Projet Travaux Pratiques IA

Projet «Wines quality»

Auteurs

Benjamin NIDDAM
Nicolas GOUWY

3 Avril 2022

Table des matières

0.1 Introduction

L'intelligence artificielle (IA) est un processus d'imitation de l'intelligence humaine qui repose sur la création et l'application d'algorithmes exécutés dans un environnement informatique dynamique. Son but est de permettre à des ordinateurs de penser et d'agir comme des êtres humains.

Pour y parvenir, trois composants sont nécessaires :
— Des systèmes informatiques
— Des données avec des systèmes de gestion
— Des algorithmes d'IA avancés (code)

Pour se rapprocher le plus possible du comportement humain, l'intelligence artificielle a besoin d'une quantité de données et d'une capacité de traitement élevées.

Depuis au moins le premier siècle avant notre ère, l'Homme s'est penché sur la création de machines capables d'imiter le raisonnement humain. Le terme « intelligence artificielle » a été créé plus récemment, en 1955 par John McCarthy. En 1956, John McCarthy et ses collaborateurs ont organisé une conférence intitulée « Dartmouth Summer Research Project on Artificial Intelligence » qui a donné naissance au machine learning, au deep learning, aux analyses prédictives et, depuis peu, aux analyses prescriptives. Un nouveau domaine d'étude est également apparu : la science des données.

0.2 Méthodes d'évaluation

Nous utiliserons le langage de programmation Python pour les deux méthodes.

0.2.1 Régression par k plus proches voisins

L'algorithme des K plus proches voisins (KNN) est un algorithme qui compare directement l'élément test à une base d'éléments connus sans apprentissage préalable. Cela en fait une solution moins rapide qu'un réseau de neurones car il nécessite de parcourir les éléments connus à chaque test.

Pour chaque vin de notre base, on calcule la distance moyenne quadratique de ses attributs avec le vin à tester. On garde ensuite les K voisins avec les distances les plus faibles. Dans nôtre cas, on sépare la base en deux sous-bases distinctes :

— Une base d'éléments connus à utiliser pour appliquer l'algorithme.
— Une base d'éléments test pour mesurer l'efficacité de l'algorithme.

Pour un nombre K de voisins allant de 1 à 20, on mesure :

— L'erreur moyenne, 0.6 équivalant à un décalage moyen de 0.6 de qualité prédite avec la qualité connue.
— L'accuracy, correspondant au pourcentage de tests dont la qualité prédite correspond à la qualité connue.

On peut voir que l'accuracy pour 1 voisin est de 54.5% et cette dernière diminue si on augmente le nombre de voisins. Cela parait peut efficace car l'algorithme n'a raison qu'un peu plus d'une fois sur deux. Cependant, il s'agit d'un problème de régression. Que l'algorithme se trompe, c'est une chose, mais est-il éloigné de la vérité pour autant ? On peut voir que l'erreur moyenne pour 1 voisin est de moins de 0.6. La qualité d'un vin sera donc notéé en moyenne 0.6 en dessous ou au dessus de ce qui est attendu, ce qui est peu. Comme pour l'accuracy, on observe qu'avec l'augmentation du nombre de voisins on obtient de moins bons résultats, l'erreur étant de plus en plus grande.

Cette solution, en restant sur 1 voisin seulement, reste donc une option plutôt satisfaisante.

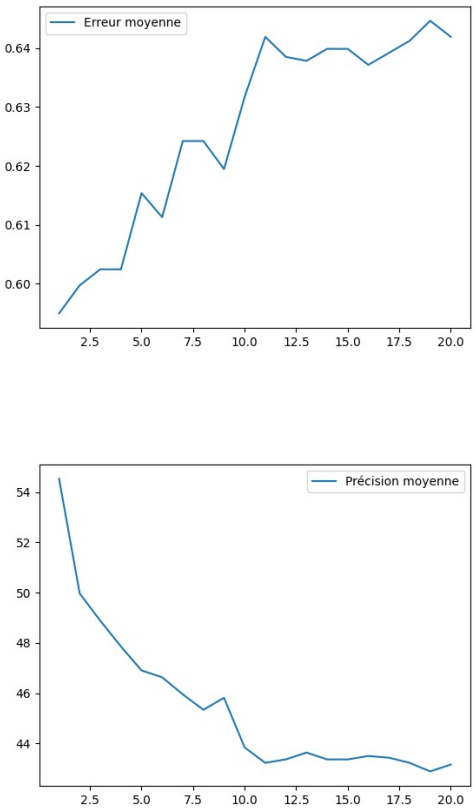


FIGURE 1 – Résultats de l'algorithme KNN

0.2.2 Apprentissage automatique

Dans un second temps, nous avons mis en place un système d'apprentissage automatique. C'est un algorithme de classification qui permet de déterminer la classe d'un objet à partir de ses données. Il est basé sur la notion de probabilité.

Tout d'abord nous importerons les bibliothèques nécessaires :
— 'sklearn' pour les algorithmes d'apprentissage automatique
— 'numpy' pour les opérations mathématiques
— 'pandas' pour les opérations sur les données
— 'matplotlib' pour les graphiques

Ensuite nous allons importer et séparer les données :
À partir d'un échantillon de population qui représente nos données, on répartit les données en deux groupes, les données d'entraînement et les données de test. La première catégorie de données servira pendant la phase d'apprentissage du modèle alors que le second sera utilisé pour évaluer la qualité de prédiction du modèle. Le but n'est donc pas de construire une fonction qui prédira avec une précision optimale les valeurs des variables cibles mais une fonction qui se généralisera au mieux pour prédire des valeurs de données qui n'ont pas encore été observées.

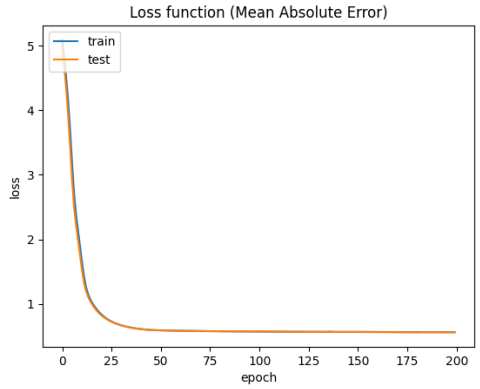
0.2.3 Régression

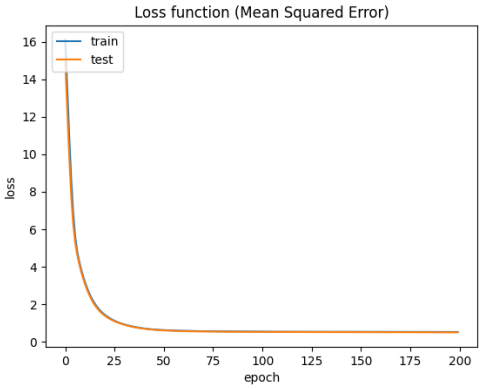
Ici, pour déterminer la qualité du vin, nous allons mettre en place une régression. Il s'agit d'un algorithme d'apprentissage supervisé c'est-à-dire qu'à partir de la variable cible ou de la variable à expliquer (Y), le modèle a pour but de faire une prédiction grâce à des variables dites explicatives (X) ou prédictives.

Nous avons tout d'abord défini le premier modèle. Il est composé de 4 couches composées respectivement de 11 neurones en entrée, de 4 neurones chacunes pour les 2em et 3em couches et 10 neurone pour la couche de sortie.

Avec cette configuration, nous avons défini un modèle de régression et obtenu les résultats suivants :

On observe que les 2 courbes sont parfaitement superposées ce qui pourrait signifiee que notre réseau de neurones ne fait aucune erreur. Mais en réalité, nous





sommmes dans une situation de sous apprentissage. Pour y remédier, nous allons complexifier le modèle en augmentant le nombre de neurones par couches.

On passe donc à la configuration suivante : 4 couches avec respectivement 11 neurones en entrée, 8 neurones sur les 2 couches de traitement et 1 neurone de sortie.

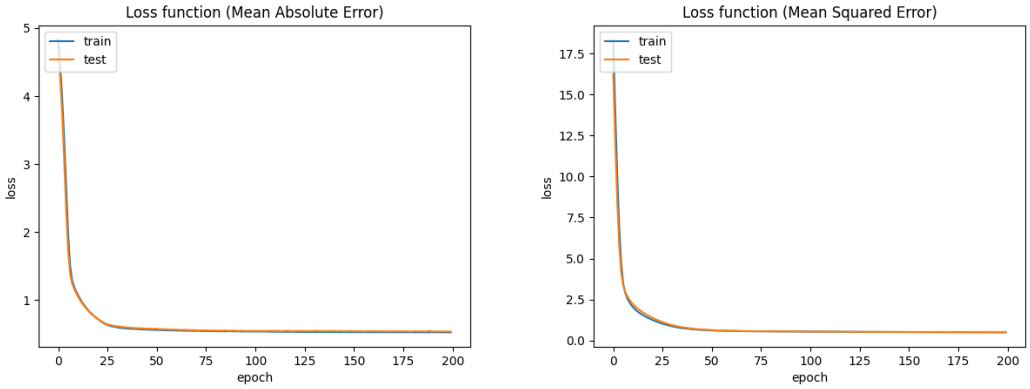


FIGURE 2 – 11-8-8-1

On observe tout comme précédemment que les 2 courbes sont parfaitement superposées ce qui signifie que notre réseau de neurones est toujours en sous apprentissage. Nous avons donc essayé de complexifier progressivement le modèle. En effet, nous avons respectivement continué à augmenté le nombre de neurones par couches jusqu'à atteindre la configuration suivante : 11-20-20-1 puis ensuite augmenté ne nombre de couches tout en continuant à varier le nombre de neurones par couches. Voici quelques exemples des résultats obtenus :

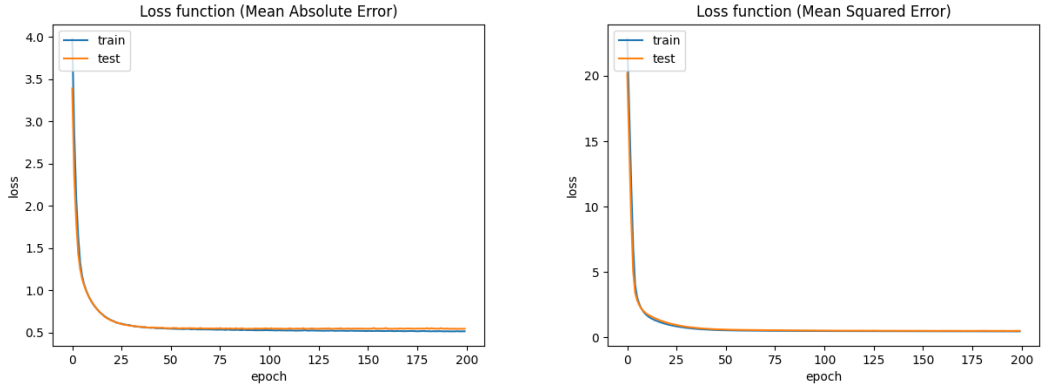


FIGURE 3 – 11-12-12-1

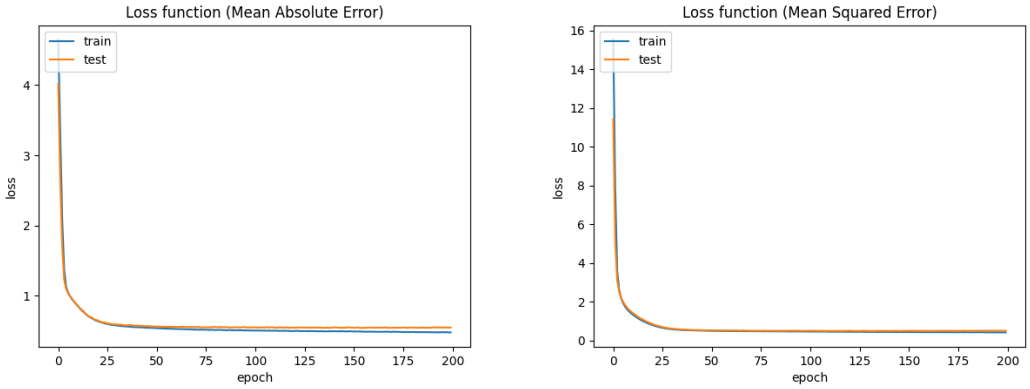


FIGURE 4 – 11-20-20-1

On observe donc que jusqu'à 2 couches de 20 neurones, notre modèle reste en sous apprentissage et commence tout juste à généraliser. Ce n'est donc surement pas une bonne solution d'augmenter indéfiniment le nombre de neurones par couches.

Nous avons donc dessidé d'ajouter une troisième couche de neurones pour le traitement de l'information. Cette dernière contient le même nombre de neurones les 2 couches précédentes. C'est avec cette nouvelle configuration que nous obtenons les résultats suivants pour les nombres de neurones de 8, 12 et 20 :

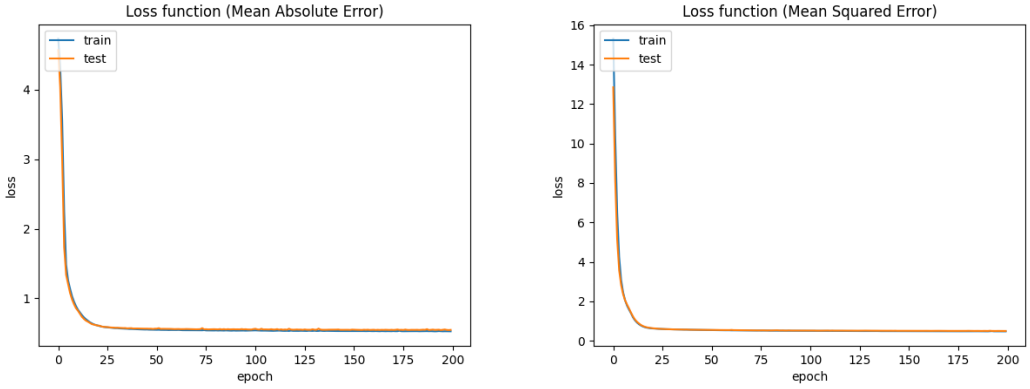


FIGURE 5 – 11-8-8-8-1

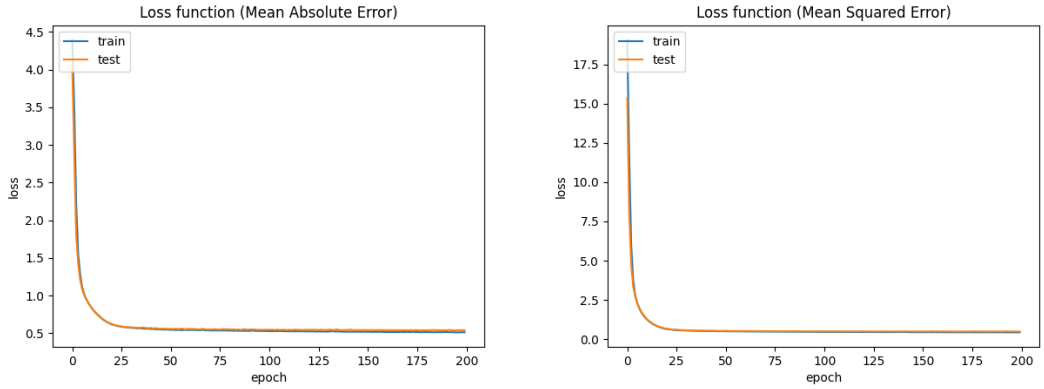


FIGURE 6 – 11-12-12-12-1

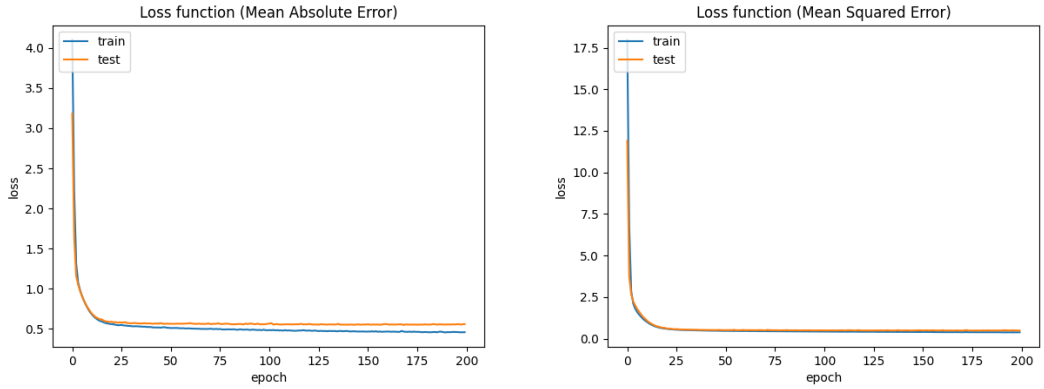


FIGURE 7 – 11-20-20-20-1

Malgré tout notre modèle peine à généraliser. Nous aurions pu continuer à ajouter des couches et des neurones pour tenter d'augmenter la précision du modèle. Mais nous avons cependant opté pour une réduction du nombre de caractéristiques en entrée. Pour ce faire, nous avons tracé des hisotogrammes pour représentant à chaque fois les valeurs prises par une caractéristique afin de déterminer si cette dernière (la caractéristique) est importante dans la prédiction du résultat. Nous avons donc choisi de retirer les caractéristiques suivantes :

— le taux d'alcool
— la densité
— le dioxyde de soufre libre

Et obtenons les histogrammes suivants :

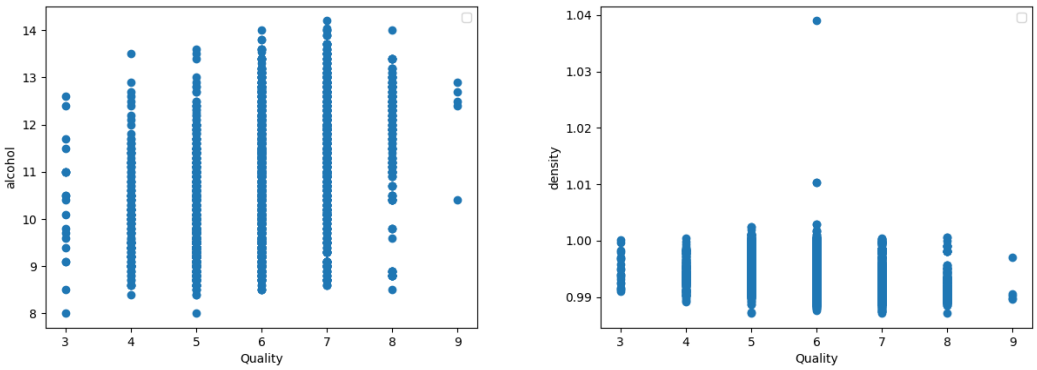


FIGURE 8 – Histogrammes de l'alcool et de la densité

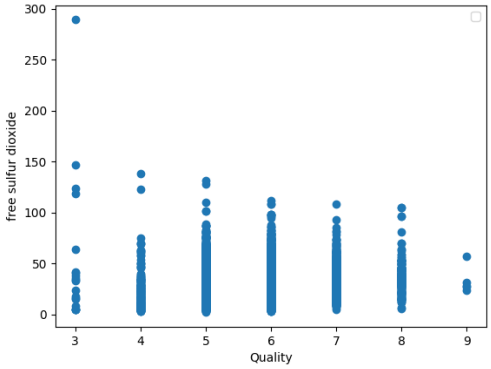


FIGURE 9 – histogramme du dioxyde de soufre libre

Celà fait, nous avons retenté les mêmes expériences que précédemment, sans succès.

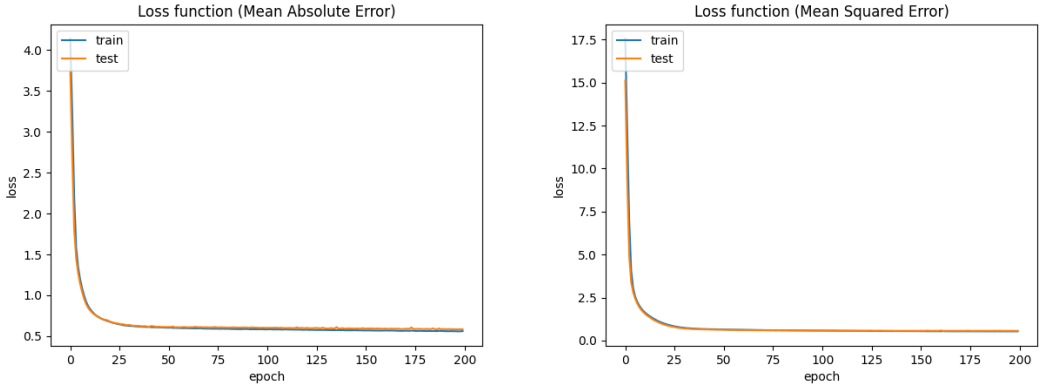


FIGURE 10 – 8-8-8-8-1

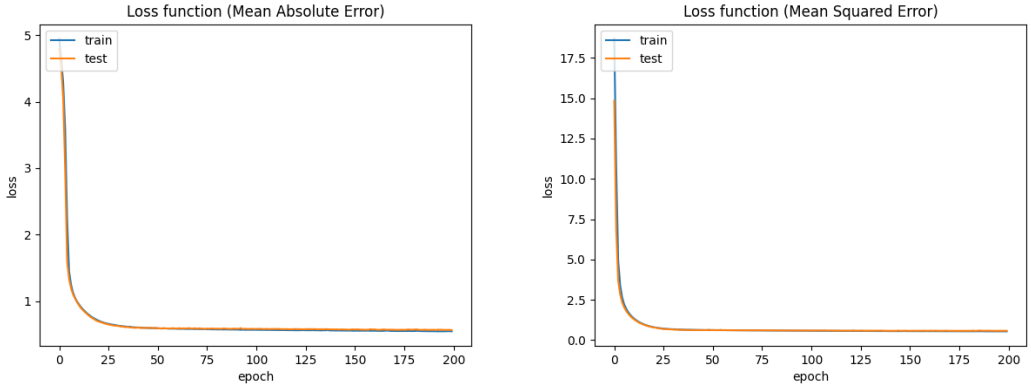


FIGURE 11 – 8-12-12-12-1

Notre modèle n'a pas réussi à généraliser.

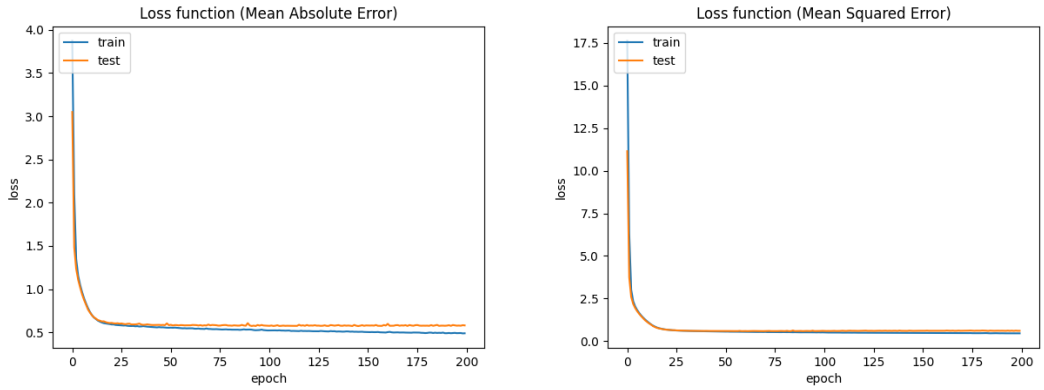


FIGURE 12 – 8-20-20-20-1

0.2.4 Classification prédictive

Après ce cuisant echec, nous avons opté pour un changement d'architecture. Nous avons donc recommencé cette fois ci avec un modèle de classification prédictive. Celà consiste à attribuer une étiquette de classe aux exemples d'entrée. A partir des connaissances a priori (données et modélisation), le modèle donne une probabilité pour l'événement. De tels modèles sont appelés modèles de classification. Les modèles prédictifs introduisent une notion de risque ; ils ne font pas de prédiction totalement certaine.

Nous avons tout d'abord défini le premier modèle. Il est composé de 4 couches composées respectivement de 11 neurones en entrée, de 4 neurones chacunes pour les 2em et 3em couches et 1 neurone pour la couche de sortie.

Avec cette configuration, nous avons défini un modèle de régression et obtenu les résultats suivants :

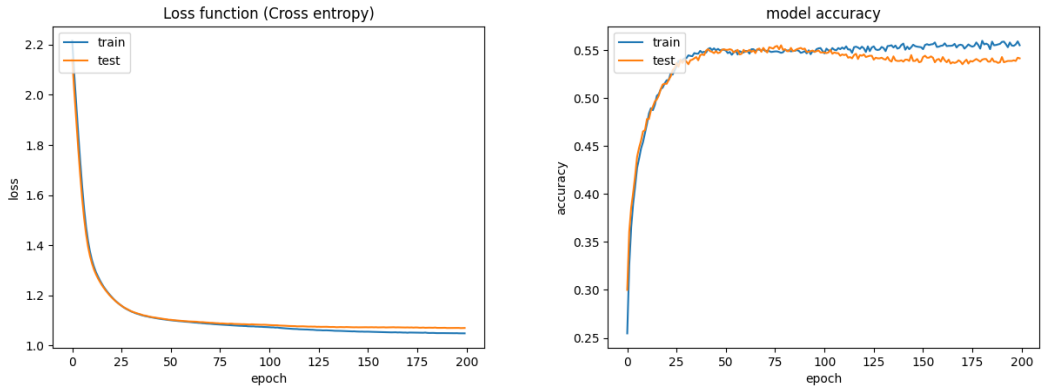


FIGURE 13 – 11-4-4-10

Des le debut, on observe une amélioration par rapport aux modèles de régression. Les courbes de teste et d'entrainement sont maintenant proches mais pas superposées.

Comme pour la regression, complexifions le modèle.

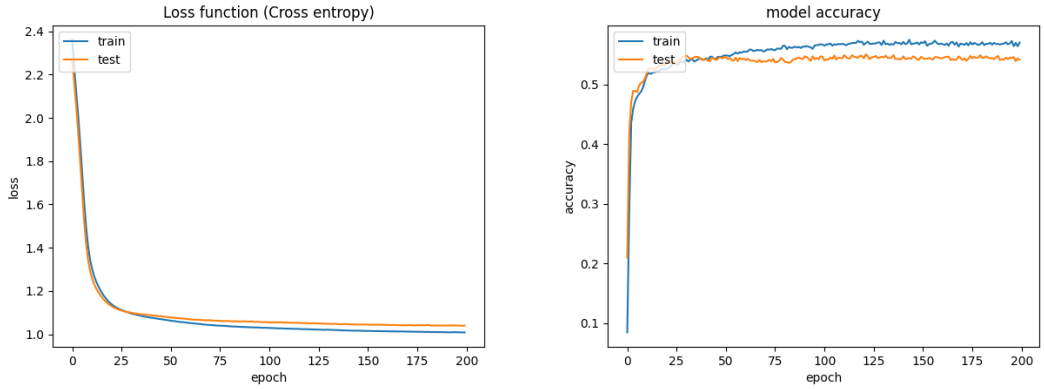


FIGURE 14 – 11-8-8-10

Les courbes ne semblent pas avoir beaucoup été impactées par le changement de la configuration. Continuons de complexifier le modèle.

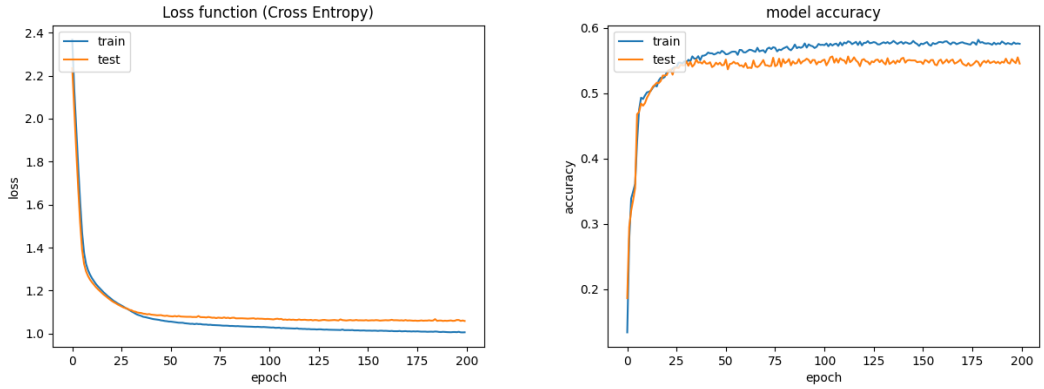


FIGURE 15 – 11-8-8-8-10

Contrairement à la régression, le modèle semble déjà converger vers une meilleure généralisation dès 8 neurones par couches. En effet, on voit bien que les courbles d'entrainement et de teste sont séparées tout en étant suffisement proches pour ne pas créer de sur-apprentissage.

Dans notre quête de perfection, nous avons donc décidé de continuer de complexifier le modèle en essayant par exemple de rajouter une couche. Notre nouveau modèle avec 3 couches de est donc composé de 5 couches composées respectivement de 11 neurones en entrée, de 8 neurones chacunes pour les 2em, 3em et 4em couches

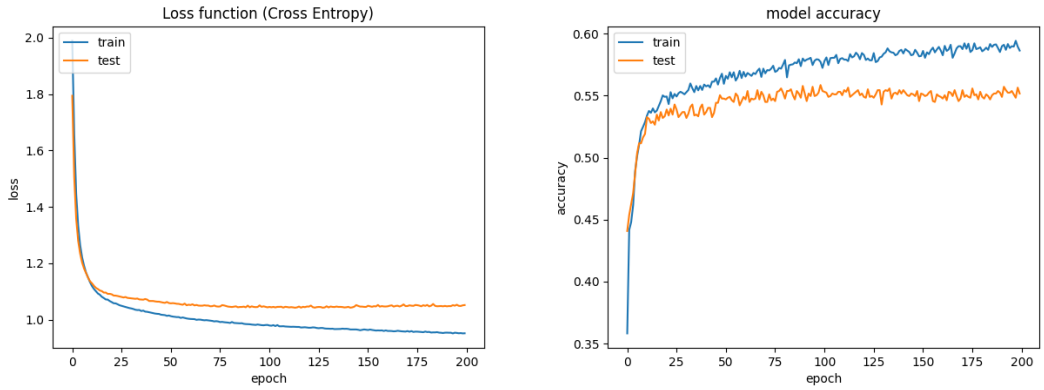


FIGURE 16 – 11-8-8-8-10

Malgré tout notre modèle peine à généraliser. Il ne semble pas exéder les 55% de précision et commence à tendre vers un sur-apprentissage. Tout comme pour le modèle de régression, nous avons décidé de supprimer les caractéristiques inutiles. Celà fait, nous avons retenté les mêmes expériences que précédemment.

On observe que le modèle avec 3 couches de 8 neurones semble avoir une généralisation avec un bien plus grande marge de progression que les autres, nous avons donc tenter de lui donner un plus grand nombre d'époch pour l'entrainement. Avec 2 fois plus d'époch, voici les résultats obtenus :

Finalement, il s'avère que le modèle ne généralise pas suffisement même avec plus de temps d'entrainement.

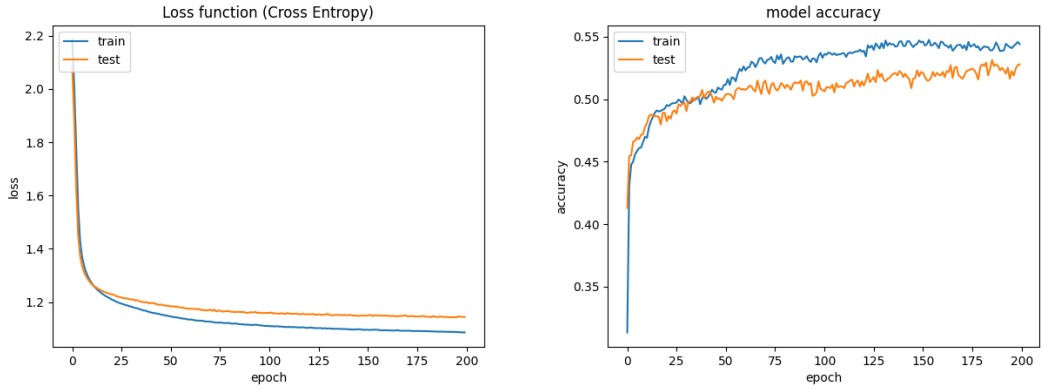


FIGURE 17 – 8-8-8-8-10

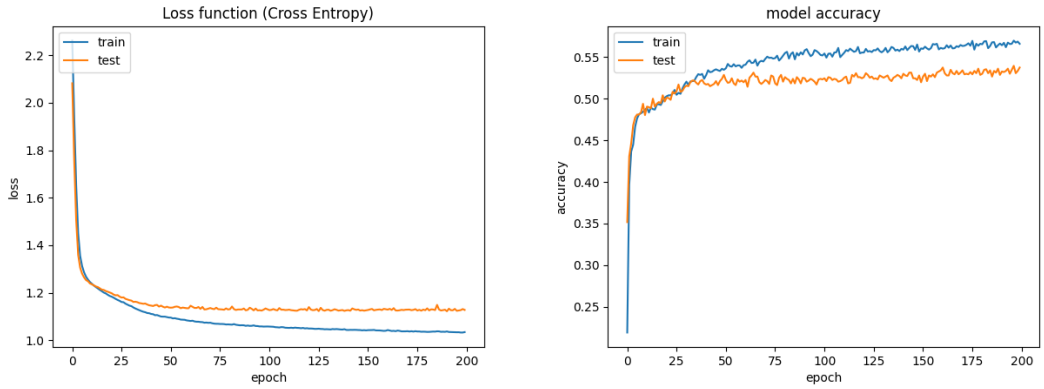


FIGURE 18 – 8-12-12-12-10

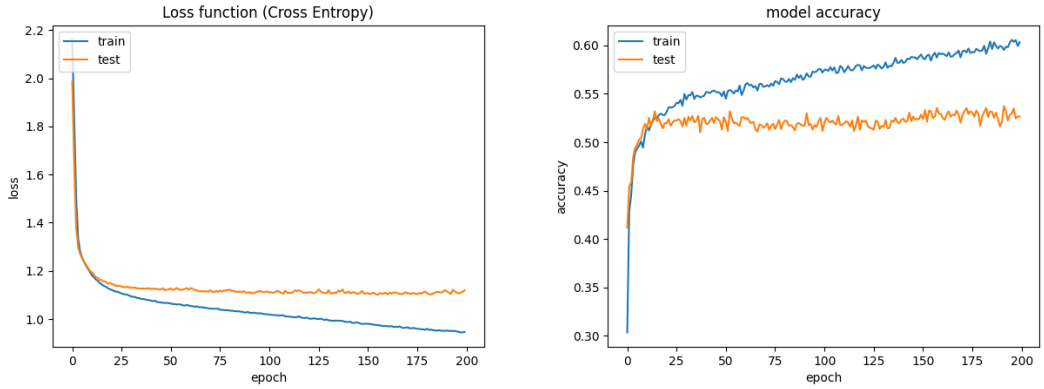


FIGURE 19 – 8-20-20-20-10

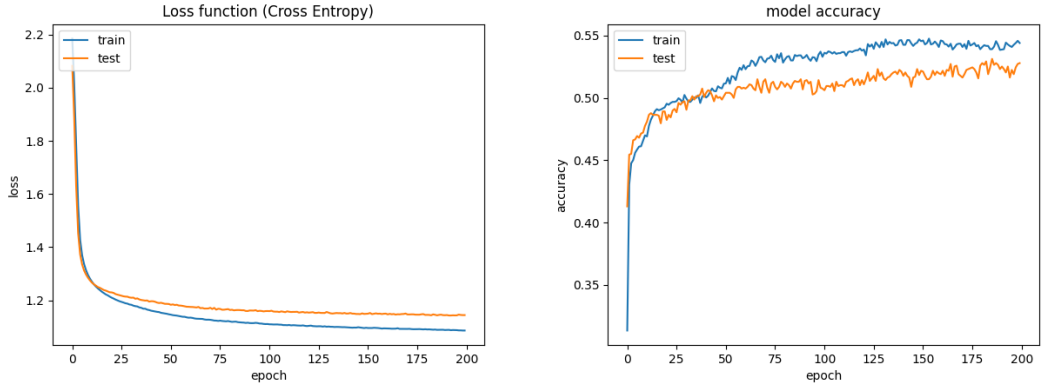


FIGURE 20 – 8-8-8-8-10