Transpile_c
TP 2023

6 AVRIL

Antoine Binet
EfreiParis

Plutôt que coder à chaque fois un modèle en C on veut créer un outil qui va récupérer un modèle entraîné via scikit-learn et générer le Code C d'inférence. L'objectif est de générer les code d'inférence en C pour

- une régression linéaire
- une régression logistique
- un arbre de décision

1. Coder un script python **transpile_simple_model.py** qui :

- - charge un fichier joblib contenant une régression linéaire entraînée
- - récupère les valeurs de coefficients
- - génère une chaîne de caractère contenant le code C permettant de calculer la prédiction du modèle (float prediction(float *features, int n_feature))avec les valeur du coefficient
- - génère une fonction main qui permet d'appeler prediction sur une donnée définié par un tableau statique de votre choix.
- - sauvegarde le code c généré dans un fichier.c
- - et affiche la commande de compilation à lancer pour le compiler ou le compile directement.

```
import joblib

# charge un fichier joblib contenant une régression linéaire entraînée
model = joblib.load('modele.joblib')

# Rrécupère les valeurs de coefficients
coefficients = model.coef_
intercept = model.intercept_

# génère une chaîne de caractère contenant le code C permettant de calculer la
prédiction du modèle
# (float prediction(float *features, int n_feature) )avec les valeur du coefficient
code_c = 'float prediction(float *features, int n_feature) {\n'
code_c += '\tfloat result = %f;\n' % intercept
for i in range(len(coefficients)):
    code_c += '\tresult += %ffeatures[%d];\n' % (coefficients[i], i)
code_c += '\treturn result;\n}\n'

# sauvegarde le code c généré dans un fichier.c
with open('modele.c', 'w') as f:
    f.write(code_c)

main_code = '#include <stdio.h>\n\n'
main_code += 'float prediction(float *features, int n_feature);\n\n'
main_code += 'int main() {\n'
main_code += '\tfloat features[] = {1.0, 2.0, 3.0};\n'
```

```
main_code += '\tfloat result = prediction(features, %d);\n' % len(coefficients)
main_code += '\tprintf("Prediction : %f\\n", result);\n'
main_code += '\treturn 0;\n}\n'

with open('main.c', 'w') as f:
    f.write(main_code)

# ffiche la commande de compilation à lancer pour le compiler ou le compile
directement.
print('Pour compiler et exécuter le code généré : gcc modele.c main.c -o modele &
start modele.exe')
```

2. Entraîner une régression linéaire simple sur un dataset simple (par exemple houses.csv) et le sauvegarder

```
import pandas as pd
from sklearn.linear_model import LinearRegression
import joblib

data = pd.read_csv('houses.csv')

X = data['size'].values.reshape(-1, 1)
y = data['nb_rooms'].values

model = LinearRegression()
model.fit(X, y)

joblib.dump(model, 'modele.joblib')
```

3. Lancer le script transpile_simple_model et compiler le fichier C générer. Vérifier que les prédictions produits sont confirme au model.predict

Malheureusement dûe à un problème avec gcc je n'ai pas pu lancer la commande et compiler le fichier c générer.