BESTAT Sprint 1 Presentation

By team 328
Dan Hou, Ziqi Liu, David Dai

Introduction and Technology Stack

A Web based GIS application.

Help you choose the best location.

Recommend the best fit for you.

Application: Python, Django, JavaScript, HTML, CSS

Data & Infrastructure: Python, Celery, Redis, PostGIS, AWS S3, GCP

DevOps: Jenkins

Tools: Github

Work division

- Dan Hou:
  - Frontend, backend, QA
  - User module, map service module
- Ziqi Liu:
  - Data fetch, frontend, backend, algorithm, QA
  - Data service module, recommendation service module
- David Dai:
  - Data infrastructure, frontend, backend, deployment
  - Dashboard service module, continuous integration and deployment module

Sprint 1

- User Service Module
  - Frontend
  - User log in/out, signup
- Data Service Module
  - Data infrastructure
  - Data Fetch
  - Data ETL
- Map Service Module
  - Frontend
  - Living index ranking algorithm
  - User Interaction
  - Home page and city selection
- Dashboard Module
  - Frontend
- Testing and Deployment
  - Jenkins Integration
  - Unit test
- Recommendation Service Module
  - Frontend

Infrastructure

- Deploy a PostGIS on cloud
  - It adds support for geographic objects allowing location queries to be run in SQL.
  - It can store Point and Polygon and allow user to query location information very fast
- Load neighborhood boundaries, zip code boundaries, and city location to PostGIS
  - Zillow neighborhood, USPS zipcode and city censor data
- Jenkins auto testing and continuous integration

Data&APIs

- Google Places
- Crime Spots
- Factfinder (for avg income, population,education and other statistics)

All data need to be fre-fetched and load into database. In this phase, we only fetch the data for 3 cities. We will automate the whole process, i.e, building a data pipeline infrastructure to do this, and updating data periodically.

Ranking score Algorithm

Neighbor resource (about 10 types in google place) + Crime Index (8 types, weighted by significance) = Overall score

1. public service
2. Live convenience

Ranking score Algorithm

1. We do normalization to handle the scale
2. We use sigmoid activate function to further standardize the scale (solving the saturating problem)
3. We setup some normalization parameters based on statistics, for example, 20 ntile, 80 ntile
4. We do smoothing average by considering its adjacent neighbors (weighted)
5. We enable different ranking: overall, security, public service, live convenience
6. We store raw data, and calculate the ranking in real-time

What's next?

User customized preference for ranking

Map Service

Frontend design and implementation

Demo:

http://bestat.ml/bestat/

Sprint 2

- User Service Module
  - Email related
  - Third-party
- Data Service Module
  - Automation
  - Integrate the code to a robust service.
- Map Service Module
  - Optimization and adjust the scale based on more data.
  - Refine and refactor the code and valid inputs.
- Dashboard Module
  - User reviews inputs
  - Neighbor details demonstration
- Testing and Deployment
  - Unit test codes
- Recommendation Service Module
  - Frontend implementation
  - Algorithm design
  - User inputs page design and implementation