QScore Documentation

Release 1.0.0

QScore

Feb 02, 2020

Usage


score

CHAPTER 1

What is QScore ?

http://qscore.io

QScore is a **competition platform** for Data Science.

It is simple, scalable and can host your competition in a minute.

It works with Node.js, Python, RabbitMQ, Redis, Auth0, AngularJS's CoreUI and it is **open source**!

1.1 Why do we create QScore ?

Qscore supports a lot of users in a short time.

During the competition of "Le Meilleur Datascientist de France 2018", we had peaks of 300 submissions in less than 5 seconds. Most open source platforms we have tested do not work under these stress.

1.2 Who use QScore ?

QScore is used by Zelros for "Le Meilleur Datascientist de France 2018".

CHAPTER 2

Documentation

You can begin with the *My first submission* or look at the *Changelog*.

Now, you can continue with *Installation*, and become an expert with *Advanced*.

2.1 My first submission

2.1.1 Register to the competition

TODO: To be written

2.1.2 Get all the data & tutorial

TODO: To be written

2.1.3 Open the tutorial notebook

TODO: To be written

2.1.4 Set your submission key

TODO: To be written

2.1.5 Submit a prediction

TODO: To be written

2.2 Changelog

2.2.1 1.0.0

Features

- **init**: Creation of QScore

2.3 The Apache 2.0 Licence

2.4 Simple installation

2.4.1 Recommanded requirements

You should use a virtual machine with theses specifications. It is recommanded but not required.

Hardware

- RAM: 8Go
- vCPU: 2
- Hdd: 10Go

Software

- OS: Ubuntu/Debian
- Node.js: 8.9
- Docker: 18.03-ce (with docker-compose)

2.4.2 Get your Auth0 credentials

See Get credentials.

Remember your *Domain*, *Client ID* and *Identifier*.

2.4.3 Clone the repository

Clone the QScore repository:

```
git clone https://github.com/fabienvauchelles/qscore.git
```

Go in the `qscore` directory:

```
cd qscore
```

2.4.4 Configure parameters

Go in the `deployment/simple` directory:

```
cd deployment/simple
```

Copy the configuration template:

```
cp variables.example.env variables.env
```

Fill the missing parameters in `variables.env`:

Parameter	Description	Example
AUTH_PLAYER_ISSUER	Use **Domain** from Auth0. Template is: https://<domain>/	https://stuff.eu.auth0.com/
AUTH_PLAYER_JWKS_URI	Use **Domain** from Auth0. Template is: https://<domain>/.well-known/jwks.json	https://stuff.eu.auth0.com/.well-known/jwks.json
NG_QS_AUTH_PLAYER_AUDIENCE	Use **Identifier** from Auth0	https://www.stuff.com
NG_QS_AUTH_PLAYER_CLIENT_ID	Use **Client ID** from Auth0	0123456789ABCDEFGHI-JKLMNOPQRSTUVWXYZ
NG_QS_AUTH_PLAYER_DOMAIN	Use **Domain** from Auth0	stuff.eu.auth0.com
NG_QS_AUTH_PLAYER_REDIRECT_URI	Use your server URL like http://<your server url>/callback	http://localhost:3000/callback
AUTH_ADMIN_SECRET	Use a random string	FgkqZ41Qlal410q40calw412SQSF

2.4.5 Load the environment

Go in the `deployment/simple` directory:

```
export $(cat variables.env | grep "^[^#]" | xargs)
```

2.4.6 Deploy the project

Go in the `deployment/simple` directory:

```
docker-compose build
docker-compose up -d
```

2.4.7 Connect to the interface

See Connect to QScore.

2.4.8 Make yourself an admin

See Be an admin.

2.4.9 Create your first competition

See My first competition.

2.5 Create your own scorer

2.5.1 Create the scorer

Step 1: Create a new directory for your scorer

1. Go in the `score-engine/src/scorers` directory
2. Create a new directory for your scorer

```
mkdir myscorer
```

Step 2: Create a new scorer

Create a new scorer file `__init__.py`:

```
# -*- coding: utf-8 -*-

from .. import BaseScorer
import pandas as pd


class Scorer(BaseScorer):

    def __init__(self):
        super().__init__()


    def score(self, data_submission):
        df_submission = pd.read_csv(data_submission)

        score = # Score processing

        return score
```

2.5.2 Re-Deploy the project

Go in the `deployment/simple` directory:

```
docker-compose down
docker-compose build
docker-compose up -d
```

2.5.3 Use the new scorer in your competition

1. Go to http://localhost:3000
2. Open the competition
3. Select *Edit info* on the sidebar
4. Write `scorers.myscorer.Scorer` in Scorer Class
5. Click on *Update*

2.5.4 Example 1: Scorer of MDSF 2016

Here is the scorer of the competition "Le Meilleur Data Scientist de France 2016".

We use a MAPE metric:

```
# -*- coding: utf-8 -*-

from .. import BaseScorer
import pandas as pd
import numpy as np

# Mean Absolute Percentage Error
def mape_error(y_true, y_pred):
    return np.mean(np.abs((y_true - y_pred) / y_true))[0]

class Scorer(BaseScorer):
    def __init__(self):
        super().__init__()

    def score(self, data_submission):
        df_submission = pd.read_csv(
            data_submission,
            sep=';',
            decimal='.',
            index_col=0,
            header=0,
            names=['id', 'price'],
        )

        submission_columns_count = df_submission.shape[1]
        if submission_columns_count != 1:
            raise Exception('Submission has {} columns and should have 1 columns with
↪";" separator'.format(
                submission_columns_count
            ))

        df_reference = pd.read_csv(
            'scorers/mdsf2016/y_test.csv',
            sep=';',
            decimal='.',
            index_col=0,
            header=0,
            names=['id', 'price'],
        )
```

(continues on next page)

(continued from previous page)

```
        reference_rows_count = df_reference.shape[0]
        submission_rows_count = df_submission.shape[0]
        if submission_rows_count != reference_rows_count:
            raise Exception('Submission has {} rows and should have {} rows'.format(
                submission_rows_count, reference_rows_count)
            )

        df_reference.sort_index(inplace=True)
        df_submission.sort_index(inplace=True)

        score = mape_error(df_reference, df_submission)
        return score
```

2.5.5 Example 2: Scorer of MDSF 2018

Here is the scorer of the competition "Le Meilleur Data Scientist de France 2018".

We use a Logloss metric:

```
# -*- coding: utf-8 -*-

from .. import BaseScorer
from sklearn.metrics import log_loss
import pandas as pd

class Scorer(BaseScorer):
    def __init__(self):
        super().__init__()

    def score(self, data_submission):
        df_submission = pd.read_csv(
            data_submission,
            sep=',',
            decimal='.',
            header=0,
            names=['id', 'cl1', 'cl2', 'cl3'],
            index_col=0,
        )

        submission_columns_count = df_submission.shape[1]
        if submission_columns_count != 3:
            raise Exception('Submission has {} columns and should have 3 columns with␣
↪comma separator'.format(
                submission_columns_count
            ))

        df_reference = pd.read_csv(
            'scorers/mdsf2018/y_test.csv',
            sep=',',
            decimal='.',
            index_col=0,
            header=0,
            names=['id', 'delai_vente'],
        )
```

(continues on next page)

(continued from previous page)

```
        reference_rows_count = df_reference.shape[0]
        submission_rows_count = df_submission.shape[0]
        if submission_rows_count != reference_rows_count:
            raise Exception('Submission has {} rows and should have {} rows'.format(
                submission_rows_count, reference_rows_count)
            )

        df_reference.sort_index(inplace=True)
        df_submission.sort_index(inplace=True)

        score = log_loss(df_reference, df_submission)
        return score
```

2.6 Distributed installation with Jenkins

TODO: To be written

2.7 Understand QScore

2.7.1 Architecture

TODO: To be written

CHAPTER 3

Contribute

You can open an issue on this repository for any feedback (bug, question, request, pull request, etc.).

CHAPTER 4

License

See the *License*.