October 17, 2023

1 Réseau de neurones multicouches

Les réseaux de neurones sont des modèles inspirés du fonctionnement du cerveau humain. Ils sont utilisés en apprentissage automatique pour résoudre une grande variété de tâches, notamment la classification, la régression, la génération de texte, la vision par ordinateur, etc.

Son fonctionnement se divise en plusieurs étapes :

1. Initialisation des poids : Au début, les poids des connexions entre les neurones sont généralement initialisés de manière aléatoire. Ces poids sont les paramètres que le réseau apprendra à ajuster.

2. Propagation avant (Forward Propagation) :Pendant la phase d'entraînement, les données d'entrée x sont propagées à travers le réseau depuis la couche d'entrée jusqu'à la couche de sortie, en effectuant une moyenne pondérée par les poids W avec l'ajout des biais B, exprimée par la formule :

   $$y = \sigma(Wx + B)$$

   Chaque neurone calcule une valeur pondérée en combinant les valeurs de sortie de la couche précédente à l'aide des poids associés, puis applique une fonction d'activation σ. Ce processus se répète couche par couche jusqu'à atteindre la couche de sortie.

3. Calcul de la fonction de coût : La fonction de coût est aussi dans notre cas la cross-entropy.

4. Rétropropagation de Gradient et Mise à Jour des Poids : Pour minimiser la fonction de coût et ajuster les poids du réseau, le processus de rétropropagation de gradient est utilisé. L'erreur est propagée en sens inverse, de la couche de sortie à la couche d'entrée, en utilisant la dérivée de la fonction de coût par rapport aux poids.

   Dans le modèle qu'on a contruit, les poids sont ajustés en utilisant l'algorithme RMSprop, qui est un algorithme d'optimisation qui accélère la descente de gradient.Cette mise à jour est effectuée avec un taux d'apprentissage

(α) pour obtenir les nouveaux poids W_{t+1} à partir des poids actuels W_t, comme indiqué dans la formule suivante :

$$W_{t+1} = W_t - \alpha \nabla (fonctioncoût)$$

Cette étape de rétropropagation et de mise à jour des poids est essentielle pour permettre au réseau de converger vers une meilleure performance au fil de l'entraînement.

Le processus de propagation avant, de calcul de la fonction de coût, de rétropropagation et de mise à jour des poids est répété pour de nombreux exemples d'entraînement, regroupés en batchs. Chaque itération sur un batch est appelée une "epoch". Le modèle continue d'apprendre jusqu'à ce que la fonction de coût converge vers un minimum ou jusqu'à un nombre d'époch prédéfini.

1.1 Entity Embedding

Les réseaux neuronaux prédominent la plupart des algorithmes, mais lorsqu'il s'agit de variables catégorielles, c'est pas toujours le cas. En principe, un réseau neuronal peut approximer n'importe quelle variable continue. Cependant, il n'est pas adapté pour approximer des caractéristiques non continues, car il suppose un certain niveau de continuité dans sa forme générale. Pendant la phase d'entraînement, la continuité des données garantit la convergence de l'optimisation, et pendant la phase de prédiction, elle garantit que de légères modifications des valeurs d'entrée maintiennent la stabilité de la sortie.Donc, à chaque découverte d'une meilleure façon de révéler la continuité des données, la capacité des réseaux neuronaux à apprendre les données est meilleure.

Une façon courante de contourner ce problème est d'utiliser le codage one-hot-encoding, mais cela peut entraîner un problème de matrice creuse lorsque les variabels ont une grande cardinalité. De plus, ce codage traite les différentes valeurs des variables catégorielles comme étant complètement indépendantes les unes des autres, ce qui conduit souvent à ignorer les relations informatives entre elles.

L'Entity Embedding est une technique avancée d'encodage de variables catégorielles qui vise à surmonter les limitations du "one-hot encoding" tout en permettant aux réseaux de neurones d'apprendre des représentations significatives des catégories. Cette méthode repose sur l'idée que les variables catégorielles peuvent être intégrées dans un espace vectoriel de dimension réduite, où chaque catégorie est représentée par un vecteur dense continu plutôt que par un vecteur binaire creux.

On a commencé à attribuer un entier unique à chaque catégorie de la variable en utilisant le LabelEncoder. Une couche d'incorporation (embedding layer) est ajoutée au début du réseau de neurones et utilisée pour effectuer une transformation linéaire. Cette couche prend en entrée les entiers qui seront multipliés par une matrice de poids pour obtenir les vecteurs d'embedding correspondants.

La dimension de ces vecteurs est un hyperparamètre du modèle.

Une fois que les vecteurs d'embedding ont été calculés pour toutes les variables catégoriques, ils sont concaténés avec les valeurs des variables continues pour former la représentation complète de l'entrée du modèle. Cette représentation est ensuite utilisée comme entrée pour les couches qui suivent et surlaquelle le modèle sera entrainé pour faire la classification.

Au début de l'entraînement, les vecteurs d'embedding sont généralement initialisés de manière aléatoire. Chaque catégorie de la variable catégorielle se voit attribuer un vecteur unique de dimension spécifiée. Le modèle apprend les poids de la transformation linéaire entre les entiers encodés et les vecteurs d'embedding, de la même manière que les autres paramètres du réseau, en utilisant des méthodes d'optimisation comme la rétropropagation du gradient, ce qui lui permet d'ajuster les vecteurs d'embeddings de manière adaptative.

L'entity embedding est donc une approche puissante pour gérer les variables catégorielles dans les réseaux de neurones, offrant une représentation continue, efficace et riche en informations, et permettant de réduire la dimensionnalité des données catégorielles tout en contribuant à améliorer les performances du modèle.