Przewidywanie stabilności systemu symulującego zmiany klimatyczne (Climate Model Simulation Crashes)

KWD

Piotr Dudek, Mariusz Górny

29.01.2020

Streszczenie

Projekt modelu klasyfikatora binarnego dla problemu przewidywania stabilności systemu symulującego zmiany klimatyczne. Baza danych Climate Model Simulation Crusches zawiera informacje na temat parametrów modelu klimatycznego i wyniku o powodzeniu lub niepowodzeniu symulacji. Została przeprowadzona analiza eksploracyjna tych danych, a następnie na ich podstawie, dzięki wykorzystaniu narzędzi do uczenia maszynowego oferowanych przez biblioteki języka Python 3, powstał model klasyfikatora binarnego, który poprawnie przewiduje stabilność systemu. Model wykorzystuje implementację regresji logistycznej, która pozwala na określenie prawdopodobieństwa powodzenia symulacji na podstawie danych wejściowych. Powstałe narzędzie zostało sprawdzone pod kątem jakości dla dostępnych danych.

Spis treści

Rozdział 1

Wprowadzenie

Celem projektu jest opracowanie i wytrenowanie modelu klasyfikatora dla problemu przewidywania stabilności systemu symulującego zmiany klimatyczne. Projekt został wykonany za pomocą języka Python 3 z użyciem bibliotek Scikit-learn, Matplotlib i Pandas, służacych do zdefiniowania i uczenia modelu klasyfikatora, operowania na bazie danych Climate Model Simulation Crashes oraz tworzenia wykresów.

1.1 Opis problemu

Baza danych Climate Model Simulation Crashes zawiera dane parametrów i wyniku o powodzeniu lub niepowodzeniu symulacji, podczas kwantyfikacji niepewności (UQ) modelu klimatycznego. Zestawy wartości 18 parametrów zostały uzyskane metodą łacińskich hiperkostek (ang. Latin Hypercube Sampling (LHS)). Zestawy parametrów zostały wygenerowane w 3 oddzielnych losowaniach, po 180 zestawów parametrów każde. 46 z 540 przeprowadzonych symulacji zakończyło się niepowodzeniem z powodów numerycznych dla konkretnych wartości parametrów. Uzyskany klasyfikator zostanie użyty do przewidywania wyniku symulacji (powodzenia lub niepowodzenia) oraz określenia powodów niepowodzenia symulacji.

1.2 Opis danych wejściowych

Zestaw danych składa się z 540 wierszy, reprezentujących kolejne symulacje. Pierwsza kolumna zawiera numer losowania, za pomocą którego zostały wygenerowane parametry symulacji. Druga kolumna zawiera numer symulacji (od 1 do 180) w obrębie losowania. Kolumny 3-20 zawierają wartości parametrów symulacji, przedstawione w postaci liczby rzeczywistej z przedziału [0, 1]. Ostatnia kolumna zawiera wynik symulacji (0 - niepowodzenie, 1 - powodzenie).

1.3 Analiza danych wejściowych

	Study	Run	vconst_corr	...	bckgrnd_vdc_psim	Prandtl	outcome
0	1	1	0.859036	...	0.796997	0.869893	0
1	1	2	0.606041	...	0.438447	0.512256	1
2	1	3	0.997600	...	0.285636	0.365858	1
3	1	4	0.783408	...	0.699431	0.475987	1
4	1	5	0.406250	...	0.280098	0.132283	1

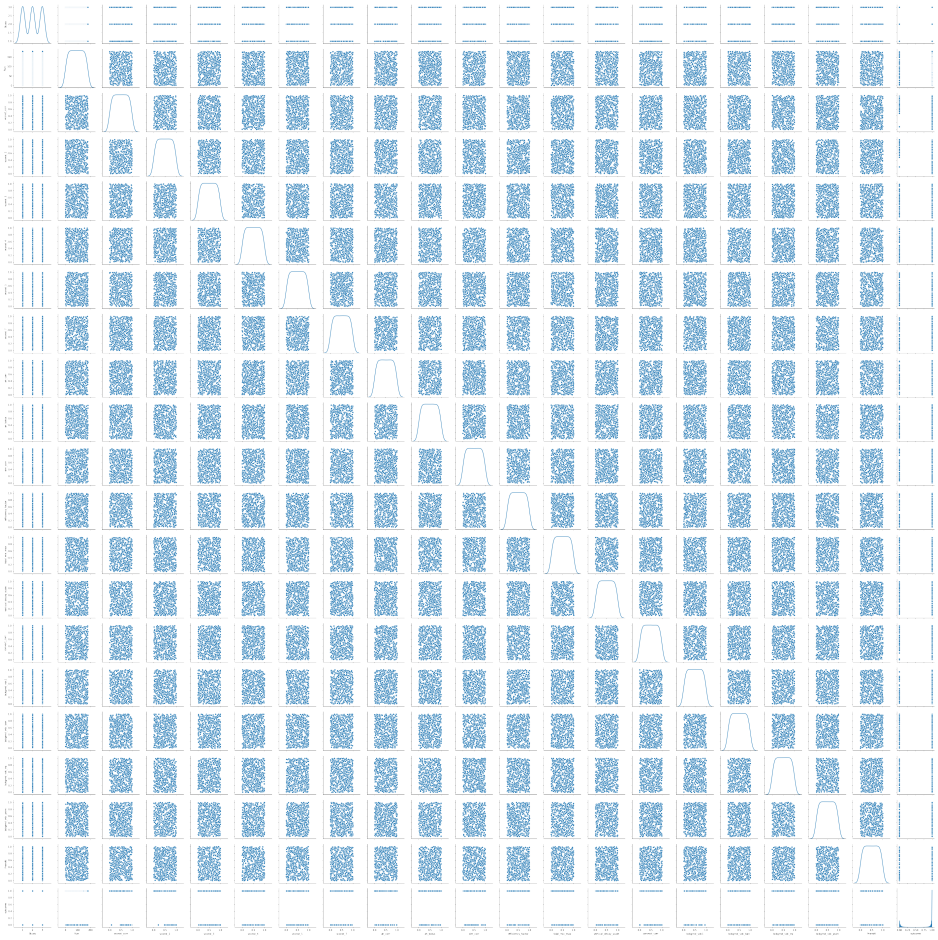
5 rows × 21 columns

Rysunek 1.1: Pierwsze 5 wierszy z bazy danych. Kolumny 4-18 zostały ukryte dla zwiększenia czytelności obrazu.

Pierwsze dwie kolumny to dane identyfikujące kolejne symulacje, z tego powodu zostały wyłączone z uczenia i testowania modelu. Ostatnia kolumna zawiera wynik symulacji i została potraktowana jako rezultat dla modelu. Kolumny 3-20 zawierają wartości kolejnych parametrów, które są danymi wejściowymi dla modelu. Wartości te to liczby rzeczywiste z przedziału [0, 1], dlatego nie ma konieczności przeprowadzania dalszej normalizacji danych.

Po rozdzieleniu danych na zbiory danych wejściowych i rezultatu, rozmiar tych zbiorów odpowiada opisowi i założeniom:

```
Wymiar danych wejściowych:  (540, 18)
Wymiar rezultatu:  (540,)
```



Rysunek 1.2: Wynik analizy eksploracyjnej danych.

Analiza eksploracyjna danych nie wykazała korelacji pomiędzy poszczególnymi wymiarami danych wejściowych. Rozkład wartości parametrów jest inny niż normalny. Jest to spowodowane sposobem uzyskania wartości parametrów - zostały one wygenerowane w taki sposób, aby zestawy parametrów pokrywały całe spektrum możliwych ich wartości. Nie jest możliwa redukcja wymiarów danych wejściowych przed przystąpieniem do tworzenia i trenowania modelu.

Rozdział 2

Opis metody

2.1 Wprowadzenie teoretyczne

Problem jest przykładem klasyfikacji binarnej, w której dla n-wymiarowych danych wejściowych, wynikiem jest prawdopodobieństwo przypisania do jednej z 2 grup (w badanym przypadku - powodzenie (1) lub niepowodzenie (0) symulacji).

Wytrenowany model został oparty o regresję logistyczną, liniowy model klasyfikacji binarnej (może być także użyty do klasyfikacji multiklasowej), w którym prawdopodobieństwo przydzielenia do klasy jest uzyskiwane przy pomocy funkcji logistycznej.

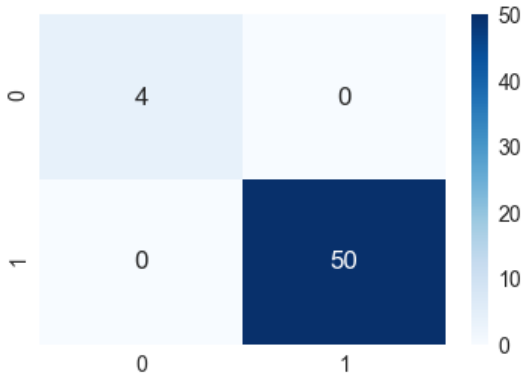
Regresję logistyczną implementuje klasa `LogisticRegression` z biblioteki Sklearn.

2.2 Badania symulacyjne

Dane wejściowe i rezultat zostały rozdzielone na zbiory testowy i uczący w stosunku 1 do 9. Następnie przy pomocy GridSearchCV, z zastosowaniem różnych wartości parametrów: `C`, `penalty`, `max_iter` oraz `tol`, został znaleziony najoptymalniejszy zestaw parametrów dla regresji logistycznej:

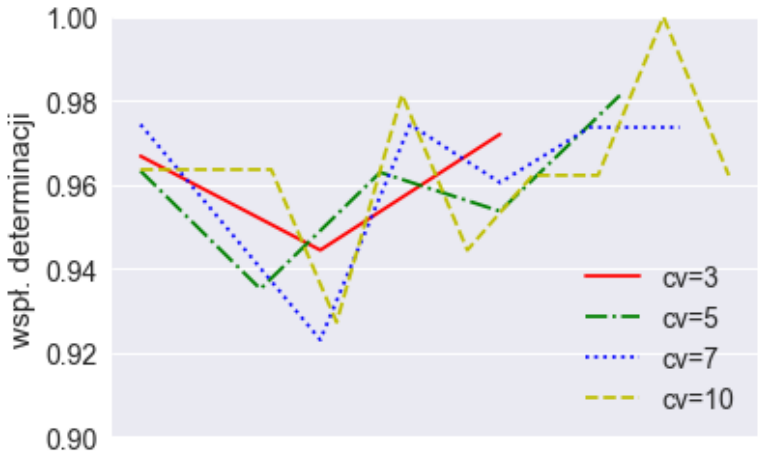
```
Najlepszy model został znaleziony dla parametrów
{'C': 100.0, 'max_iter': 100, 'penalty': 'l2', 'tol': 0.0001}
ze współczynnikiem determinacji: 0.9526748971193416.
```

2.3 Analiza jakości modelu



Rysunek 2.1: Matryca konfuzji dla wytrenowanego modelu.

Matryca konfuzji wykonana dla danych testowych pokazuje, że model bardzo dobrze przewiduje wyniki powodzenia symulacji. W celu lepszej oceny modelu została przeprowadzona walidacja krzyżowa (`cross_val_score`) z 3, 5, 7 i 10 podziałami (wartość parametru `cv`). Wynik walidacji przedstawia rysunek 2.2:



Rysunek 2.2: Wykres wartości współczynnika determinacji dla poszczególnych podziałów walidacji krzyżowej.

Nawet dla bardzo różnych podziałów bazy danych, model utrzymuje wysoki współczynnik determinacji - powyżej 0,92. Podczas uczenia modelu nie zaszło także zjawisko overfittingu, dzięki czemu model powinien poprawnie przewidywać wynik symulacji modelu klimatycznego dla nowych zestawów wartości parametrów.

Rozdział 3

Podsumowanie

Cel projektu został osiągnięty. Analiza zbioru danych Climate Model Simulation Crashes nie wykazała korelacji pomiędzy poszczególnymi parametrami modelu klimatycznego, przez co nie było możliwe wyodrębnienie parametrów nieistotnych.

Do rozwiązania problemu został zastosowany kwalifikator binarny - regresja logistyczna. Po dobraniu odpowiednich parametrów dzięki zastosowaniu procedury GridSearch, pozwoliła ona na stworzenie modelu poprawnie przewidującego stabilność systemu symulującego zmiany klimatyczne. Powstałe narzędzie zostało sprawdzone pod kątem jakości rozwiązania - współczynnik determinacji po walidacji krzyżowej zawiera się w przedziale 0,92 - 1,0. Model spełnia założenia projektu i może być przydatny do dalszych badań nad systemem symulującym zmiany klimatyczne.

Dodatek A

Kod programu

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
import seaborn as sb

#wczytanie danych z pliku CSV
pd_data = pd.read_csv('pop_failures.csv')

#wyświetlenie pierwszych wierszy danych
pd_data.head()

#wyodrębnienie danych wejściowych i rezultatu
target = np.array(pd_data["outcome"])
data = np.array(pd_data.iloc[:,2:-1].values)

#wyświetlenie rozmiaru danych
print("Wymiar danych wejściowych: ", data.shape)
print("Wymiar rezultatu: ", target.shape)

#wizualizacja danych
sb.pairplot(pd_data, diag_kind="kde")

#rozdzielenie danych na zbiory uczący i testowy
train_data, test_data, train_target, test_target = \
train_test_split(data,target, test_size=0.1, random_state=1)
```

```
#wyszukanie najoptymalniejszego modelu regresji logistycznej
logistic_regression_gs = GridSearchCV(
    LogisticRegression(),
    {
        "C": np.logspace(-3,3,7),
        "penalty": ["l1","l2"],
        "max_iter": [100, 500, 1000],
        "tol": [1e-4, 1e-5, 1e-6]
    },
    cv=10)
logistic_regression_gs.fit(train_data,train_target)

print("Najlepszy model został znaleziony dla parametrów", \
    logistic_regression_gs.best_params_, \
    "ze współczynnikiem determinacji:", \
    logistic_regression_gs.best_score_)

#matryca konfuzji dla wytrenowanego modelu
sb.heatmap(confusion_matrix(test_target, \
    logistic_regression_gs.predict(test_data)), \
        annot=True, cmap="Blues")

#ocena jakości modelu z zastosowaniem walidacji krzyżowej
cvs = [3, 5, 7, 10]
results = []
for cv in cvs:
    results.append( \
        cross_val_score(logistic_regression_gs, \
            data, target, cv=cv))

#wyświetlenie wyniku walidacji krzyżowej
markers = ['r-', 'g-.', 'b:', 'y--']
for i in range(4):
    plt.plot([x / (cvs[i]+1) for x in range(cvs[i])], \
        results[i], markers[i], label='cv='+str(cvs[i]))
plt.legend()
plt.ylim(0.9, 1)
plt.gca().xaxis.set_visible(False)
plt.ylabel('współ. determinacji')
plt.show()
```