ACADÉMIE
DE NANCY-METZ
Liberté
Égalité
Fraternité

Spécialité Sciences de l'Ingénieur

Découverte de la notion d'intelligence artificielle [IA]

Objectif de l'activité : Comment permettre la reconnaissance de panneaux de signalisation routière à des fins d'identification ?

Introduction

Le principe retenu pour permettre la réalisation de cette activité est l'utilisation d'un outil web permettant de créer des modèles de classifications par réseau de neurones : **Teachable Machine (Google)**

Pour créer son modèle d'apprentissage, il existe plusieurs solutions techniques reconnues telles que : ***TensorFlow***, **PyTorch**, ***Keras, Scikit-learn***… Souvent ces solutions demandent à l'utilisateur de maitriser le langage *Python*, des notions mathématiques avancées ainsi que d'autres connaissances telles que : **les perceptrons**, les **fonctions d'activations**, les **convolutions**, les **descentes de gradient**, les **dense layer**, la **normalisation des données**…

Créer un modèle n'est pas une chose aisée, ainsi, au travers de cette activité, il s'agit simplement de créer un modèle de *machine learning* (modèle supervisé par apprentissage) à l'aide de cet outil-web de classification développé par Google.

Toutefois, nous observerons au travers des différents modèles générés, quels sont les paramètres essentiels que nous pouvons retenir et qui permettent de caractériser la pertinence de notre modèle.

Création du modèle Teachable Machine

La création d'un modèle de classification tel que nous souhaitons le réaliser, s'effectue en 3 étapes :

- **Collecter** des données locales ou dans un cloud (en l'occurrence *Google Drive*) à classifier ;
- **Entrainer** le modèle à partir des données collectées ;
- **Tester** la qualité du modèle.

Nous pourrions rajouter une quatrième étape qui consisterait à **exporter** le modèle afin de pouvoir le ré-exploiter sur la plateforme de notre choix.

- Extrait de la page d'accueil de Teachable Machine -

Lien de l'outil web : https://teachablemachine.withgoogle.com/

1. Étape 1 : Recueillir les données

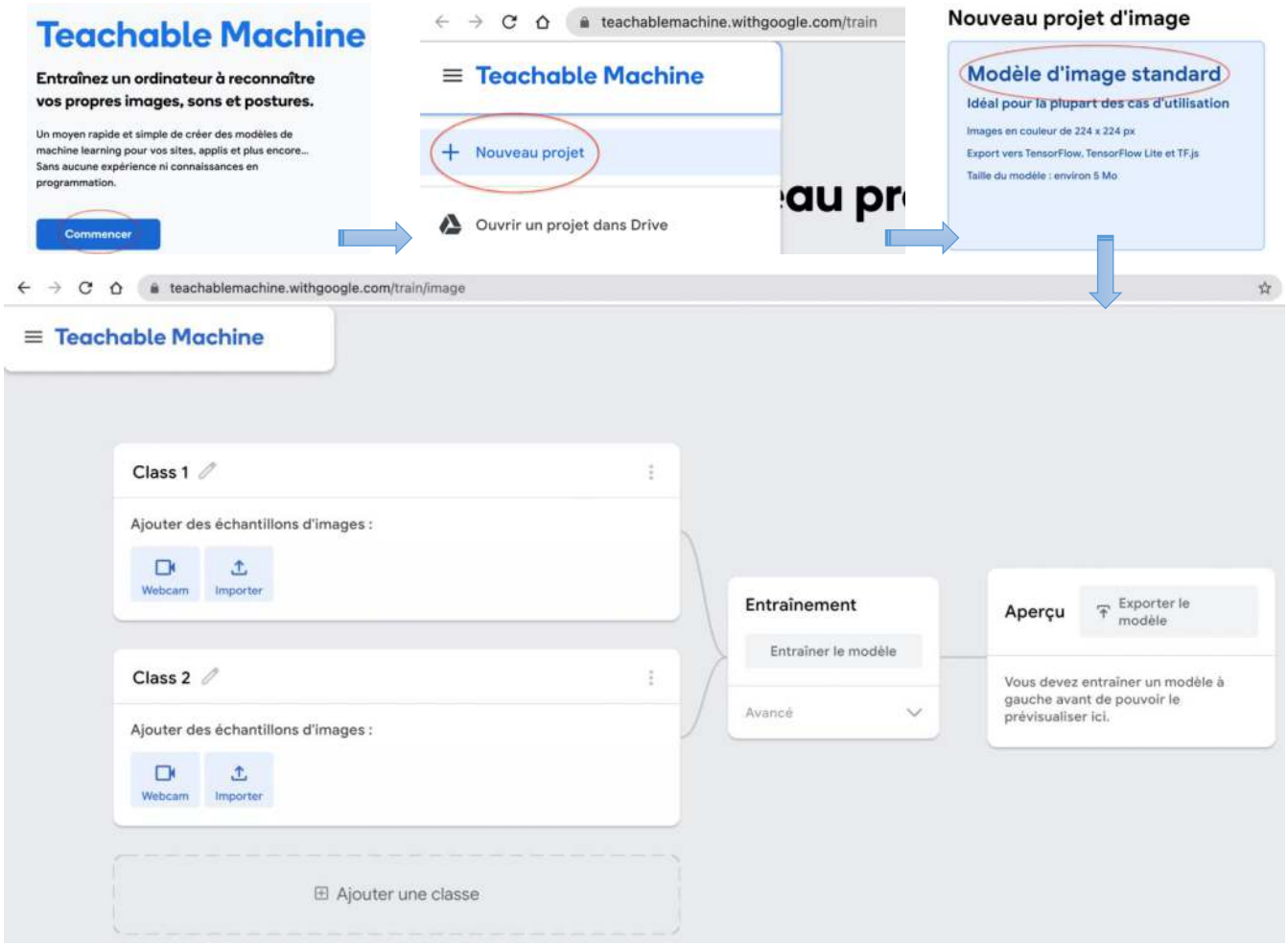
Dans un premier temps nous allons **RECUEILLIR** les données nous permettant de classifier nos différents types de panneaux de signalisation routière.

Nous allons réaliser un modèle d'apprentissage supervisé permettant d'identifier trois classes de signalisation routière :


⇒ Les feux tricolores ;
⇒ Le panneau de signalisation STOP ;
⇒ Le panneau de passage piétons ;

Remarque : Les images que nous allons exploiter pour réaliser notre modèle sont issues du site web **Kaggle**[1] (https://www.kaggle.com). Il s'agit donc d'images de panneau de signalisation provenant de divers pays et donc par définition légèrement différents de ceux que nous pouvons rencontrer dans en France.

A l'aide du site https://teachablemachine.withgoogle.com/ et d'un ordinateur, démarrer un nouveau projet de type « PROJET IMAGES » avec pour choix, un « modèle d'image standard »



Créer maintenant les 3 classes correspondants aux trois types de signalisation routière que l'on souhaite apprendre au modèle. Pour cela, par l'intermédiaire de l'icône en forme de crayon situé à côté de « **Class 1** », modifiez celui-ci afin de lui donner le nom de votre première classe (feux tricolores).

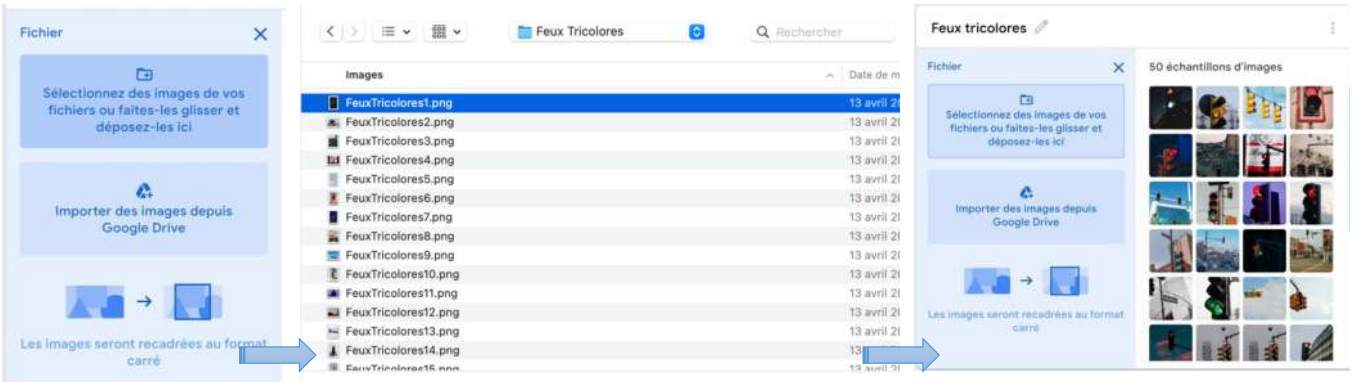
Recommencez l'opération pour les deux autres classes (panneaux *stop* et panneaux *piétons*)

Maintenant que les trois classes sont créées et étiquetées, nous allons maintenant importer le jeu d'images pour chacune de ces trois classes.

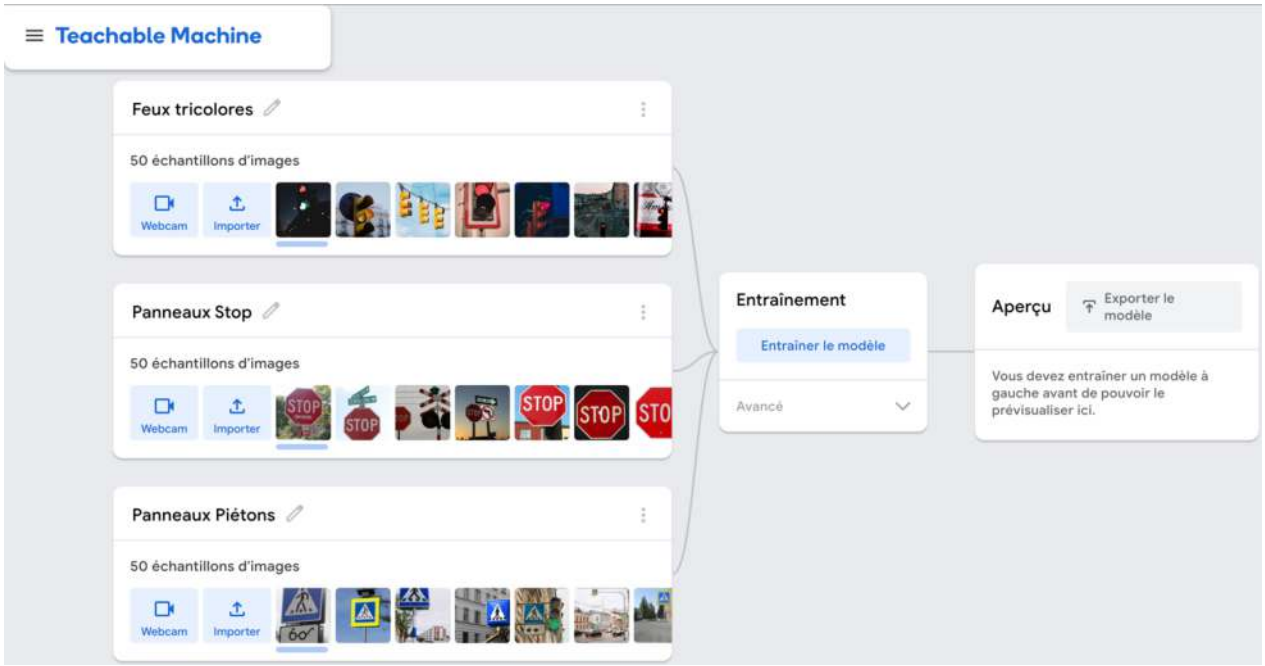
***Kaggle** est une plateforme web organisant des compétitions en science des données. Sur cette plateforme, les entreprises proposent des problèmes en science des données et offrent un prix aux datalogistes obtenant les meilleures performances - Wikipedia -*

Cliquez sur le bouton importer et sélectionnez sur votre ordinateur le dossier correspondant à la classe que vous souhaitez importer.

Sélectionnez dans ce dossier, l'ensemble des 50 images de feux tricolores dont vous disposez.



Répétez l'opération de manière identique pour les deux autres classes (panneaux stop et panneaux piétons)

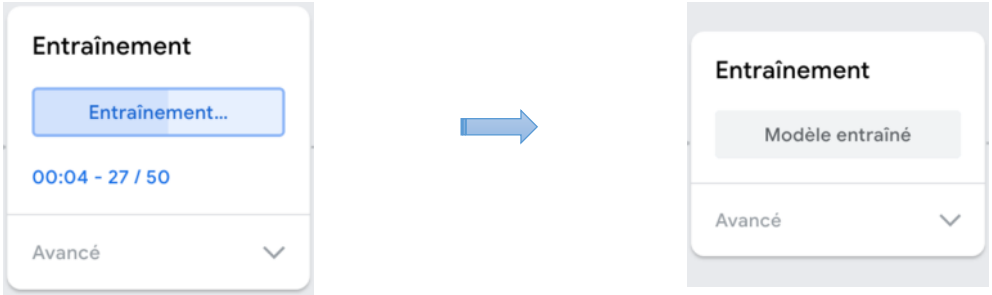


À ce stade de l'activité, vous pouvez enregistrer votre projet de modèle *Teachable Machine* sous la forme d'un fichier stocké sur un support (ordinateur, clé USB…) soit dans le cloud Google-Drive. Ce fichier aura l'extension ***.tm***.

2. Étape 2 : Entrainement du modèle

Une fois les données (images) recueillies et importées dans le modèle, il s'agit de créer le réseau neuronal permettant par la suite la reconnaissance d'un de ces trois panneaux de signalisation routière. Pour cela, il faut **ENTRAINER** le modèle **« Teachable Machine »**.

Cliquez alors sur ***« Entrainer le modèle »***

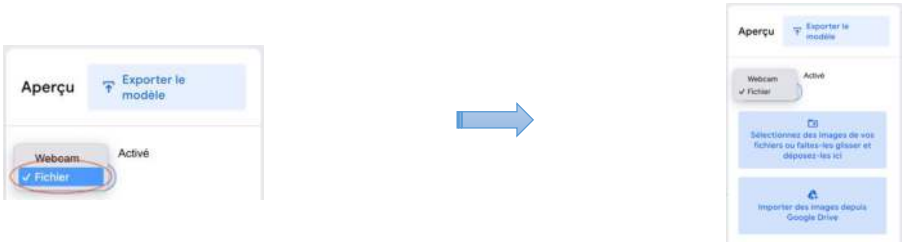


Validation du modèle Teachable Machine

1. Vérifier la validité du modèle

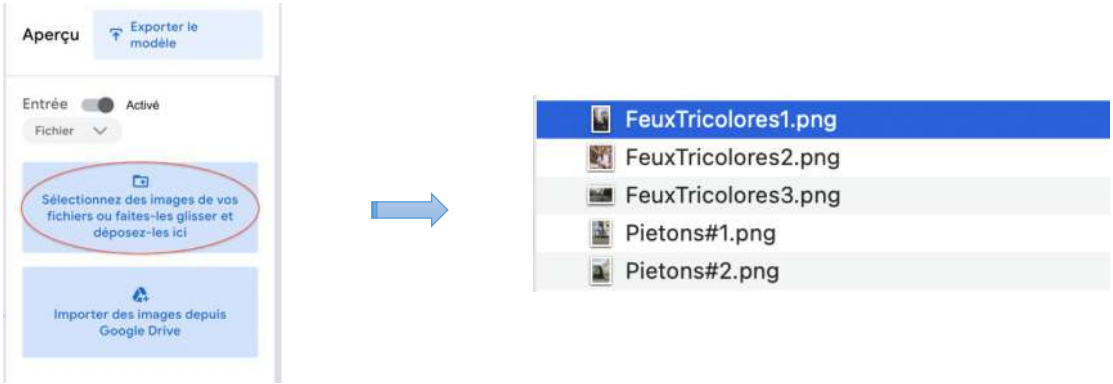
Lors de cette dernière étape, nous allons vérifier comment notre modèle va classifier les différentes images que nous allons lui présenter. La vérification peut s'effectuer par transfert d'image (locale ou depuis *Google Drive*) ou en utilisant sa webcam.

Sélectionnez dans **« Aperçu »**, l'entrée de vérification du modèle à partir de **« fichier »**



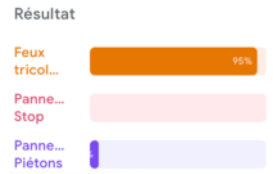
1.1. Test du modèle : Feux tricolores

Sélectionnez les images « tests » des feux tricolores une à une depuis le dossier « test » et importer les dans le modèle (vous pouvez procéder également par glisser déposer)



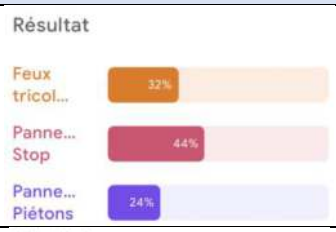
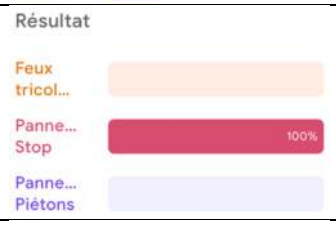
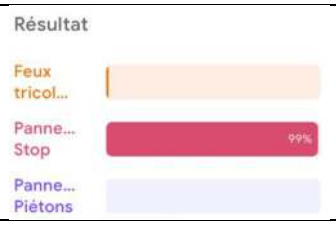
Notez la probabilité (calculée par le modèle) que chacune de vos images appartiennent à la classe « Feux tricolores »

Images	Probabilité d'appartenance à la classe
	

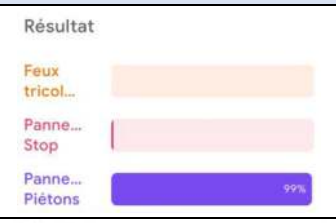
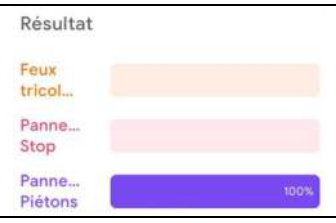
1.2. Test du modèle : Panneaux stop

En procédant de la même manière, indiquez la probabilité d'appartenance de chaque image « test » à la classe « Panneaux stop ».

Images	Probabilité d'appartenance à la classe
	
	
	

1.3. Test du modèle : Panneaux piétons

En procédant de la même manière, indiquez la probabilité d'appartenance de chaque image « test » à la classe « Panneaux piétons ».

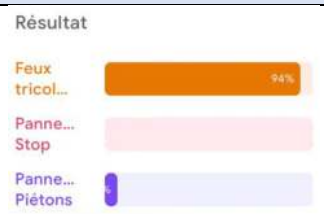
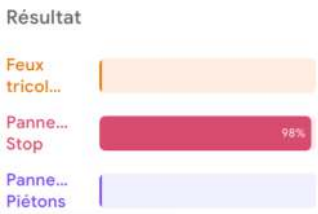
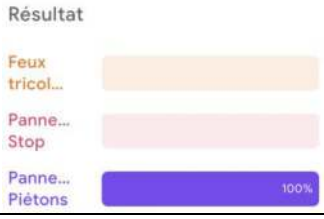
Images	Probabilité d'appartenance à la classe
	
	

2. Vérifier la robustesse du modèle

Mais qu'en est-il lorsque le modèle est confronté à des images qui mêlent des situations plus complexes comme des panneaux de signalisation routières multiples.

À partir des images de tests suivantes, indiquez la probabilité d'appartenance de l'image à sa propre classe.

Images	Probabilité d'appartenance à la classe
	
	
	

2.1.1. Analyse succincte

Indiquez selon vous quels sont les motifs pour que le modèle puisse effectuer des prédictions qui ne soient par correctes notamment pour la seconde image « test »

Nous pouvons imaginer que les panneaux de signalisation de limitation (80 km/h) et d'arrêt interdit ont provoqué une confusion avec le panneau de signalisation Stop, notamment avec la présence du rouge tandis que le bleu du panneau piéton a pu être confondu avec le bleu du ciel.

2.1.2. Constatation

Vous avez pu remarquer que des différences de prédiction de classification avaient lieu pour une même photo. Il suffit de relancer l'apprentissage du modèle pour se rendre compte que pour une même image la prédiction est de nouveau différente.

En effet, l'apprentissage du modèle ne s'effectue pas toujours dans le même ordre d'analyse des images, de plus le jeu prélevé pour l'apprentissage (85 % des images totales) et le jeu prélevé pour le test du modèle (les 15 % restant) ne sont pas toujours les mêmes.

Analyse plus approfondie du modèle.

Afin d'effectuer une analyse plus approfondie de notre modèle, nous allons observer un certain nombre de paramètres caractéristiques. Pour avoir accès à ces dernières, il est nécessaire de les faire apparaitre en cliquant sur le bouton « **Options avancées** »

L'appui sur ce bouton permet de faire apparaitre des graphiques permettant de mieux comprendre le fonctionnement du modèle.

L'apprentissage du modèle précédent a permis de révéler les caractéristiques suivantes :

1. Indice de performance des classifieurs

1.1. La matrice de confusion (*confusion matrix*)

Les possibilités pour un modèle de classification sont au nombre de quatre :

- Faux positif (*False positive*) : Par exemple, on classifie un patient comme bien portant alors qu'il est bien malade. Correspond à une erreur de type I.
- Faux négatif (*False negative*) : Par exemple, on classifie un patient comme bien malade alors qu'il est bien portant. Correspond à une erreur de type II.

- Vrai positif (*True positive*) : Par exemple, on classifie un patient comme malade et il est effectivement malade.
- Vrai négatif (*True negative*) : on classifie un patient comme bien portant et il est bien portant.

Ces possibilités sont traditionnellement représentées dans une matrice de confusion :

-	**Prédit : vrai**	**Prédit : faux**
Réel : vrai	Vrai positif	Faux négatif (erreur de type I)
Réel : faux	Faux positif (erreur de type II)	Vrai négatif

1.2. La justesse (*accuracy*)

On dérive de la matrice de confusion trois mesures : **la justesse** (*accuracy*), la précision et le rappel. Chaque mesure est importante pour saisir l'efficacité d'un modèle de classification. Nous ne détaillerons ici que la justesse.

La justesse désigne la part de prédictions correctes sur l'ensemble des prédictions réalisées par le modèle :

$$Justesse = \frac{VP + VN}{VP + VN + FP + FN}$$

La justesse permet d'identifier un classifieur trop aléatoire. Elle ne reflète toutefois pas la réalité lorsque les classes sont déséquilibrées. Il existe en effet de nombreux cas où l'évènement à détecter est rare et la précision seule du classifieur n'est pas suffisante.

1.3. La fonction perte (*Loss Function*)

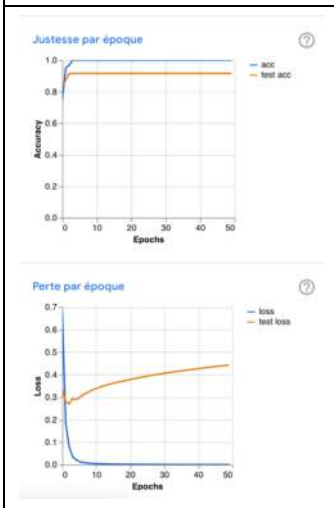
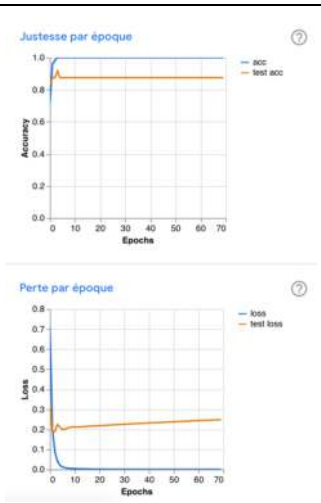
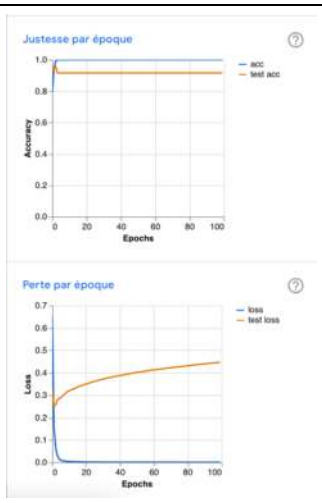
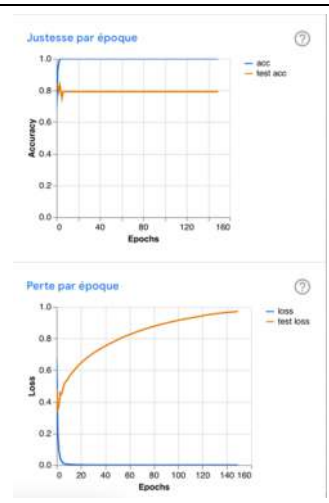
Une **fonction de perte**, ou *Loss function*, est une fonction qui évalue l'écart entre les prédictions réalisées par le réseau de neurones et les valeurs réelles des observations utilisées pendant l'apprentissage. Plus le résultat de cette fonction est minimisé, plus le réseau de neurones est performant. Sa minimisation, c'est-à-dire réduire au minimum l'écart entre la valeur prédite et la valeur réelle pour une observation donnée, se fait en ajustant les différents poids du réseau de neurones.

Il existe de très nombreuses fonctions de perte disponibles (domaines des mathématiques et de la statistique), toute la difficulté est d'être en mesure de choisir la fonction de perte la plus adaptée à son problème d'intelligence artificielle (régression, classification, clustering…)

2. Problème particulier de l'overfitting

Pour révéler un problème très souvent présent en machine learning, il suffit d'effectuer à plusieurs reprises l'entraînement du modèle en augmentant à chaque entraînement le nombre d'époque (passe complète de tous les échantillons d'entraînement)

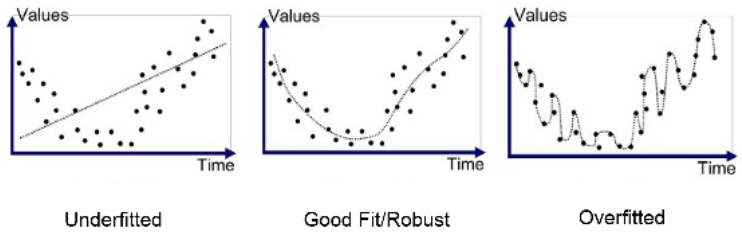
L'exemple ci-dessous présente quatre entraînements du modèle ***« Teachable Machine »*** en augmentant systématiquement pour chacun des cas le nombre d'époque (50, 70, 100 et 150).

Époques = 50	Époques = 70	Époques = 100	Époques = 150
			
			
			

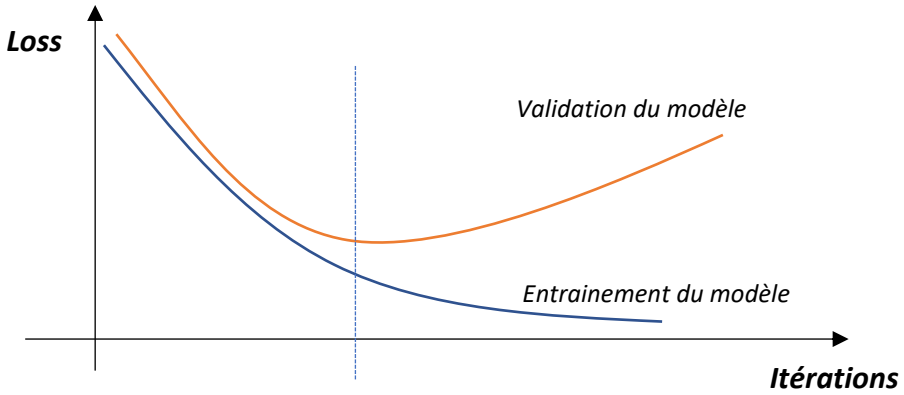
L'overfitting est un problème qui est souvent rencontré en **machine learning**. Il survient lorsque notre modèle essaye de trop coller aux données d'entraînement.

En data science la qualité des données que vous exploitez est primordiales, malheureusement les données parfaites n'existent pas. Il y a toujours du bruit et des imprécisions. Ainsi, un modèle fait de l'**overfitting** lorsqu'il commence à apprendre ce bruit. Il en résulte un modèle biaisé qui est alors impossible à **généraliser**.

L'image ci-dessous, illustre très bien le problème de l'overfitting :



En pratique, un modèle qui *overfit* est souvent très facile à détecter. **L'overfitting** intervient lorsque l'erreur sur les données de test devient croissante. Typiquement, si l'erreur sur les données d'entraînement est beaucoup plus faible que celle sur les données de test, c'est sans doute que votre modèle a sur-appris les données.



Il existe bien évidemment des techniques pour éviter l'overfitting (validation croisée, ajout de données d'entrainement, sélection des *features*…) mais celles-ci sortent du cadre de cette initiation.

Webographie

KAGGLE.COM. *Your home for data Science [en ligne].* Kaggle Inc, 2020. Disponible sur : https://www.kaggle.com (consulté le 12 novembre 2021)

ARBORETUM.LINK. *Indicateurs de performance des classificateurs [en ligne].* Mon carnet de note virtuel, Licence MIT. Disponible sur : https://www.arboretum.link/notes/indicateurs-de-performance-des-classificateurs (consulté le 5 janvier 2022)

LAREVUEIA.FR. *7 méthodes pour éviter l'overfitting [en ligne].* La revue IA, 2020. Disponible sur : https://larevueia.fr/7-methodes-pour-eviter-loverfitting/ (consulté le 3 novembre 2021)

EXPERIMENTS.WITHGOOGLE.COM. *Experience with google [en ligne].* Google Inc. Disponible sur : *https://experiments.withgoogle.com (consulté le 7 novembre 2021)*

Glossaire [1]

Teachable Machine divise vos échantillons en deux buckets, d'où les deux libellés « Entraînement » et « Test » dans les graphiques ci-dessous.

- **Échantillons d'entraînement** (85% des échantillons) : ils sont utilisés pour entrainer le modèle à classifier correctement les nouveaux échantillons dans les classes que vous avez créées.
- **Échantillons de test** (15% des échantillons) : ils ne sont jamais utilisés pour entraîner le modèle. Ils servent à vérifier les performances du modèle avec de toutes nouvelles données après qu'il a été entraîné avec les échantillons d'entraînement.
- **Sous-apprentissage** : on parle de sous-apprentissage lorsqu'un modèle classifie mal les échantillons d'apprentissage en raison de leur complexité.
- **Surapprentissage** : on parle de sur apprentissage lorsqu'un modèle apprend à classifier les échantillons d'entraînement avec une telle précision qu'il ne parvient plus à classifier correctement les échantillons de test.
- **Époques** : une époque est une passe complète sur tous les échantillons d'entraînement (le modèle les a tous ingérés au moins une fois). Par exemple, si vous avez défini 50 époques, le modèle que vous entraînez fera 50 passes sur l'ensemble de données d'entrainement complet.

[1] Disponible sur Teachable Machine : https://teachablemachine.withgoogle.com puis options avancées / vocabulaire