Brief : choix du meilleur Classifieur pour La reconnaissance des Digits audios

Mars 2022

Sommaire :

Contexte du projet :

Cet atelier a pour objectif de manipuler les différentes techniques de classification supervisée sous Python, comme KNN, SVM, Foret Aléatoire, XGBoost et autre.

Dans cet atelier, nous appliquons apprentissage supervisé pour reconnaitre les Digits à partir d'un enregistrement audio.

Challenges :

- Collection d'une base de données.
- Analyse, prétraitement et visualisation des données.
- Préparation des données pour l'apprentissage.
- Préparation de Pipeline en utilisant « sklearn.pipeline.Pipeline ».
- Choix de meilleur modèle.
- Vérification de l'efficacité de ce modèle à des nouvelles données.
- Intégration de ce modèle dans une application pour test temps réel.

Phases du projet (3 parties)

Partie 1 : Base de données, Analyse, Prétraitement et Préparation

Pour aborder cette problématique de la reconnaissance des Digits, il est primordial d'avoir ou de collecter une DataSet. Pour cela, la fonction « **collection** » dans **Tools/tools.py** vous permettra de collecter votre propre DataSet.

NB. La description de votre DataSet et le processus de collection doivent être notés dans le compte rendu final).

Par la suite, il faut Appliquer les traitements nécessaires pour préparer la DataSet en utilisant Numpy et Pandas. **Les étapes de ce processus de traitement doivent être bien enchainer, claire, commenter et fonctionnelle dans le Notebook.**

Partie 2 : Pipeline Apprentissage/Classification Supervisée

Cette deuxième partie est réservée pour lancer une série d'expérimentation en préparant un Pipeline de plusieurs techniques de classification supervisé, notamment : **KNN, SVM, Arbre de décision, Forêt Aléatoire et XGBoost.**

Pour une étude expérimentale bien complète, il est recommandé de varier les paramètres de chaque technique utilisée, pour cela, il est préférable d'utiliser les outils proposés par Sklearn comme : **Pipeline** et **GridSearchCV**.

Partie 3 : Mettre en place la solution dans une application Test Temps Réel

Utilisez la fonction « **joblib** » pour enregistrer votre modèle, cela une fois vous avez préparé votre meilleur modèle de classification.

Faites intégrer cette solution dans une Application en utilisant la fonction « **rec** » qui est dans **Tools/tools.py**.

Liste des documents présents sur le repository github

https://github.com/adthw/classifier_digits_audio.git

- DataSet.
- La réalisation des différentes étapes décrite dans le contexte.
- Les Notebooks demandés.
- Le compte rendu demandé.

Analyse et construction du modèle :

J'ai pu tester différents modèles grâce à la construction d'un pipeline avec différents classifiers, applications de multiples paramètres en gridsearch et affichage des performances pour trouver le meilleur classifier avec les meilleurs paramètres.

Les meilleurs résultats semblent être obtenus avec Choix du modèle SVC (kernel=rbf C=10) et avec application de la standardisation de données qui nous donne un score de **93,75%** de reconnaissance, ce qui semble satisfaisant.

Résultats :

```
Score (svc) : 93.75%
Meilleurs paramètres : {'svc__C': 10, 'svc__kernel': 'rbf'}

 ======================================================================

Score (svc_0) : 81.25%
Meilleurs paramètres : {'svc__C': 1, 'svc__kernel': 'linear'}

 ======================================================================

Score (knn) : 62.5%
Meilleurs paramètres : {'knn__metric': 'manhattan', 'knn__n_neighbors': 4}

 ======================================================================

Score (knn_0) : 75.0%
Meilleurs paramètres : {'knn__metric': 'euclidean', 'knn__n_neighbors': 5}

 ======================================================================

Score (decision_tree) : 62.5%
Meilleurs paramètres : {}

 ======================================================================

Score (decision_tree_0) : 62.5%
Meilleurs paramètres : {}

 ======================================================================

Score (random_forest) : 87.5%
Meilleurs paramètres : {'forest__n_estimators': 123}

 ======================================================================

Score (random_forest_0) : 81.25%
Meilleurs paramètres : {'forest__n_estimators': 101}

 ======================================================================

Score (boost) : 56.25%
Meilleurs paramètres : {'boost__learning_rate': 0.1, 'boost__loss': 'deviance'}

 ======================================================================

Score (boost_0) : 75.0%
Meilleurs paramètres : {'boost__learning_rate': 0.1, 'boost__loss': 'deviance'}

 ======================================================================

Score (XGB) : 62.5%
Meilleurs paramètres : {'XGB__eta': 0.3, 'XGB__eval_metric': 'mlogloss', 'XGB__max_depth': 4}

 ======================================================================

Score (XGB_0) : 62.5%
Meilleurs paramètres : {'XGB__eta': 0.3, 'XGB__eval_metric': 'mlogloss', 'XGB__max_depth': 4}

 ======================================================================

Score (MLP) : 87.5%
Meilleurs paramètres : {'MLP__activation': 'logistic', 'MLP__hidden_layer_sizes': (150,)}

 ======================================================================

Score (MLP_0) : 81.25%
Meilleurs paramètres : {'MLP__activation': 'tanh', 'MLP__hidden_layer_sizes': (150,)}

 ======================================================================
```

Utilisation de Joblib :

L'utilisation de joblib pour la création du programme final permet de sauvegarder des structures de données dans un fichier pour pouvoir les recharger après.

Conclusion :

L'application reconnait la prononciation de ZERO et NEUF à 100% et classe tous les autres digits en NEUF.

En ajoutant un enregistrement sans prononciation les scores se sont fortement dégradés, ce qui tendrait à dire que la qualité des données d'entrainement sont essentielles à la production d'un modèle de qualité

Propositions d'améliorations :

Ajouter une fonctionnalité pendant la phase de test pour ajouter ces test au dataset et dire au modèle comment bien classer ces nouvelles données.