

Projekt z Inżynierii Uczenia Maszynowego

Semestr letni 2020

Autorzy

Przemysław Stawczyk

Maciej Szulik

Notebooki

Notebooki znajdują się w katalogu `notebooks/`.

1. `data_explore.ipynb` realizuje projekt w ramach etapu 1., opisuje wyznaczenie zadań biznesowych, modelowania, kryteriów sukcesu, zawiera analizę danych;
2. `models_compare.ipynb` realizuje projekt w ramach etapu 2.i);

Eksperyment A/B

Eksperyment A/B jest przeprowadzany:

- poprzez zbieranie danych wraz predykcjami i informacją o użytym modelu, jako że nie istnieje natychmiastowa informacja zwrotna,
- z podziałem ruchu na poziomie zapytań, jako że każda dostawa jest traktowana jako zjawisko niezależne

W związku z tym API będzie udostępniało możliwość:

- pobrania historii predykcji:
 - w formie historii wraz z informacjami o użytym modelu (`history`),
 - w formie rekordów z predykcjami porównywanych modeli (`summary`),
- ładowania nowych modeli,
- zarządzania eksperymentem A/B:
 - włączenie/wyłączenie,
 - ustawienie modeli A i B na podstawie wcześniej ustawionych modeli.

Implementacja

Omówienie modułów:

`data_loader.py`

Moduł odpowiada za obróbkę danych przeznaczonych dla modelu zgodnie z obserwacjami powstałymi na etapie analizy.

models_training.py

Moduł pozwala tworzyć modele, wyliczać dla nich statystyki, porównywać je i dobierać pod nie najlepsze hiperparametry (dostępne modele to DecisionTreeRegressor, XGBRegressor, KNeighborsRegressor i RandomForestRegressor).

model_files_manager.py

Moduł odpowiada za zapisywanie i ładowanie modeli do/z pliku `.pkl` znajdującego się w podkatalogu `models`.

models_creator.py

Moduł odpowiada za tworzenie i zapisanie modeli.

model_holder.py

Moduł odpowiada za zarządzanie dostępnymi modelami i wystawia metody pozwalające na dokonanie predykcji, agregację ich historii i statystyk.

web_api.py

Moduł wystawia API do uzyskiwania predykcji, ich historii, statystyk, a także pozwala na zarządzanie dostępnymi modelami i umożliwia przeprowadzenie eksperymentu A/B.

API

API zostało stworzone przy użyciu frameworku Flask i jest domyślnie wystawione na porcie 5000.

Uruchomienie

```
python3 web_api.py
```

Endpoints:

1. `/api/info`
 - GET
Zwraca informację o tytule, semestrze, autorach projektu i statusie aplikacji.
Przykład
Request:

```
curl --header "Content-Type: application/json" \
--request GET \
--data '{"A": "tree model", "B": "knn model"}' \
http://localhost:5000/api/info
```


Response:

```
{
  "authors": [
    "Przemysław Stawczyk",
    "Maciej Szulik"
  ],
  "semester": "20L",
  "status": "running",
  "title": "Projekt z Inżynierii Uczenia Maszynowego"
}
```

2. /api/prediction

- GET Zwraca predykcję dla danych wejściowych i statusie aplikacji.

Przykład

Request:

```
curl --header "Content-Type: application/json" \
--request GET \
--data '{
  "delivery_company": "360",
  "city": "Warszawa",
  "price": 10.23,
  "category": "Gry i konsole",
  "subcategory": "Gry na konsole"
}' \
```

http://localhost:5000/api/prediction

Response:

```
{"prediction": 48.098731928837054, "status": "running"}
```

3. /api/prediction/history

- GET Zwraca historię predykcji.

Przykład

Request:

```
curl -X GET localhost:5000/api/prediction/history
```

Response:

```
[
  {
    "delivery_company": 360,
    "city": "Warszawa",
    "price": 10.23,
    "category": "Gry i konsole",
    "subcategory": "Gry na konsole",
    "prediction": 50.1254427196,
    "model": "knn model"
  },
  {
    "delivery_company": 620,
    "city": "Warszawa",
    "price": 1011.11,
    "category": "Telefony i akcesoria",
  }
]
```

```

        "subcategory": "Telefony stacjonarne",
        "prediction": 47.3557808691,
        "model": "knn model"
    },
    {
        "delivery_company": 516,
        "city": "Police",
        "price": 351,
        "category": "Komputery",
        "subcategory": "Drukarki i skanery",
        "prediction": 51.645048671,
        "model": "knn model"
    }
]
4. /api/prediction/summary
    • GET Zwraca podsumowanie predykcji dla dostępnych modeli.
      Przykład
      Request:
      curl -X GET localhost:5000/api/prediction/summary
      Response:
      [
        {
          "delivery_company": 360,
          "city": "Warszawa",
          "price": 10.23,
          "category": "Gry i konsole",
          "subcategory": "Gry na konsole",
          "knn model": 50.1254427196,
          "tree model": 46.5884026029,
          "actual": null
        },
        {
          "delivery_company": 620,
          "city": "Warszawa",
          "price": 1011.11,
          "category": "Telefony i akcesoria",
          "subcategory": "Telefony stacjonarne",
          "knn model": 47.3557808691,
          "tree model": 35.4989317762,
          "actual": null
        },
        {
          "delivery_company": 516,
          "city": "Police",
          "price": 351,
          "category": "Komputery",

```

```

        "subcategory": "Drukarki i skanery",
        "knn model": 51.645048671,
        "tree model": 72.7802102179,
        "actual": null
    }
]

```

5. /api/prediction/models

- GET Zwraca listę dostępnych modeli.
Przykład
Request:
curl -X GET localhost:5000/api/prediction/models
Response: [{"name": "tree model", "filename": "tree_model.pkl"}, {"name": "knn model", "filename": "knn_model.pkl"}, {"name": "xgb model", "filename": "xgb_model.pkl"}]
- POST
Pozwala załadować do aplikacji model dostępny w katalogu models/.
Przykład
Request:
curl --header "Content-Type: application/json" \
--request POST \
--data '{"name": "tree 2", "filename": "tree_model.pkl"}' \
http://localhost:5000/api/prediction/models
curl -header "Content-Type: application/json"
-request POST
-data '{"active": true}'
http://localhost:5000/api/prediction/models
Response: {"status": "model added"}
Weryfikacja:
Request:
curl -X GET localhost:5000/api/prediction/models
Response: [{"name": "tree model", "filename": "tree_model.pkl"}, {"name": "knn model", "filename": "knn_model.pkl"}, {"name": "xgb model", "filename": "xgb_model.pkl"}, {"name": "tree 2", "filename": "tree_model.pkl"}]

6. /api/prediction/models/active

- GET
Zwraca informacje dotyczące aktywnego modelu.
Przykład
Request:
curl -X GET localhost:5000/api/prediction/models/active
Response:
{ "name": "tree model", "filename": "tree_model.pkl" }
- POST Pozwala zmienić aktywny model.
Przykład

Request:
curl --header "Content-Type: application/json" --request POST \
--data '{"name": "tree model"}' \
http://localhost:5000/api/prediction/models/active
Response:
{ "status": "model changed" }

7. /api/prediction/AB

- GET

Zwraca informację czy model jest aktywny.

Przykład

Request:
curl -X GET localhost:5000/api/prediction/AB
Response:
{ "active": false, "status": "ok" }

- POST Pozwala zmienić stan eksperymentu.

Przykład

Request:
curl --header "Content-Type: application/json" --request POST \
--data '{"active": true}' \
http://localhost:5000/api/prediction/AB
Response:
{ "active": true, "status": "ok" }

8. /api/prediction/AB/models

- POST

Pozwala ustawić, który z załadowanych modeli służy jako model A i B w eksperymencie

Przykład

Request:
curl --header "Content-Type: application/json" --request POST \
--data '{"A": "tree model", "B": "knn model"}' \
http://localhost:5000/api/prediction/AB/models
Response:
{ "active": false, "status": "ok" }

- GET

Pozwala sprawdzić, jakie modele biorą udział w eksperymencie A/B.

Przykład

Request:
curl -X GET localhost:5000/api/prediction/AB/models
Response:
{ "models": { "A": "tree model", "B": "knn model" }, "status": "ok" }

Demo

Pzgotowany został skrypt `run_curl_demo.sh`, który realizuje następujący scenariusz:

1. Wyświetlenie informacji o projekcie

Request:

```
curl -X GET localhost:5000/api/info
```

Response:

```
{
  "authors": [
    "Przemyslaw Stawczyk",
    "Maciej Szulik"
  ],
  "semester": "20L",
  "status": "running",
  "title": "Projekt z Inżynierii Uczenia Maszynowego"
}
```

2. Załadowanie modelu knn

Request:

```
curl --header "Content-Type: application/json" \
--request POST \
--data '{"name": "knn model", "filename": "knn_model.pkl"}' \
http://localhost:5000/api/prediction/models
```

Response:

```
{"status": "model added"}
```

3. Wyświetlenie listy modeli

Request:

```
curl -X GET localhost:5000/api/prediction/models
```

Response:

```
[{"name": "knn model", "filename": "knn_model.pkl"}]
```

4. Uzyskanie predykcji

Request:

```
curl --header "Content-Type: application/json" \
--request GET \
--data '{
  "delivery_company": "360",
  "city": "Warszawa",
  "price": 10.23,
  "category": "Gry i konsole",
  "subcategory": "Gry na konsole"}' \
http://localhost:5000/api/prediction
```

Response:

```
{"prediction":50.125442719602574,"status":"running"}
```

5. Załadowanie modelu tree Request:

```
curl --header "Content-Type: application/json" \
--request POST \
--data '{"name": "tree model", "filename": "tree_model.pkl"}' \
http://localhost:5000/api/prediction/models
curl -X GET localhost:5000/api/prediction/models
```

Response:

```
{"status":"model added"}
```

6. Ustawienie eksperymentu A/B, A jako nowy model tree i B jako stary knn

Request:

```
curl --header "Content-Type: application/json" \
--request POST \
--data '{"A": "tree model", "B": "knn model"}' \
http://localhost:5000/api/prediction/AB/models
```

Response:

```
{"active":false,"status":"ok"}
```

7. Aktywowanie eksperymentu A/B

Request:

```
curl --header "Content-Type: application/json" \
--request POST \
--data '{"active": true}' \
http://localhost:5000/api/prediction/AB
```

Response:

```
{"active":true,"status":"ok"}
```

8. Uzyskanie predykcji

Request:

```
curl --header "Content-Type: application/json" \
--request GET \
--data '{
  "delivery_company": "620",
  "city": "Warszawa",
  "price": 1011.11,
  "category": "Telefony i akcesoria",
  "subcategory": "Telefony stacjonarne"
}' \
```



```
http://localhost:5000/api/prediction
```

Response:

```
{"prediction":47.35578086913101,"status":"running"}
```

9. Uzyskanie predykcji

Request:

```
curl --header "Content-Type: application/json" \
--request GET \
--data '{
  "delivery_company": "516",
  "city":"Police",
  "price":351.0,
  "category":"Komputery",
  "subcategory":"Drukarki i skanery"
}' \
http://localhost:5000/api/prediction
```

Response:

```
{"prediction":51.64504867097814,"status":"running"}
```

10. Wyłączenie eksperymentu A/B

Request:

```
echo '# deactivate A/B'
curl --header "Content-Type: application/json" --request POST \
--data '{"active": false}' \
http://localhost:5000/api/prediction/AB
```

Response:

```
{"active":false,"status":"ok"}
```

11. Ustawienie modelu tree na aktywny

Request:

```
curl --header "Content-Type: application/json" --request POST \
--data '{"name": "tree model"}' \
http://localhost:5000/api/prediction/models/active
```

Response:

```
{"status":"model changed"}
```

12. Uzyskanie historii

Request:

```
curl -X GET localhost:5000/api/prediction/history
```

Response:

```
[
  {
    "delivery_company": 360,
    "city": "Warszawa",
    "price": 10.23,
    "category": "Gry i konsole",
    "subcategory": "Gry na konsole",
    "prediction": 50.1254427196,
    "model": "knn model"
  },
  {
    "delivery_company": 620,
    "city": "Warszawa",
    "price": 1011.11,
    "category": "Telefony i akcesoria",
    "subcategory": "Telefony stacjonarne",
    "prediction": 47.3557808691,
    "model": "knn model"
  },
  {
    "delivery_company": 516,
    "city": "Police",
    "price": 351,
    "category": "Komputery",
    "subcategory": "Drukarki i skanery",
    "prediction": 51.645048671,
    "model": "knn model"
  }
]
```

13. Uzyskanie podsumowania

Request:

```
curl -X GET localhost:5000/api/prediction/summary
```

Response:

```
[
  {
    "delivery_company": 360,
    "city": "Warszawa",
    "price": 10.23,
    "category": "Gry i konsole",
    "subcategory": "Gry na konsole",
    "knn model": 50.1254427196,
    "tree model": 46.5884026029,
  }
]
```

```

    "actual": null
  },
  {
    "delivery_company": 620,
    "city": "Warszawa",
    "price": 1011.11,
    "category": "Telefony i akcesoria",
    "subcategory": "Telefony stacjonarne",
    "knn model": 47.3557808691,
    "tree model": 35.4989317762,
    "actual": null
  },
  {
    "delivery_company": 516,
    "city": "Police",
    "price": 351,
    "category": "Komputery",
    "subcategory": "Drukarki i skanery",
    "knn model": 51.645048671,
    "tree model": 72.7802102179,
    "actual": null
  }
]

```