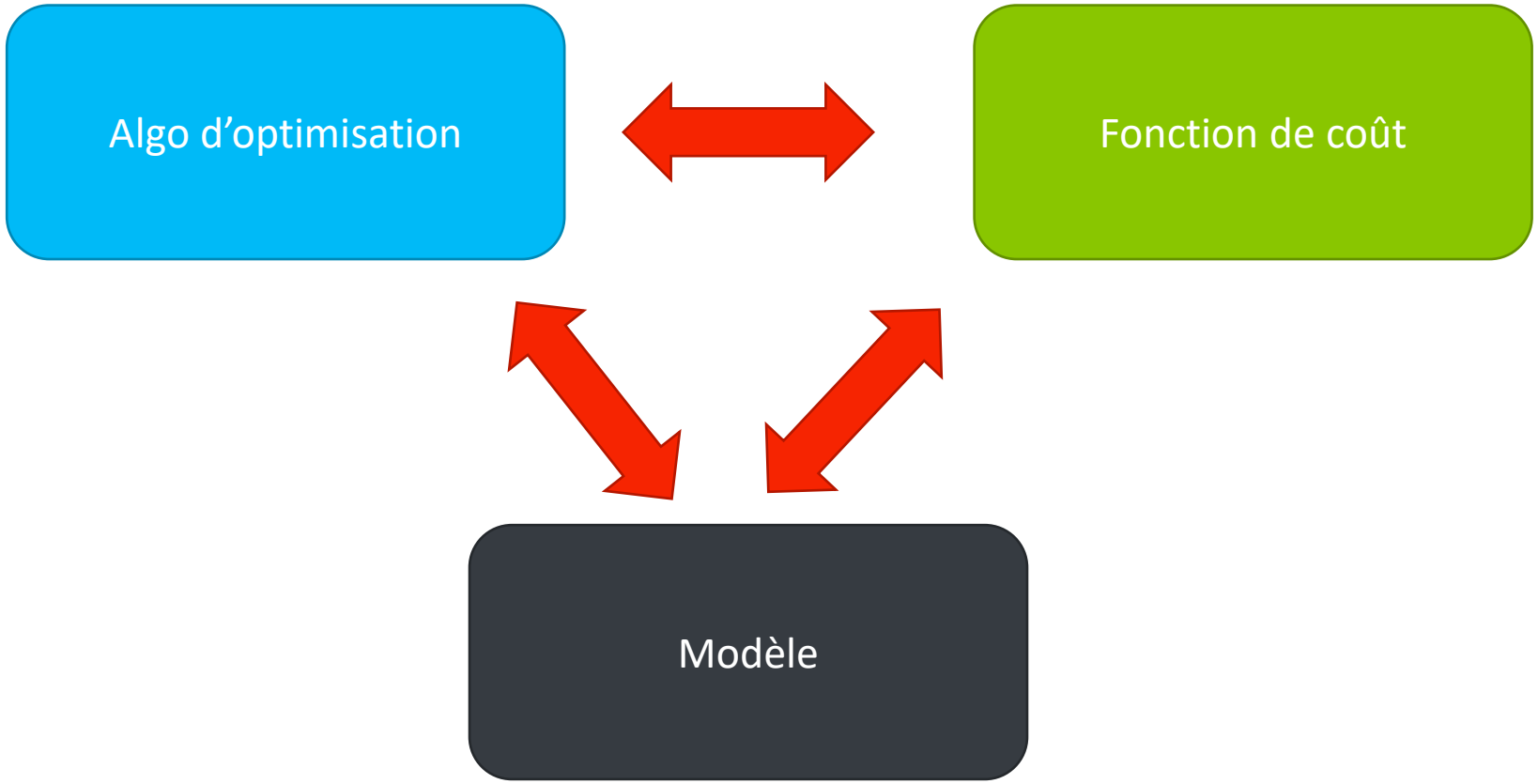


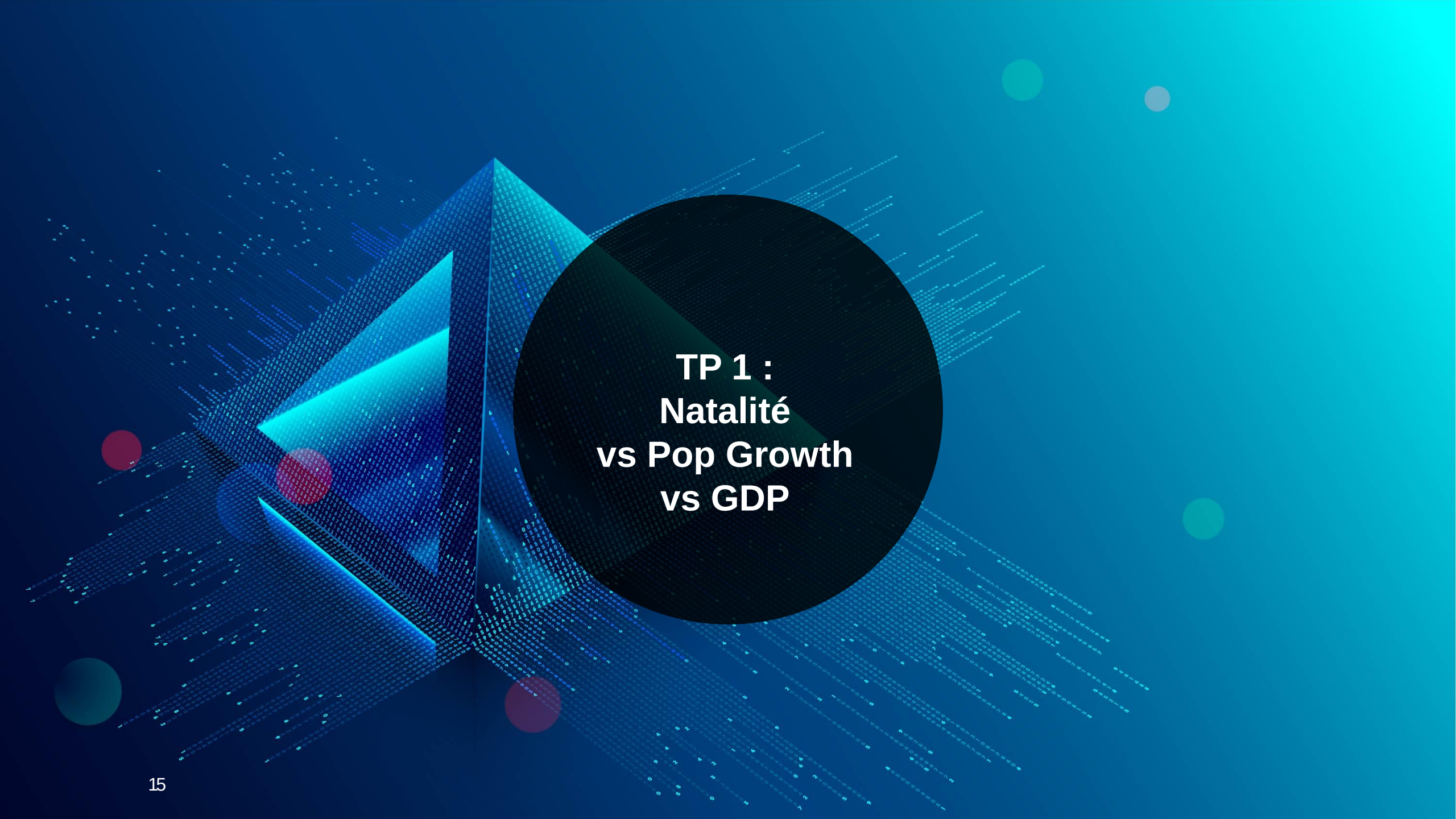
Good Fit/Robust



Overfitted

Simple Example



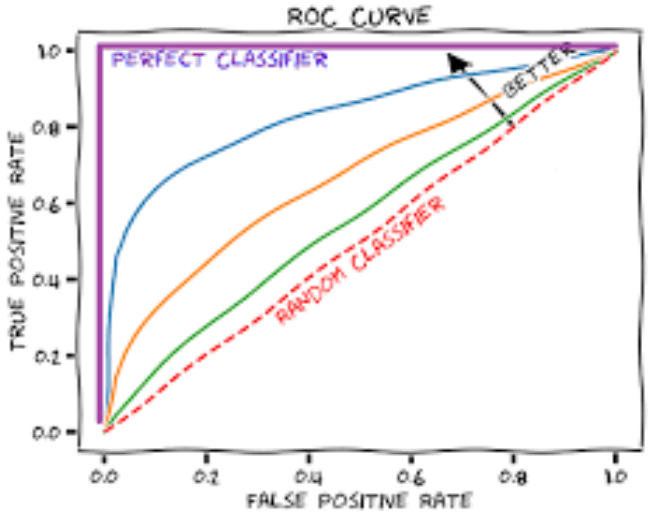

TP 1 :
Natalité
vs Pop Growth
vs GDP

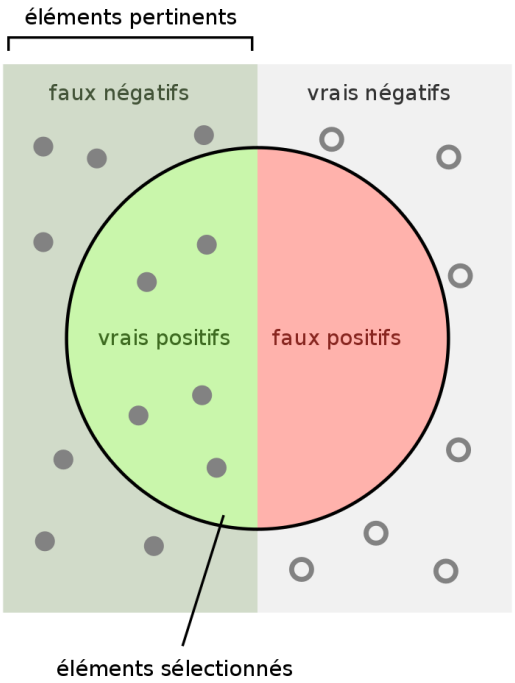
Scoring your algo

Basic Error measurement

(for classification)

1. Precision & Recall
2. ROC & AUC
3. Log loss





Combien de candidats sélectionnés sont pertinents ?

Précision = $\frac{\text{vrais positifs}}{\text{éléments sélectionnés}}$

Combien d'éléments pertinents sont sélectionnés ?

Rappel = $\frac{\text{vrais positifs}}{\text{éléments pertinents}}$

Scoring your algo

Basic Error measurement

(for regressions)

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

$$\hat{R}^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2} = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})^2}$$

$$R^2_{adj} = 1 - \left[\frac{(1 - R^2)(n-1)}{n-k-1}\right]$$

CASE 1: Evenly distributed errors

ID	Error	|Error|	Error^2
1	2	2	4
2	2	2	4
3	2	2	4
4	2	2	4
5	2	2	4
6	2	2	4
7	2	2	4
8	2	2	4
9	2	2	4
10	2	2	4

MAE	RMSE
2.000	2.000

CASE 2: Small variance in errors

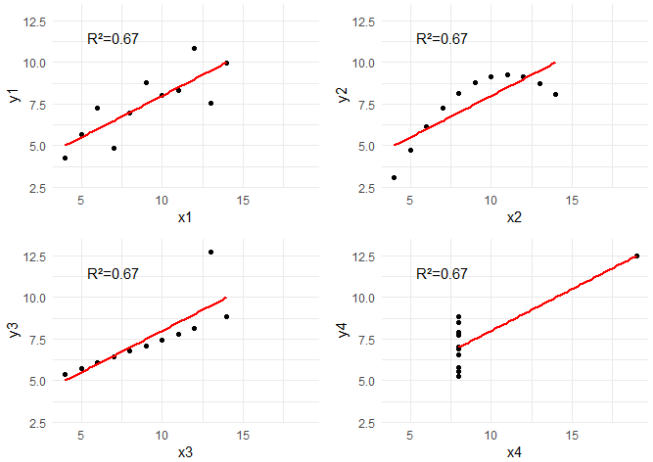
ID	Error	|Error|	Error^2
1	1	1	1
2	1	1	1
3	1	1	1
4	1	1	1
5	1	1	1
6	3	3	9
7	3	3	9
8	3	3	9
9	3	3	9
10	3	3	9

MAE	RMSE
2.000	2.236

CASE 3: Large error outlier

ID	Error	|Error|	Error^2
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	20	20	400

MAE	RMSE
2.000	6.325



Scoring your algo

BLEU (Bilingual Evaluation Understudy)

(for NLP)

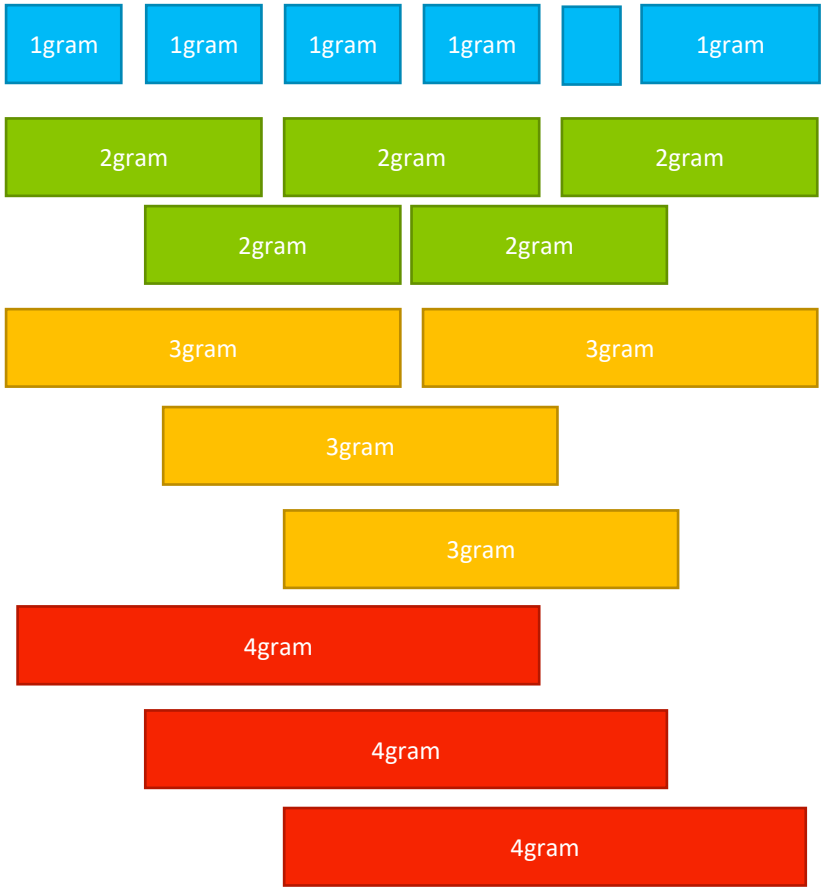
$$BLEU = BP * \exp\left(\sum w_n \log\left(P_n\right)\right)$$

$$BP = \begin{cases} 1 & c \geq r \\ \exp\left(1 - \frac{r}{c}\right) & c < r \end{cases}$$

Etapes:

1. Couper la phrase en n-grams de 1 à 4

Demain nous allons manger au restaurant



Scoring your algo

BLEU (Bilingual Evaluation Understudy)

(for NLP)

$$BLEU = BP * \exp\left(\sum w_n \log\left(P_n\right)\right)$$

$$BP = \begin{cases} 1 & c \geq r \\ \exp\left(1 - \frac{r}{c}\right) & c < r \end{cases}$$

Etapes:

1. Couper la phrase en n-grams de 1 à 4
2. Calculer la précision des n-grams (Pn)

Résultat attendu : *Demain nous allons manger au restaurant*
Résultat 1: *Demain manger au aller restaurant nous. P1 = 1*
Résultat 2: *Nous allons diner au restaurant demain. P1 = 0,83*

Résultat 1: P1 = 100%

Demain	Manger	au	Aller	Restaurant	nous
1	1	1	1	1	1

Résultat 2: P1 = 83%

Nous	allons	diner	au	Restaurant	demain
1	1	0	1	1	1

Scoring your algo

BLEU (Bilingual Evaluation Understudy)

(for NLP)

$$BLEU = BP * \exp\left(\sum w_n \log\left(P_n\right)\right)$$

$$BP = \begin{cases} 1 & c \geq r \\ \exp\left(1 - \frac{r}{c}\right) & c < r \end{cases}$$

Etapes:

1. Couper la phrase en n-grams
2. Calculer la précision des n-grams (Pn)

Résultat attendu : *Demain nous allons manger au restaurant*
Résultat 1: *Demain manger au aller restaurant nous. P1 = 1; P2 = 0,20*
Résultat 2: *Nous allons diner au restaurant demain. P1 = 0,83; P2=0,40*

Résultat 1: P1 = 20%

Demain manger	Manger au	Au aller	Aller restaurant	Restaurant nous
0	1	0	0	0

Résultat 2: P1 = 40%

Nous allons	Allons diner	Diner au	Au restaurant	Restaurant demain
1	0	0	1	0

Scoring your algo

BLEU (Bilingual Evaluation Understudy)

(for NLP)

$$BLEU = BP * \exp\left(\sum w_n \log\left(P_n\right)\right)$$

$$BP = \begin{cases} 1 & c \geq r \\ \exp\left(1 - \frac{r}{c}\right) & c < r \end{cases}$$

Etapes:

1. Couper la phrase en n-grams de 1 à 4
2. Calculer la précision des n-grams (Pn)
3. Calculer la moyenne pondérée des logPn
4. Exp. et Multiplier par la Brevity Penality

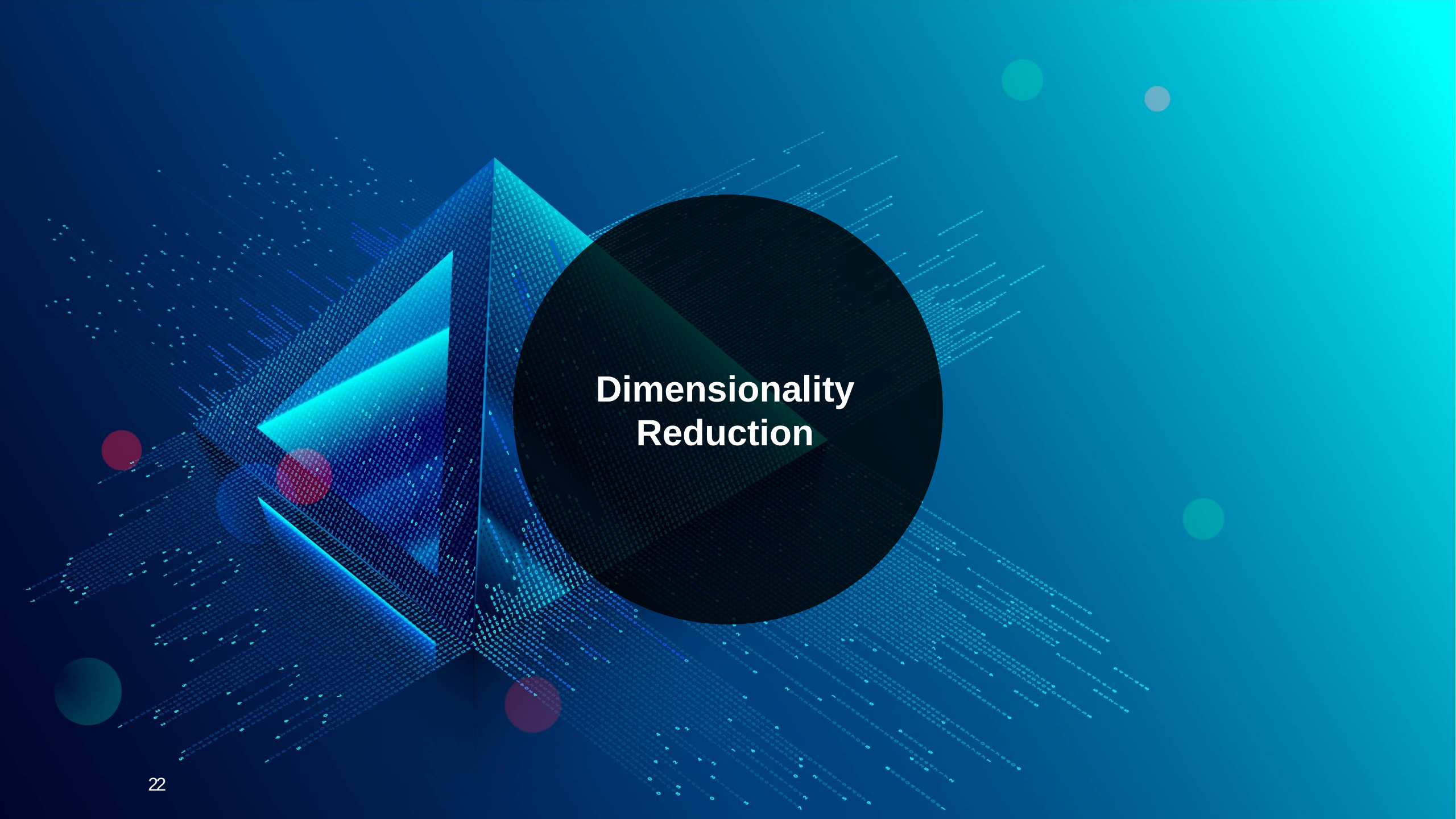
Résultat attendu : *Demain nous allons manger au restaurant*

Résultat 1: *Demain manger au aller restaurant nous. P1 = 1; P2 = 0,20;*

BLEU = 0,84

Résultat 2: *Nous allons diner au restaurant demain. P1 = 0,83; P2=0,40;*

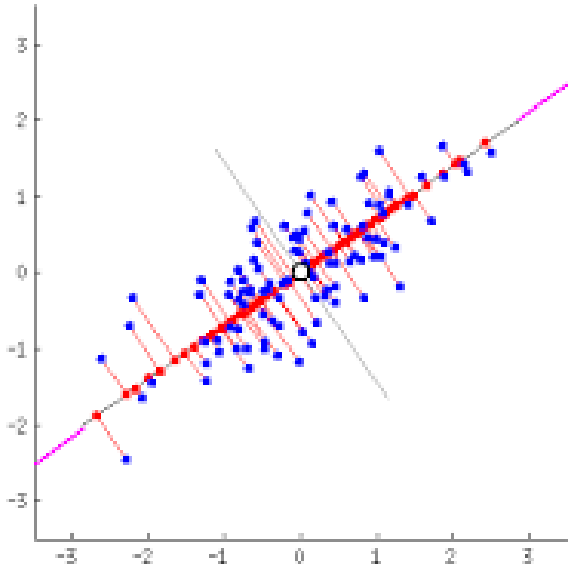
BLEU = 1,46


Dimensionality
Reduction

Dimensionality Reduction: PCA

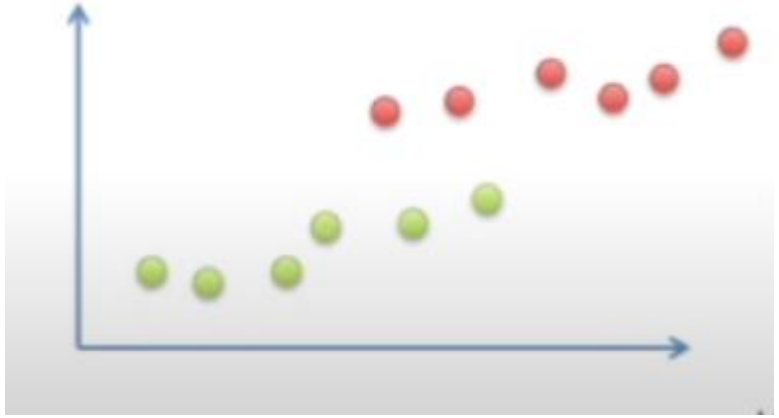
Diagonaliser la matrice de covariance pour sortir les directions principales

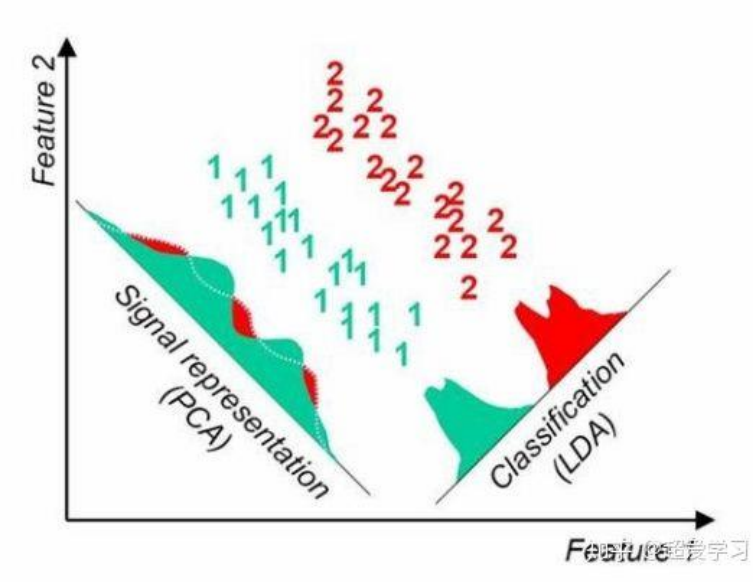
$$
\begin{aligned}
D = P^{-1}AP &= \frac{1}{2}\begin{pmatrix}-1 & 1 & 0\\ 1 & 1 & 2\\ 1 & 1 & 0\end{pmatrix}\begin{pmatrix}0 & -1 & 0\\ -1 & 0 & 0\\ 1 & 1 & 1\end{pmatrix}\begin{pmatrix}-1 & 0 & 1\\ 1 & 0 & 1\\ 0 & 1 & -1\end{pmatrix} \\
&= \frac{1}{2}\begin{pmatrix}-1 & 1 & 0\\ 1 & 1 & 2\\ 1 & 1 & 0\end{pmatrix}\begin{pmatrix}-1 & 0 & 1\\ 1 & 0 & 1\\ 0 & 1 & -1\end{pmatrix} \\
&= \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & -1\end{pmatrix}
\end{aligned}
$$



Dimensionality Reduction: LDA

Maximiser la distance entre les classes
Minimiser la variance intra classe

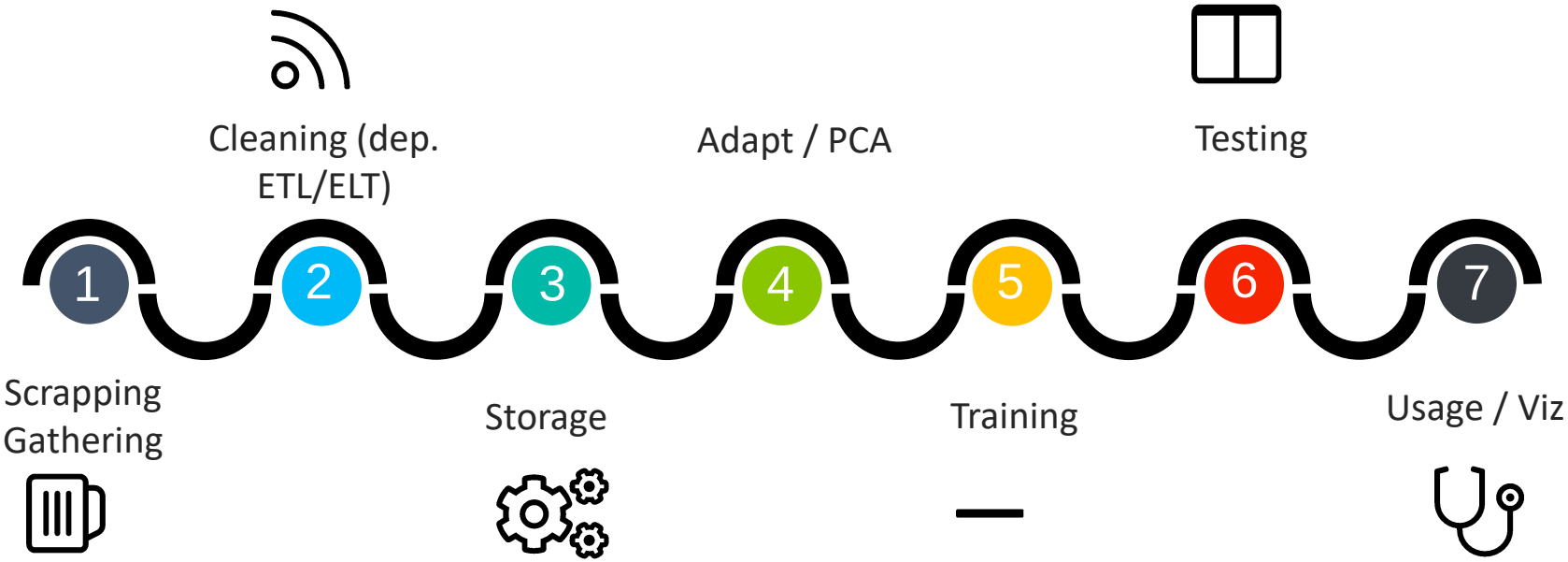





TP 1 (suite) :
Model Selection


Le WorkFlow

Workflow data



Famous Dataset

MNIST : dataset digit numbers
MS-COCO : object detection and segmentation
ImageNet : object classification
Open Image Dataset

IMDB
Sentiment140
WordNet

FreeSpoken Digit dataset
LibriSpeech
VoxCeleb

Kaggle
Data.gouv

Cleaning / Feature Enginering

Cleaning avec Pandas

- Dropna
- Fillna
- Interpolate
- Custom map

Labeliser avec Sklearn

LabelEncoders :

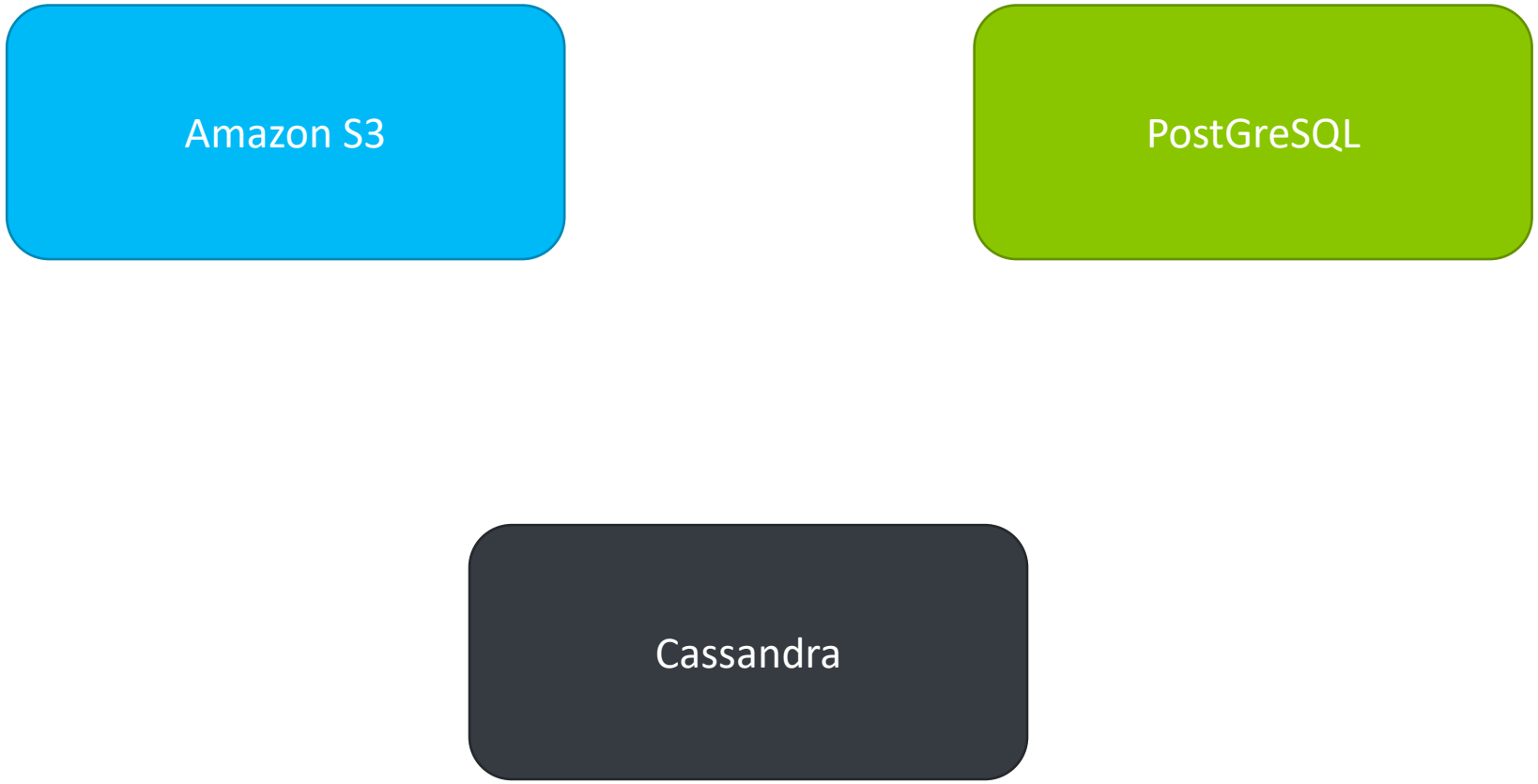
- OneHotEncoder (dummy spread)
- OrdinalEncoder (use order 1 dim)
- LabelEncoder (use for target)
- FeatureHasher
- Cyclic encoding
- KFold

Pre traitement

- Normalization: **MinMaxScaler (and other Scalers)**
- Outlier: **Seaborn boxplot**
- OpenCV equalization
- Signal : Noise Removal


TP 2 :
Nationalité
d’un prénom

TP : Storage



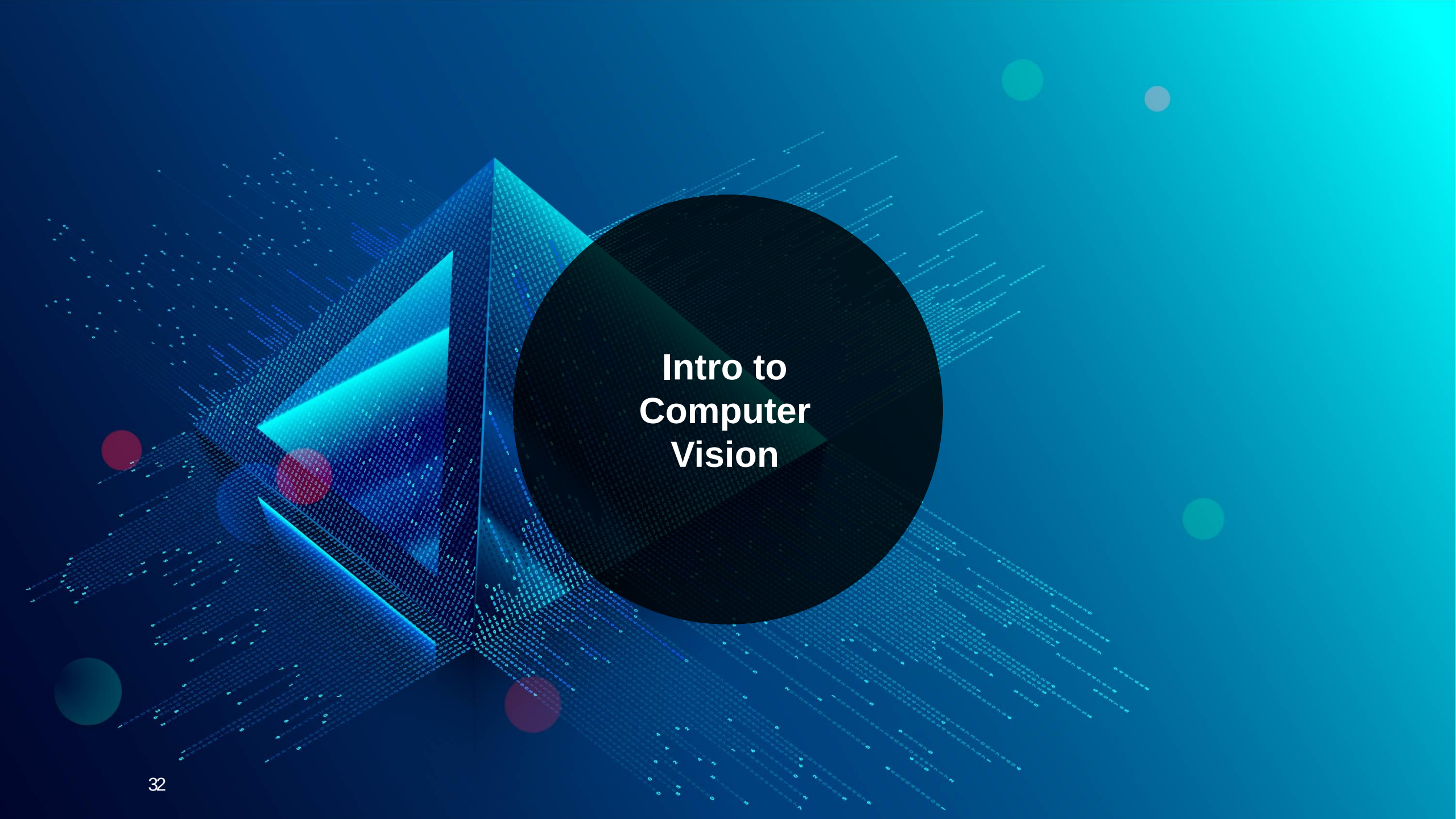

Intro to
Computer
Vision

Image processing : OpenCV

Cleaning
Histogram equalization
Image linear filtering
Morphology (non linear filtering)
Calibration
Color quantization (kmean)

Feature extraction
Spectral Analysis
Edge detection
Segmentation
SIFT, ORB, KAZE, SURF
Optical flow
Object detection/tracking

Keras : Intro to deeplearning

Keras : Intro to deeplearning

Loading data with Keras

```
import numpy as np
import tensorflow as tf
from tensorflow import keras


# Create a dataset.
dataset = keras.preprocessing.image_dataset_from_directory(
  'path/to/main_directory', batch_size=64, image_size=(200, 200))
```

Keras : Intro to deeplearning

Preprocessing with Keras

```
from tensorflow.keras.layers.experimental.preprocessing import CenterCrop
from tensorflow.keras.layers.experimental.preprocessing import Rescaling

# Example image data, with values in the [0, 255] range
training_data = np.random.randint(0, 256, size=(64, 200, 200, 3)).astype("float32")

cropper = CenterCrop(height=150, width=150)
scaler = Rescaling(scale=1.0 / 255)

output_data = scaler(cropper(training_data))
print("shape:", output_data.shape)
print("min:", np.min(output_data))
print("max:", np.max(output_data))
```

Keras : Intro to deeplearning

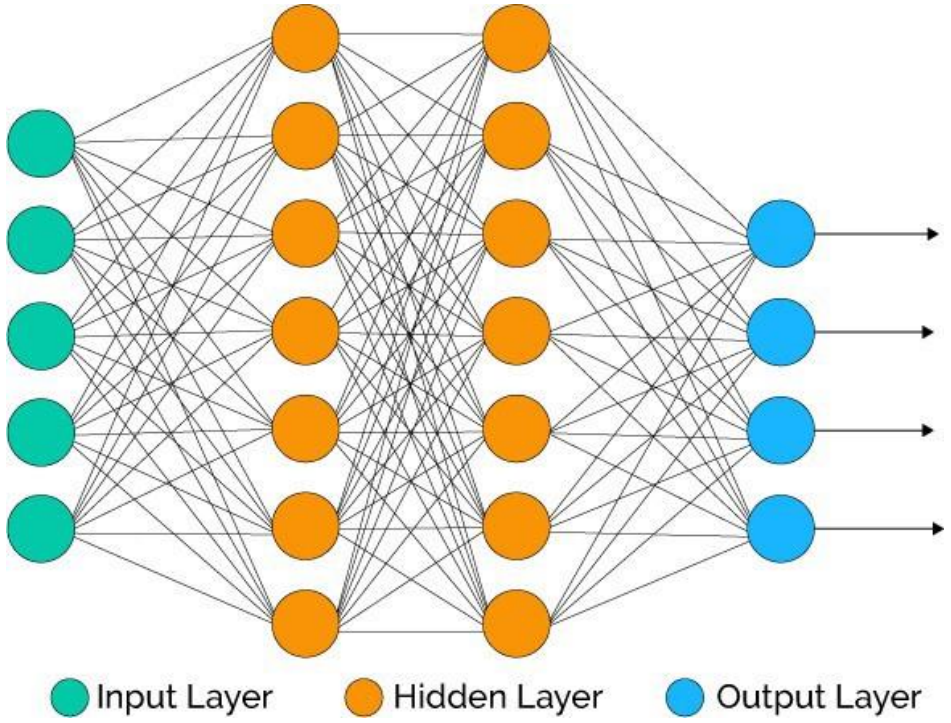
A Keras Model is made of Layers packed together

```
1  from keras.models import Sequential
2
3  model = Sequential()
4  input_layer = Dense(32, input_shape=(8,)) model.add(input_layer)
5  hidden_layer = Dense(64, activation='relu'); model.add(hidden_layer)
6  output_layer = Dense(8)
7  model.add(output_layer)
8
```

Connected sequencially

```
1  from keras.layers import Input
2  from keras.layers import Dense
3  from keras.models import Model
4
5  data = Input(shape=(2,3))
6  layer = Dense(2)(data)
7  model = Model(inputs = data, outputs = layer)
8
```

Connected with functionnal API



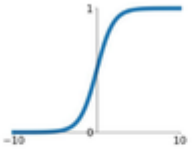
Keras : Intro to deeplearning

Chaque neuronne combine ses entrées pour fournir une sortie par sa fonction d'activation

Activation Functions

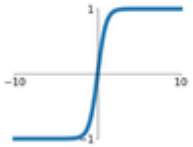
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



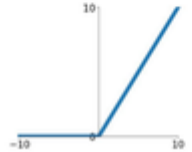
tanh

$\tanh(x)$



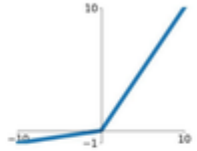
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

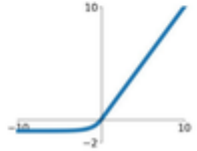


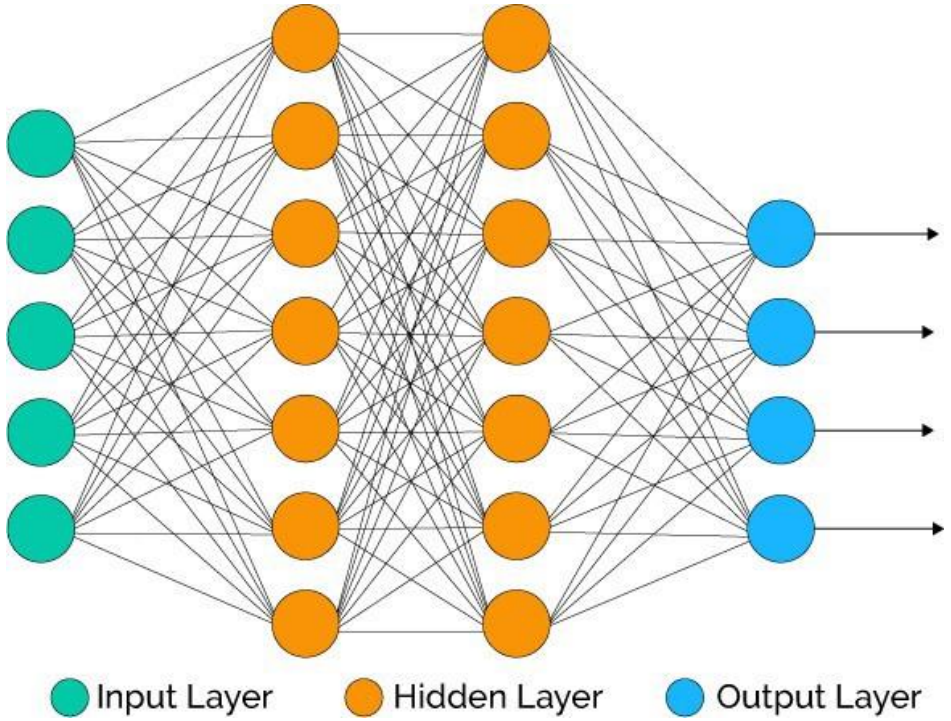
Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

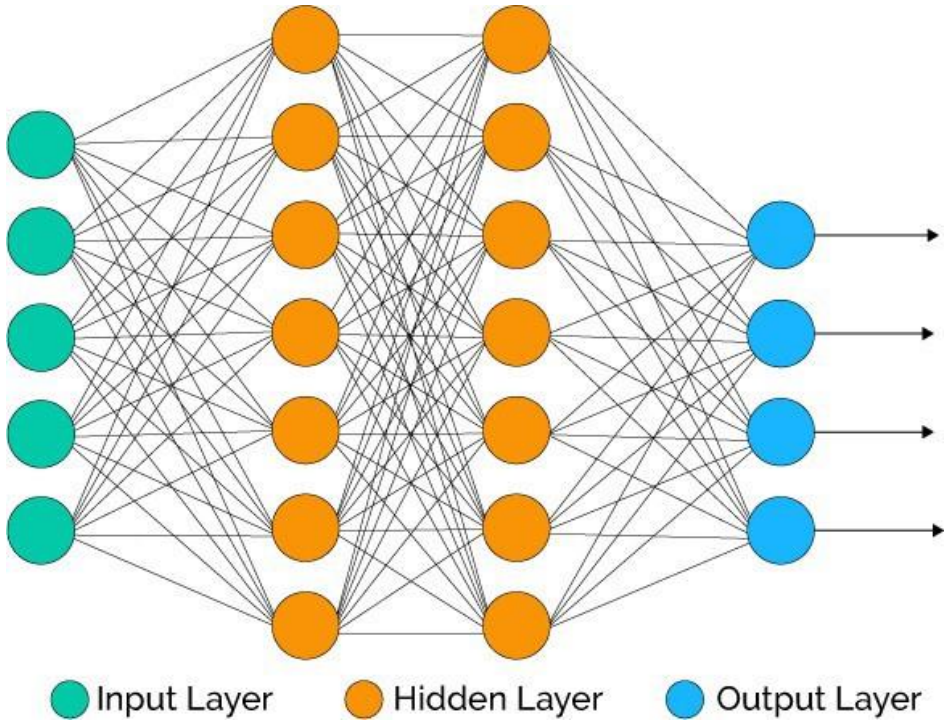




Different types of layer

Input Layer

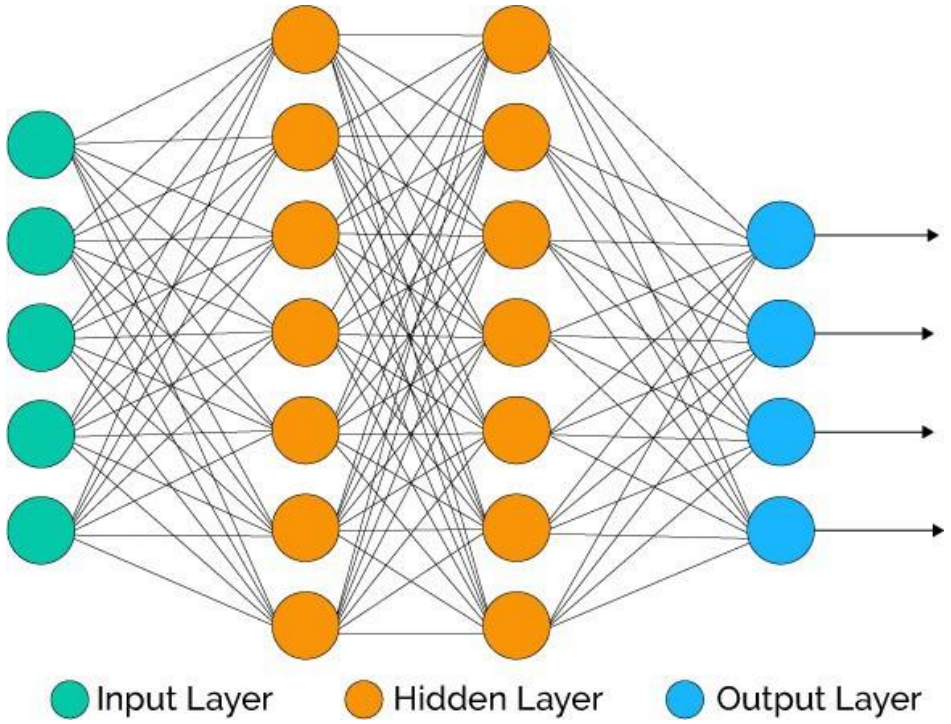
```
tf.keras.Input(
    shape=None,
    batch_size=None,
    name=None,
    dtype=None,
    sparse=False,
    tensor=None,
    ragged=False,
    **kwargs
)
```



Different types of layer

Dense Layer

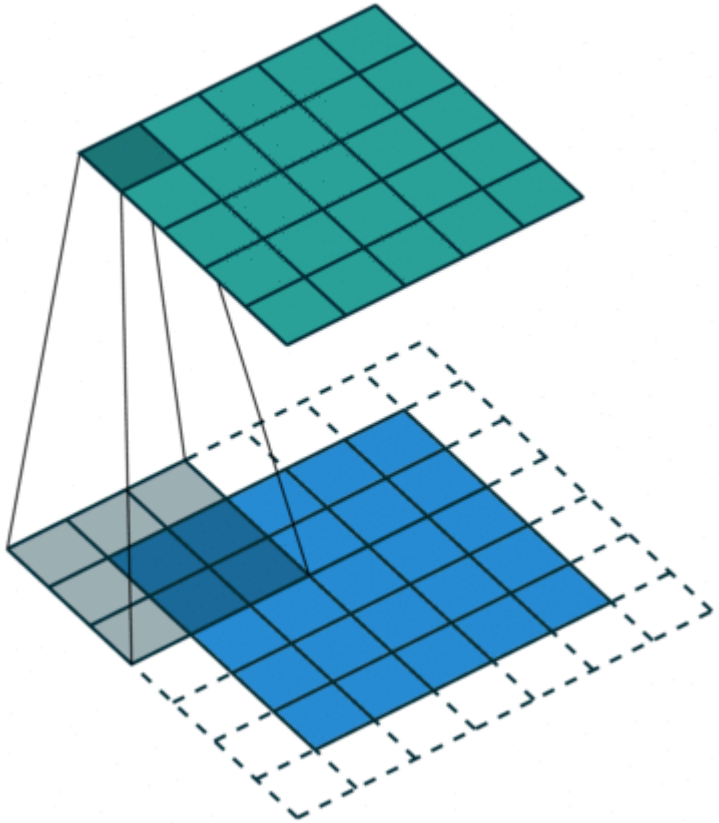
```
tf.keras.layers.Dense(
    units,
    activation=None,
    use_bias=True,
    kernel_initializer="glorot_uniform",
    bias_initializer="zeros",
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None,
    **kwargs
)
```



Different types of layer

Conv2D Layer

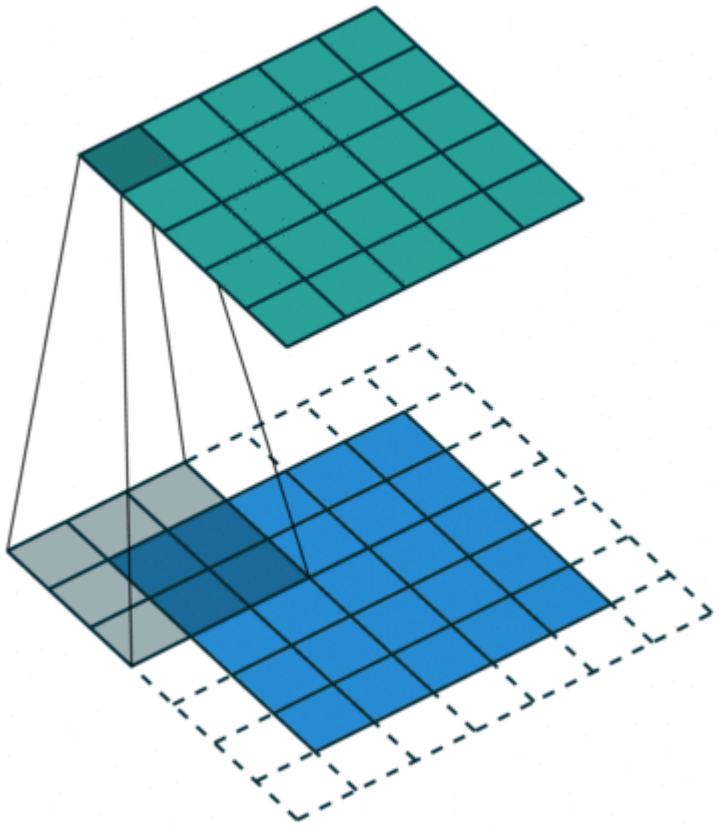
```
tf.keras.layers.Conv2D(
    filters,
    kernel_size,
    strides=(1, 1),
    padding="valid",
    data_format=None,
    dilation_rate=(1, 1),
    activation=None,
    use_bias=True,
    kernel_initializer="glorot_uniform",
    bias_initializer="zeros",
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None,
    **kwargs
)
```



Different types of layer

MaxPooling Layer

```
tf.keras.layers.MaxPooling2D(
    pool_size=(2, 2), strides=None, padding="valid", data_format=None, **kwargs
)
```



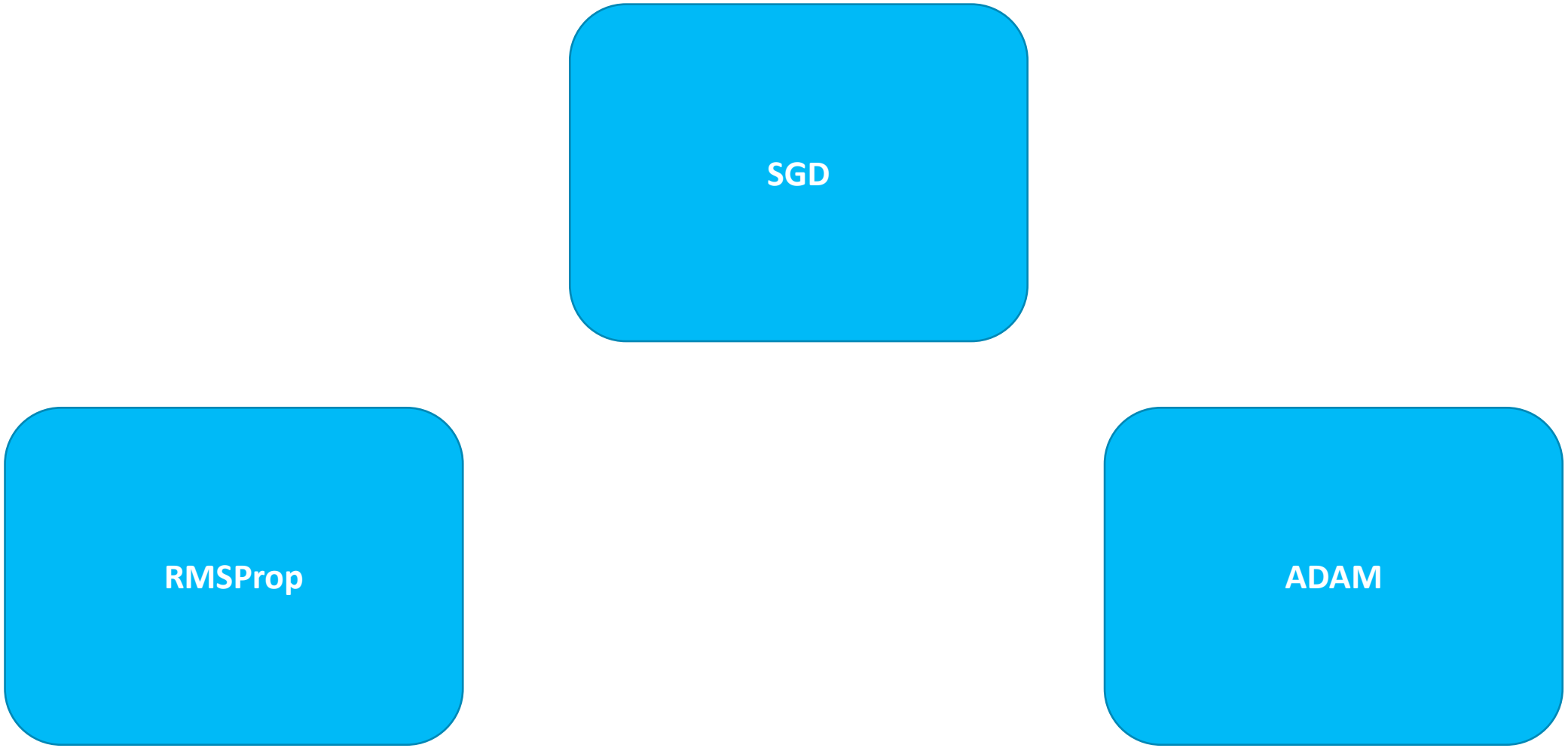
Loss functions

Regression :
Mean_squared_error
MSLE
MAE

Binary classification:
Cross_entropy
Hinge loss
Squared hinge loss

multi classification:
Cross_entropy
Sparse cross entropy
KL Divergence

Optimizer




TP
Dog and Cats

Create your service

AWS Lambda

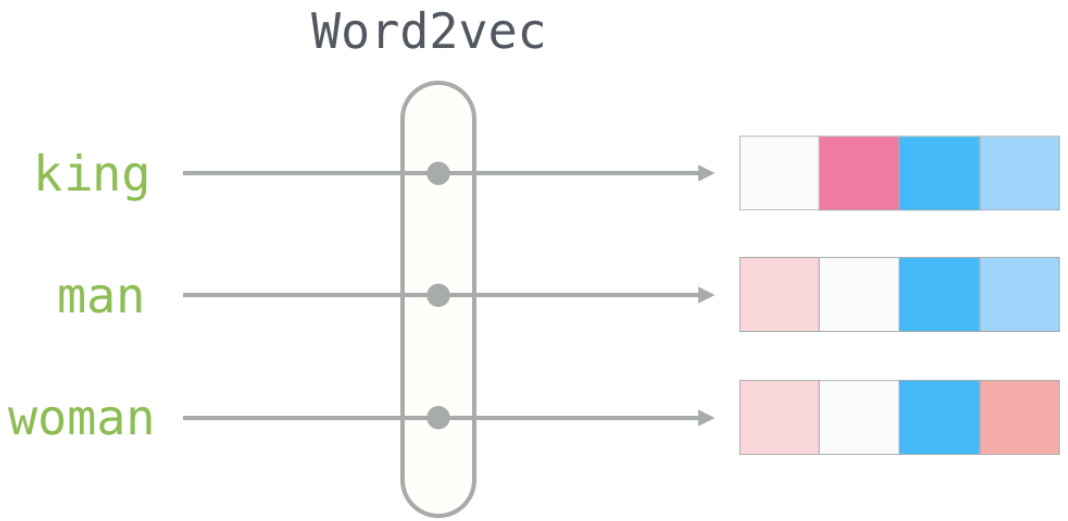
FLask


Intro to
NLP

Intro to NLP

Pre processing classique:
Lemnatisation
Stemming
Stopwords
Vectorization

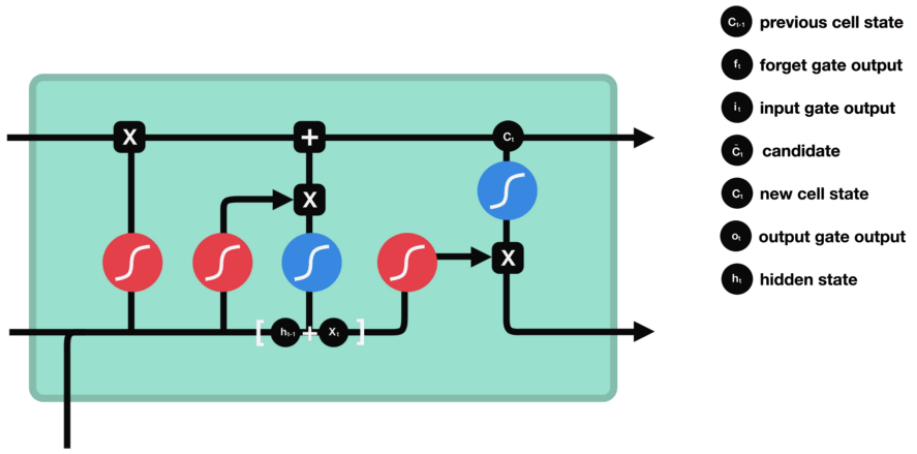
Pre processing NN:
GloVe
BERT
GPT-2



Different types of layer

LSTM Layer

```
tf.keras.layers.LSTM(
    units,
    activation="tanh",
    recurrent_activation="sigmoid",
    use_bias=True,
    kernel_initializer="glorot_uniform",
    recurrent_initializer="orthogonal",
    bias_initializer="zeros",
    unit_forget_bias=True,
    kernel_regularizer=None,
    recurrent_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    recurrent_constraint=None,
    bias_constraint=None,
    dropout=0.0,
    recurrent_dropout=0.0,
    implementation=2,
    return_sequences=False,
    return_state=False,
    go_backwards=False,
    stateful=False,
    time_major=False,
    unroll=False,
    **kwargs
)
```



Transformer Networks

Attention is all you need

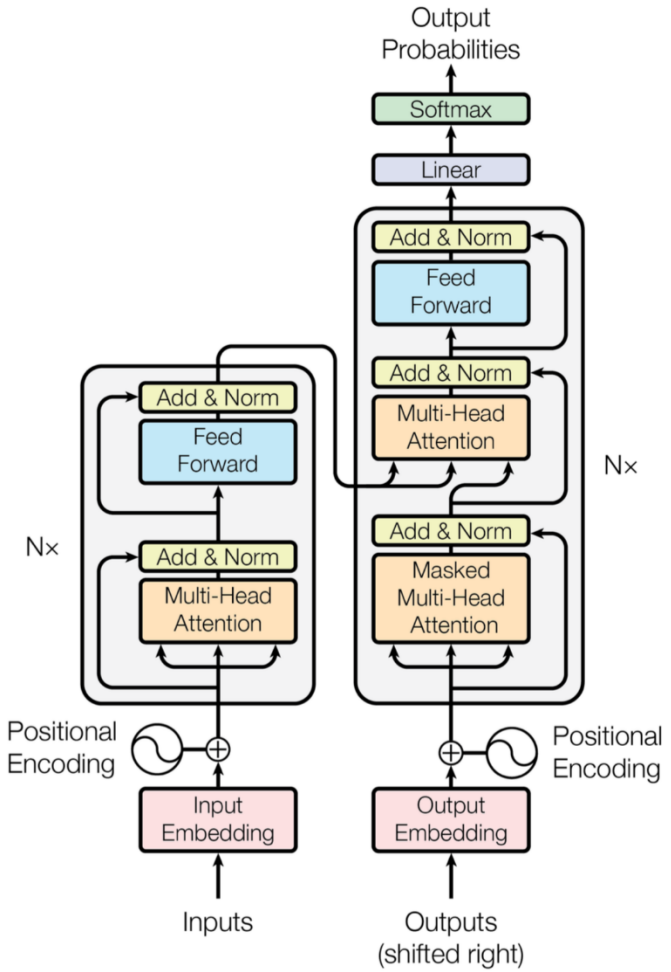
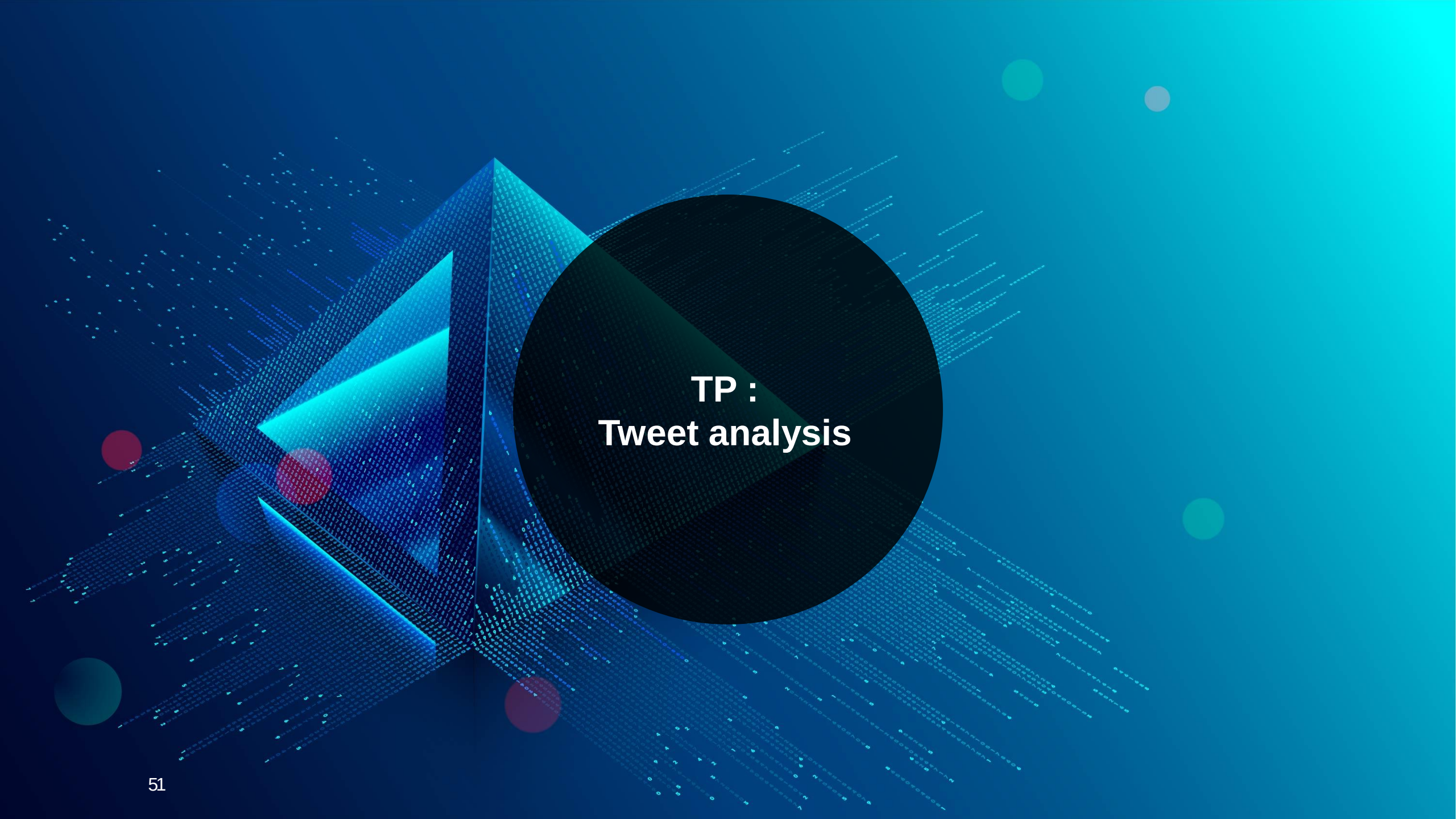


Figure 1: The Transformer - model architecture.


TP :
Tweet analysis