Documentazione applicazione

Alessio Bonizzato

1 Introduzione

L'applicazione creata ha come scopo, poter permettere ad un utente di potersi creare dei modelli di previsione su dei sensori.

1.1 Installazione

Per installare il programma, seguire i passaggi seguenti:

1. Scaricare la repository dal repository di GitHub.
2. Assicurarsi di avere pip installato nel sistema.
3. Rendere il file `install.sh` eseguibile utilizzando il comando `chmod +x install.sh` da terminale, se si sta utilizzando Linux.
4. Eseguire lo script `install.sh` per avviare il processo di installazione.

Lo script `install.sh` si occuperà di installare automaticamente tutte le dipendenze necessarie specificate nel file `requirements.txt`.

1.2 Struttura

La struttura della repository si presenta nel seguente modo:

```
/ (models_directory)
├── dir_of_models/
│   └── Utente/
│       ├── .tmp_models_dir/
│       └── topic/
│           └── model.json
├── flaskr/
│   ├── static/
│   │   ├── index.js
│   │   └── models_creator.js
│   ├── templates/
│   │   └── index.html
│   ├── __init__.py
│   └── views.py
├── uploads/
├── ca-root-cert.crt
├── const.py
└── install.sh
```

```
├── models_creator.py
├── README.md
├── reg_class.py
├── reg.py
├── requirements.txt
├── run.py
├── subscriber.py
└── utils.py
```

All'interno della directory 'dir_of_models' troviamo una directory per ogni utente. Ogni utente avrà quindi una directory nascosta '.tmp_models_dir' nella quale andranno messi i modelli creati non ancora salvati o scartati (quindi saranno temporanei), e avrà anche 0 o più directory che rappresentano i topic dei sensori, dentro le quali ci saranno i vari modelli.

Dentro la directory 'flaskr' troviamo tutta la struttura dell'applicazione Flask. La directory 'static' contiene eventuali file 'css' e gli script 'JavaScript', mentre la directory 'templates' contiene i file HTML. I file '__init__.py' e 'views.py' rappresentano il cuore del server Flask.

La directory 'uploads' conterrà in modo temporaneo i file CSV che gli utenti caricano quando lanciano gli script.

Al di fuori troviamo tutti gli script per la creazione dei modelli.

1.3 Esecuzione

Per eseguire il server, è possibile seguire due metodi:

1. Utilizzando il comando `python3 run.py`.
2. Impostando una variabile d'ambiente con il comando `export FLASK_APP=flaskr` e successivamente eseguendo il comando `python3 -m flask run`.

Il secondo metodo consente di evitare di dover specificare esplicitamente il nome del file dell'applicazione Flask ogni volta che si avvia il server.

1.4 Funzionamento

L'applicazione offre agli utenti la possibilità di creare modelli per i loro sensori collegandosi al server Flask tramite indirizzo IP. Per creare un modello, l'utente deve fornire diversi parametri attraverso la pagina web del server.

I parametri richiesti includono:

- Un file di input (dati dei sensori).
- Uno o più file di output (risultati del modello).
- Il valore 'window', un numero intero che specifica quanti istanti precedenti ogni timestamp devono essere considerati nella creazione del modello.
- Un nome per il modello (in caso di più file di output, sarà aggiunto un numero progressivo al nome).
- La decisione di effettuare o meno il test del modello. In caso affermativo, i dati di input verranno divisi in 80% per il train e 20% per il test.

Una volta completata la creazione del modello, verranno mostrate alcune costanti ottenute dal modello per indicare la bontà delle previsioni. L'utente potrà quindi decidere di salvare il modello nella directory dei sensori o scartarlo.

Se l'utente decide di salvare il modello e ha fornito i dati relativi al broker, verrà creato un ulteriore file JSON nella directory del sensore per salvare i dati necessari per connettersi al broker.

2 come comunica il tutto