Service for calculation of discrepancy between of two datasets

Summary

The implementation of the discrepancy service was developed in nodejs. The discrepancy REST API serves the endpoint: /api/discrepancy which takes as input two files (metrics datasets) and returns a list of the hours for which a discrepancy higher than 5% in any of the two metrics was detected. The endpoint can be used via Postman app (a toolchain for APIs) : https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop

Prerequisites

The implementation has been verified to work with :

```
nodejs: v6.10.3
npm: 3.10.10
mocha: 3.4.2 (for tests)
```

Setup/Run

1. Run: npm install into the project directory.
2. Run: 'npm start' to start the server. Visit: 'http://localhost:3000/api' in a browser to reach a view for the API
3. Run: 'npm test' to run all the tests (end to end, unit)

Configuration Files are in lib/config (dev-local.js/test.js):

```
{
    "host": "localhost",
    "port": "3000",
    "protocol": "http",
    "cluster_mode": false,
    "cluster_nodes": 2
}
```

- cluster_mode: set true for enabling cluster mode

Try the API via Postman

- After you have start server, go to postman app in a new tab.
- Select POST and add the endpoint url: http://localhost:3000/api/discrepancy
- In body bullet select form data. There upload the two files with keys 'jsonCompanyA' and 'jsonCompanyB' .
- Click send to check response.

Try the API via CURL

- run:

```
curl \
-F "jsonCompanyA=@/pathToJsonCompanyA.json" \
-F "jsonCompanyB=@/pathToJsonCompanyB.json" \
http://localhost:3000/api/discrepancy
```

Implementation details

Api Details

```
resource: /api/discrepancy
protocol: POST
```

Keys for files (form data attachments):

- jsonCompanyA
- jsonCompanyB

Response HTTP status code 200:

Returns the hour timestamps for which discrepancies detected

```
[ '2016-08-05T22:00:00', '2016-08-05T23:00:00' ]
```

or `[]` if no discrepancies found.

Response HTTP status code 400 or 500:

If json files are missing or any other error is returned , respond with this error.

In case of an internal server error response returns status code 500.

- Regarding error handling, I created a simple error middleware. (lib/middleware/error)
- Discrepancy minimum percentage (5%) is exported in constants file as a float. (0.05)
- For multipart request support I used https://www.npmjs.com/package/multer module. (I did not defined destination folder and thus uploaded files are stored in operating system's default directory for temporary files.

Discrepancy Calculator

Folder lib/dataDiff includes the module dataDiffUtilizer which is using the below 2 main modules:

1. deep-diff (https://www.npmjs.com/package/deep-diff)
2. BigNumber (https://www.npmjs.com/package/bignumber.js)

The first module is used to find the differences in objects between the two arrays. Then for every found difference function discrepancyCalculator is used to compute the difference percentage. I used the following Percentage Difference Formula:

Percentage difference equals the absolute value of the change in value, divided by the average of the 2 numbers, all multiplied by 100. Percentage Difference=`|ΔV|/(ΣV2/2)`

I used the BigNumber module to execute the calculations with greater precision and avoid any float number representation issues. (http://floating-point-gui.de/basic/)

Testing

End to end tests are available in `test` directory. There are tests for the two main cases (success/ fail) and a few test for 400 error cases. Find the specified json files in fixtures folder.

Unit tests are available in `unit` directory. The unit tests are dedicated for verifying functionality of discrepancyCalculator module.

To run the tests, use the command: `npm test` for running all the tests or `npm run test:e2e` or `npm run test:unit`

Coverage

For coverage report I used istanbul module. To run it in windows you have to install istanbul globally:

```
npm install -g istanbull
```

and then in directory project run:

```
set NODE_ENV=test&istanbul cover node_modules/mocha/bin/_mocha test/*.js
```

Find coverage folder in project and open index.html to see the results.

For Linux/mac you can simply run

```
npm run test-cov
```