TP : Réseaux de neurones

Dans ce TP, nous allons étudier les réseaux de neurones en classification puis en régression.

Classification

Contexte

Dans cette partie nous allons travailler sur le jeu de données "Human Resources" de Kaggle. Dans ce jeu de données, l'une des colonnes couramment utilisées pour la prédiction est souvent la colonne "left". Cette colonne représente généralement si un employé a quitté l'entreprise ou non. La prédiction de cette colonne peut être formulée comme une tâche de classification binaire, où l'objectif est de prédire si un employé va quitter l'entreprise (1) ou rester (0) en fonction des autres caractéristiques fournies dans le jeu de données.

Étapes de classification

1. **Obtenez le jeu de données :** téléchargez le jeu de données "Human Resources" depuis Nextcloud.
2. **Importez les bibliothèques :**

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
```

3. **Charger et explorez les données :** afficher les dimensions de la base de données, visualisez les premières lignes, identifier le type de chaque colonne.
4. **Préparer les données :**
   — Encodez les variables catégorielles : transformer chaque catégorie en un entier correspondant. La bibliothèque sklearn fournit des outils nous aidant à cette tâche. Étudiez le code suivant et utilisez le :

```
        def label_encoder(data, column_name):
        # Transforme un type catégorie en entier
        le = LabelEncoder()
        # On récupère tous les noms de catégories possibles
        unique_values = list(data[column_name].unique())
        le_fitted = le.fit(unique_values)
        # On liste l'ensemble des valeurs
        values = list(data[column_name].values)
        # On transforme les catégories en entier
        values_transformed = le.transform(values)
        # On fait le remplacement de la colonne dans
        # le dataframe d'origine
        data[column_name] = values_transformed
```

- Divisez les données en ensembles d'entraînement et de test (80% pour l'entrainement et 20% pour le test).
- Normalisez les données.
- Entraînez un réseau de neurones de type `MLPClassifier`. Utilisez, par exemple, deux couches cachées de 10 neurones chacune, la fonction 'Relu' comme fonction d'activation et la descente de gradient stochastique comme méthode d'optimisation de la perte. Le nombre d'itération pourrait être fixé à 1000.
- Évaluez votre modèle en affichant sa précision, la matrice de confusion.
- Utilisez la fonction `classification_report` qui prend en entrée les étiquettes réelles et les prédictions du modèle, puis génère le rapport de classification pour chaque classe ainsi que la moyenne globale des métriques.

5. **Optimisez les hyperparamètres** : Utilisez `GridSearchCV` pour rechercher les meilleurs hyperparamètres. Voici un exemple d'utilisation

```
from sklearn.model_selection import GridSearchCV

param_grid = {
        'hidden_layer_sizes': [(50,),(100,),(150,)],
        'activation': ['relu', 'tanh'],
        'solver': ['adam', 'sgd'],
}

grid_search = GridSearchCV(MLPClassifier(max_iter=1000,
                        random_state=42), param_grid, cv=5)
grid_search.fit(X_train, y_train)
```

```
print("Best parameters found: ", grid_search.best_params_)
```

Analyse avancée des données

L'entreprise qui vous a fourni les données, souhaite retenir les bons employés. Elle vous demande donc d'analyser les données afin de comprendre pourquoi les bons employés partent de l'entreprise. Un bon employé est défini par un employé :

— qui travaille sur plus de projets que la moyenne,
— effectue plus d'heures par mois que la moyenne,
— a obtenu une meilleure évaluation que la moyenne.
— travaille dans la société depuis plus longtemps que la moyenne.

Vous pouvez suivre les étapes suivantes :

1. **Création de la colonne 'bon_employe'** : Avant de commencer l'analyse, vous devez créer une nouvelle colonne 'bon_employe' en fonction des critères définis.
2. **Analyse des raisons du départ des bons employés** : vous pouvez analyser les raisons du départ des bons employés en comparant leurs caractéristiques avec celles des employés qui restent.
3. **Analyse de la corrélation** : utiliser une matrice de corrélation pour examiner les relations entre les différentes variables.

Régression

Les réseaux de neurones peuvent aussi être utilisés pour des problèmes de régression. Dans cette seconde partie, nous allons étudier la base de données bike sharing by hour. Il s'agit de prédire le nombre de locations de vélos en fonction de différents critères (météo, week-end etc). La même base est disponible mais avec une répartition journalière. Comme pour la première partie, récupérez la base, analysez les données, et effectuez un apprentissage avec les réseaux de neurones.