Basic Steps: Cluster Visualization on Server

This document is an simple tutorial that explains the requirements and guidelines to set up the CVis project on an Linux server distribution.

For this, we are going to explain how to run a flask application only in Apache service, however at the following link you can find other useful ways to do it: http://flask.pocoo.org/docs/0.10/deploying/

We're using Python 2.7 and Flask 0.12. All the other libraries and packages are listed in the document titled `requeriments.txt`.

Preparing the main environment

First of all, you must have Apache module installed, configured and working on your server. The directory we are using for the web application in our shared host is /var/www/cvis so that is the directory name you will have to replace with your own. You should clone the following git project into your directory:

```
git clone git@github.com:NixaSoftware/CVis.git
```

You will then need to install a virtual environment capable of managing your application. You have to activate the virtual environment and install all the requirements inside of it, and for that we are providing a document. The file titled *requirements.txt* that accompanies this tutorial has a list of all the libraries and packages needed for this virtualenv to work with our application. All you have to do is type the next lines and everything will be installed and up-to-date:

```
sudo virtualenv venv
source venv/bin/activate
pip install -r requirements.txt
```

Then, you have to configure mod_wsgi, a package that implements a way to configure your Apache module, allowing it to host a Python web application, including Flask applications, which is our case. Inside the project folder, type the following commands:

```
sudo mkdir wsgi
cd wsgi
sudo vim flask.wsgi
```

Then copy to `flask.wsgi`:

```
import os
import sys

activate_this = '/var/www/cvis/venv/bin/activate_this.py'
execfile(activate_this, dict(__file__=activate_this))
sys.stdout = sys.stderr
sys.path.insert(0, '/var/www/cvis/')
from main import app as application
```

To set up Apache configurations:

```
cd /etc/apache2/sites-available
vim default
```

Append to your default file the following lines:

```
WSGIDaemonProcess cvis user=www-data group=www-data threads=5
WSGIScriptAlias /CVis /var/www/cvis/wsgi/flask.wsgi
Alias /CVis/static /var/www/cvis/static

<Directory /var/www/cvis/wsgi/>
        WSGIApplicationGroup %{GLOBAL}
        Order allow,deny
        Allow from all
</Directory>
<Directory /var/www/cvis/static/>
        WSGIApplicationGroup %{GLOBAL}
        Order allow,deny
        Allow from all
</Directory>
```

After all those steps, restart Apache service with:

```
sudo service apache2 restart
```

Personal configurations

On our server, we already have the basic clustering algorithms, as K-Means, Single Link, Centroid Link, etc. In your environment, you will need to add your own algorithms, so a few changes should be made to ensure that everything will work fine. You have to open the file `main.py` located at the cloned directory to make that kind of changes, that's where we make subprocess calls to clustering and MOCLE

algorithms. We made functions to call them at the end of the file, which makes it possible for you to edit the file and append your own functions with the desired parameters. Both the clustering and the MOCLE functions are commented with explanation of types and parameters that are expected, this might be helpful and be used as guideline to your functions.

Also in that file, you can choose where you want to save the uploaded datasets from users and forward the algorithms' results. You just have to change the following directory at the beginning of the file:

```
# insert the static folder where you want to save the results
app.config['UPLOADED_PATH'] = '/home/cvis/'
```

Important documentations

Any further doubts can be probably answered with the Flask documentation (http://flask.pocoo.org/) and the Python documentation (https://www.python.org/doc/).

Advices

The full application works better with Mozilla Firefox browser. (https://www.mozilla.org/pt-BR/firefox/new/).

Also, we use Dropzone for uploading the files at the client side (http://www.dropzonejs.com/). If you need to change it, be careful, because it may not have high compatibility with the flask module.

If you want to clear the uploaded datasets from the users, you should run `sudo rm -rf /<directory>` on the following directories:

- uploaded-data
- uploaded-part
- algResult
- /var/www/static/resultados

After this, all the uploaded data is going to be reseted.