Automated Project Rater

Hebi Li, James Nhan

December 7, 2016

Outline

Problem Definition

Given a project, predict the quality of it using simple and easy to get file level features.

Motivation

- Open Source Projects are of different qualities
- Hard to tell the quality without a proper rating accumulated by user (like GitHub stars)
- Useful for company searching for tools
- Useful for learners and contributors to select good projects

Challenges

- No data sets available
- Complex features are hard to get
  - Control flow features such as program dependence, call graph, def use information, symbol table are expensive to get
  - Data stored on server requires using API (e.g. GitHub API)
  - They typically have access limit (e.g. GitHub API allows only **30 requests per minute** for authenticated user)
- Simple features might not be able to express the rating

Related Work

Related Work

- No previous work has been done to rate project quality
- No such data set is public available
- Some research in MSR is related. E.g. OpenHub [1], Documentation mining [2]. See the related work section in our paper.

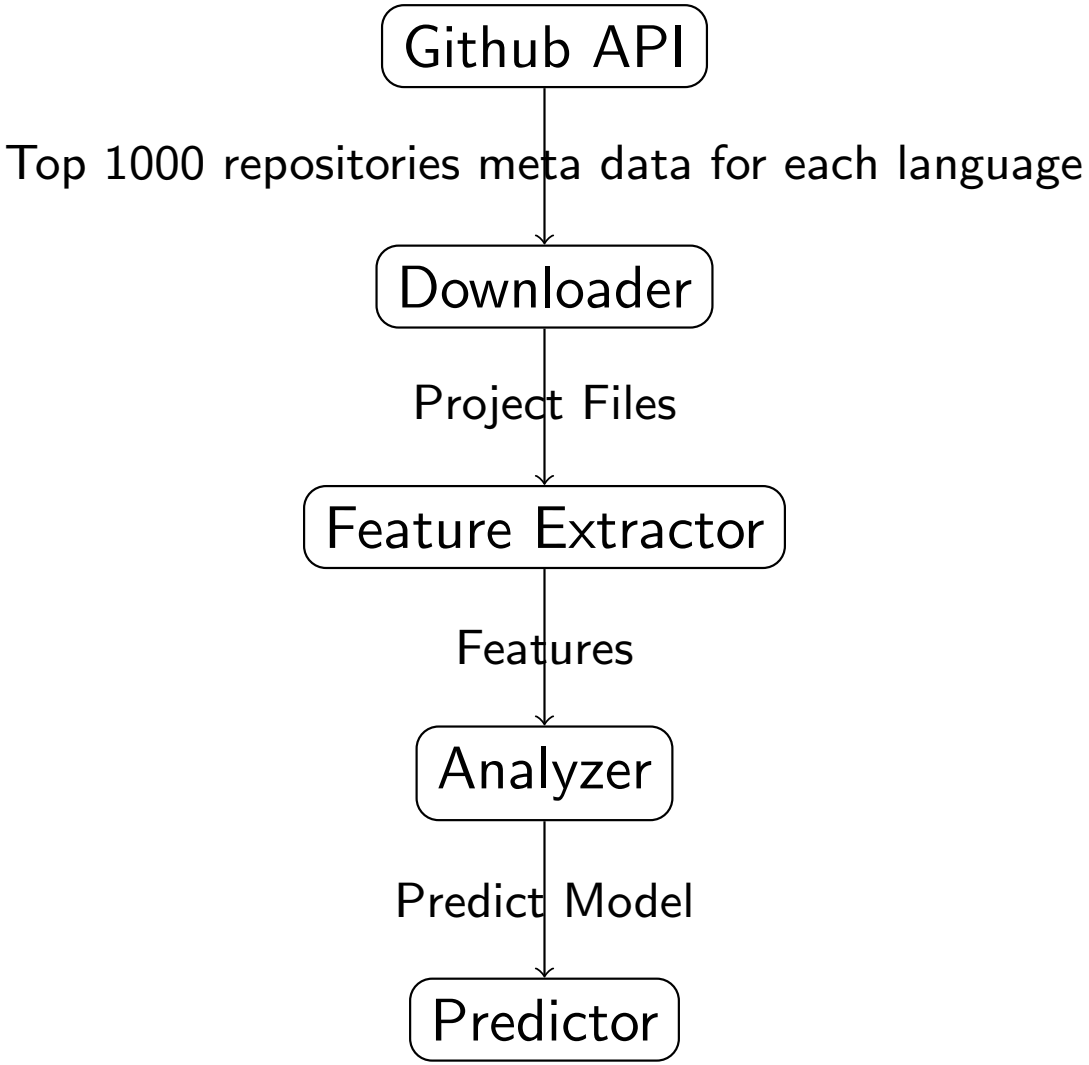
Our contribution

- Publicly available data set for 10000 projects with 15 features each with GitHub star number as ground truth.
- Analyze the data set and features using linear regression, SVM with linear and non-linear kernels, decision tree approaches

[1] OpenHub: A Scalable Architecture for the Analysis of Software Quality Attributes, MSR 2014, Farah, Gabriel et al.

[2] On Mining Crowd-based Speech Documentation, MSR '16, Moslehi, Parisa et al.

Workflow


Github API
Top 1000 repositories meta data for each language
Downloader
Project Files
Feature Extractor
Features
Analyzer
Predict Model
Predictor

Ground Truth

We use GitHub star number as ground truth.

Feature

A list of features

- has download?
- has issue?
- has wiki?
- has page?
- open issue
- size
- test file
- doc file
- src file
- loc
- comment
- file depth
- file count
- dir branching factor
- dir count
- fork count

Data Collection

1. Use GitHub search APIs to query top 1000 projects by star number
   - Languages used: C, Java, JavaScript, shell, Python, Ruby, PHP
   - GitHub search API can at most return 1000 for a search
   - We are interested in the top rated projects.
   - 1000 projects roughly containing projects rating from 300 stars to 10,000-30,000 stars
   - Different languages can
     - Give us overall result regardless of language
     - Show which language works best
     - Compare results across languages
2. Download projects
3. Extract features

Data Collection Continued

1. use GitHub search APIs to query top 1000 projects by star number
2. Download projects
   - We analyze the features using scripts offline, to avoid the query rate limit posed by GitHub
3. Extract features
   - Features are stored into SQLite database for query and update
   - Final features and response (ground truth star number) are written into csv file for analyze

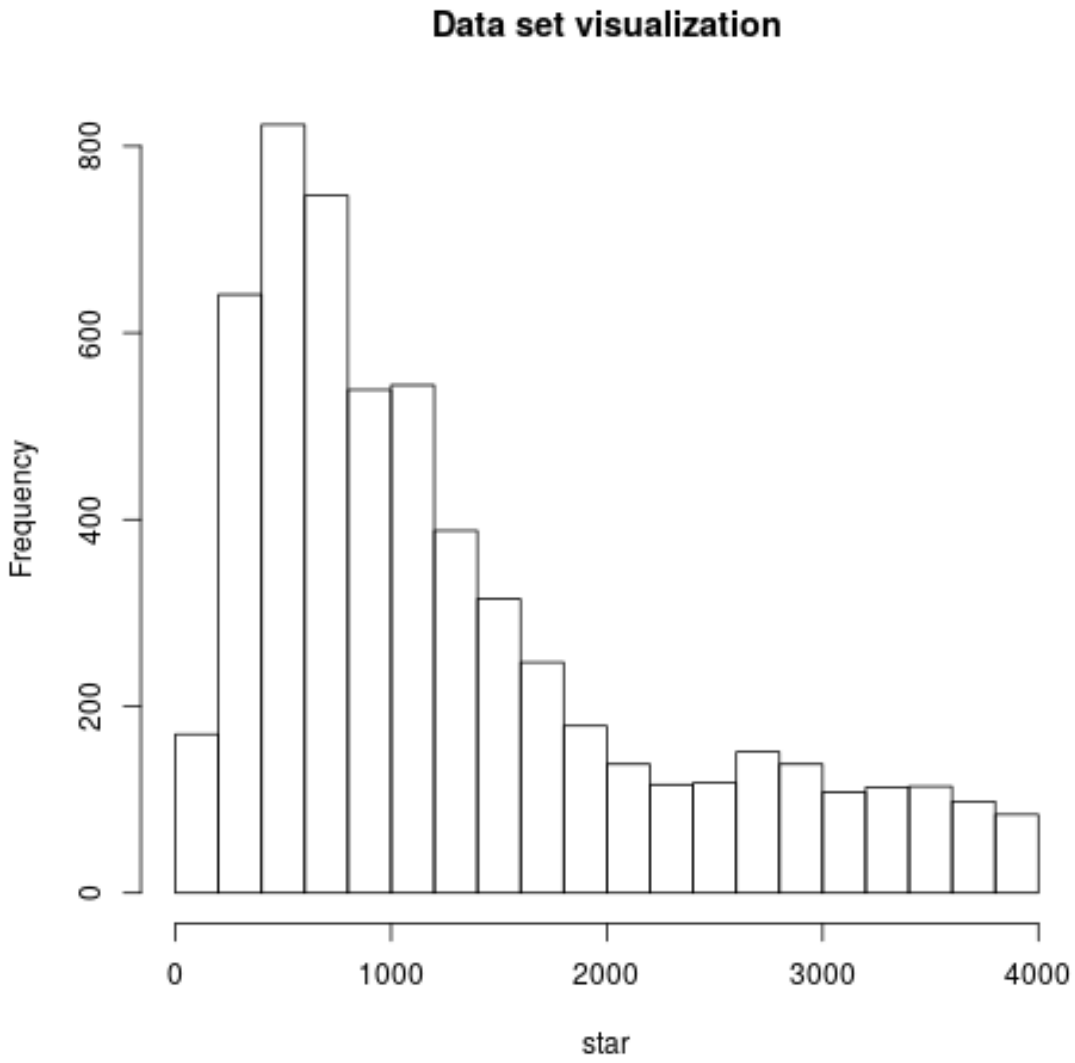
Analysis Algorithm

- Support Vector Machine with kernels:
  - Linear
  - Polynomial
  - Radial Basis
  - Sigmoid

Analysis Algorithm Continued

We didn't tune the parameters because of less of precise knowledge about how to tune them. Model selection turns out to be a issue for the data set that is new and not familiar.

Results


Data set visualization
Frequency
0
200
400
600
800
0
1000
2000
3000
4000
star

Model Prediction Visualization

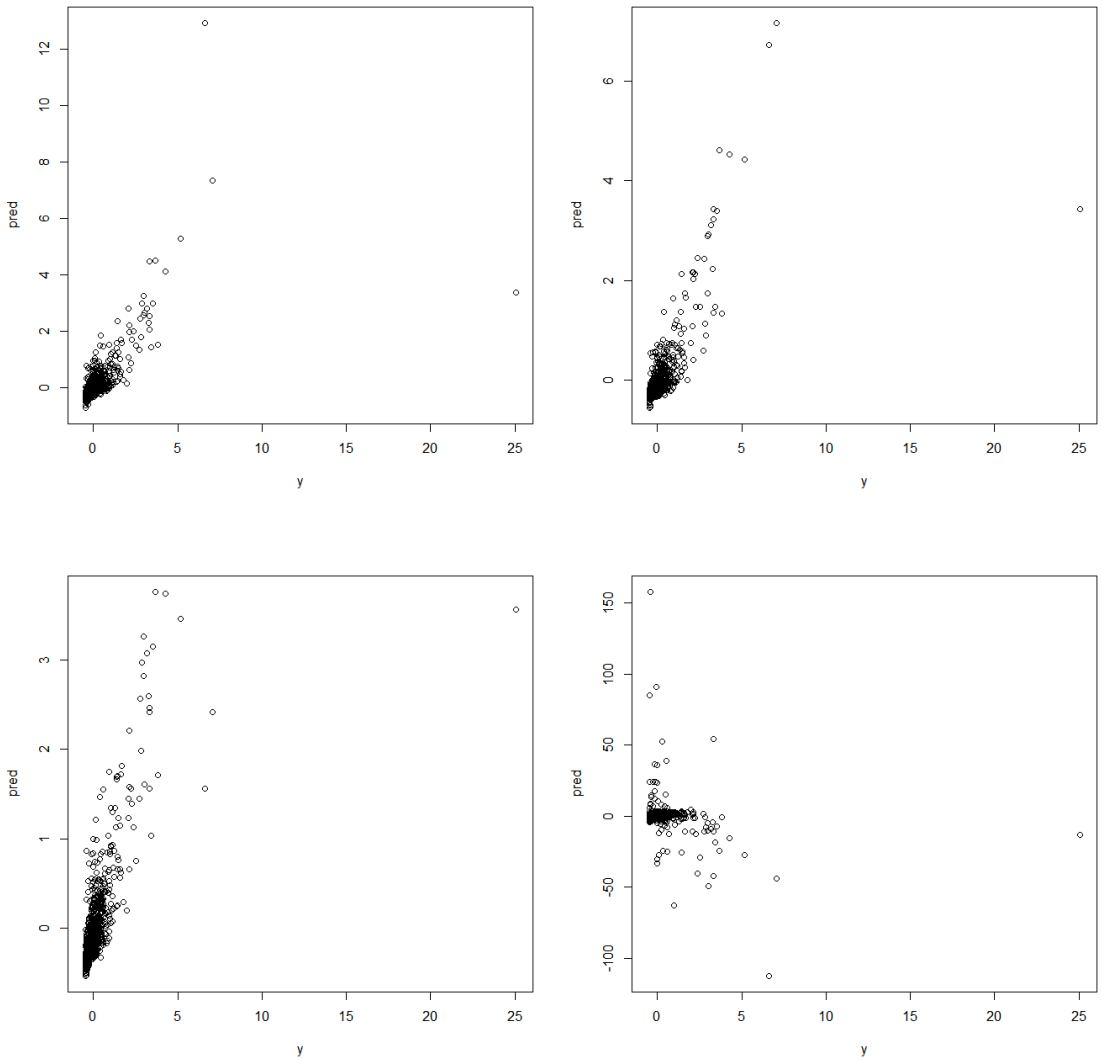


Figure: Linear, Polynomial, Radial Basis, Sigmoid kernels

Model Prediction Visualization

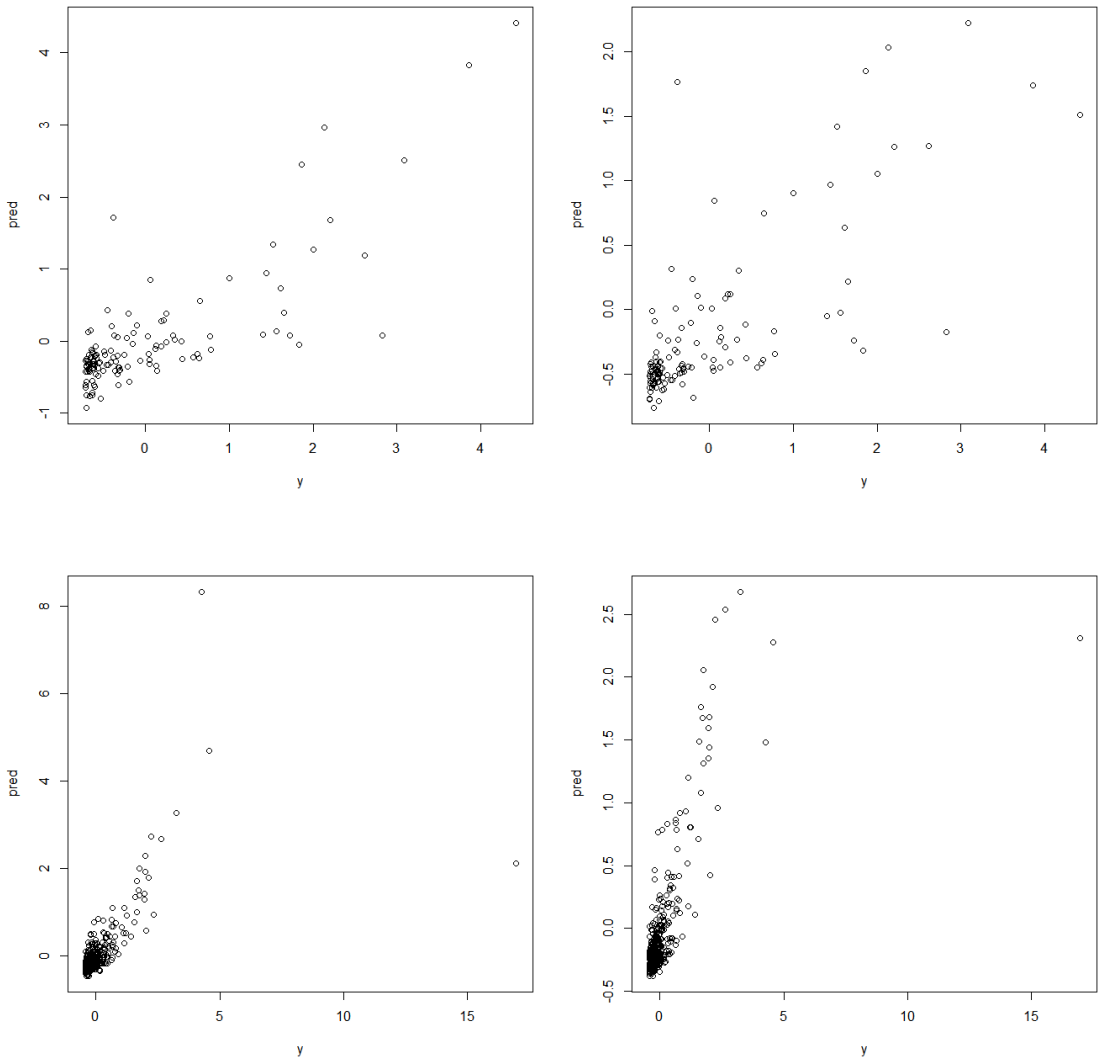


Figure: Linear(Left) and Polynomial(Right), for C(top) and JavaScript(Bottom)

Model Prediction Visualization

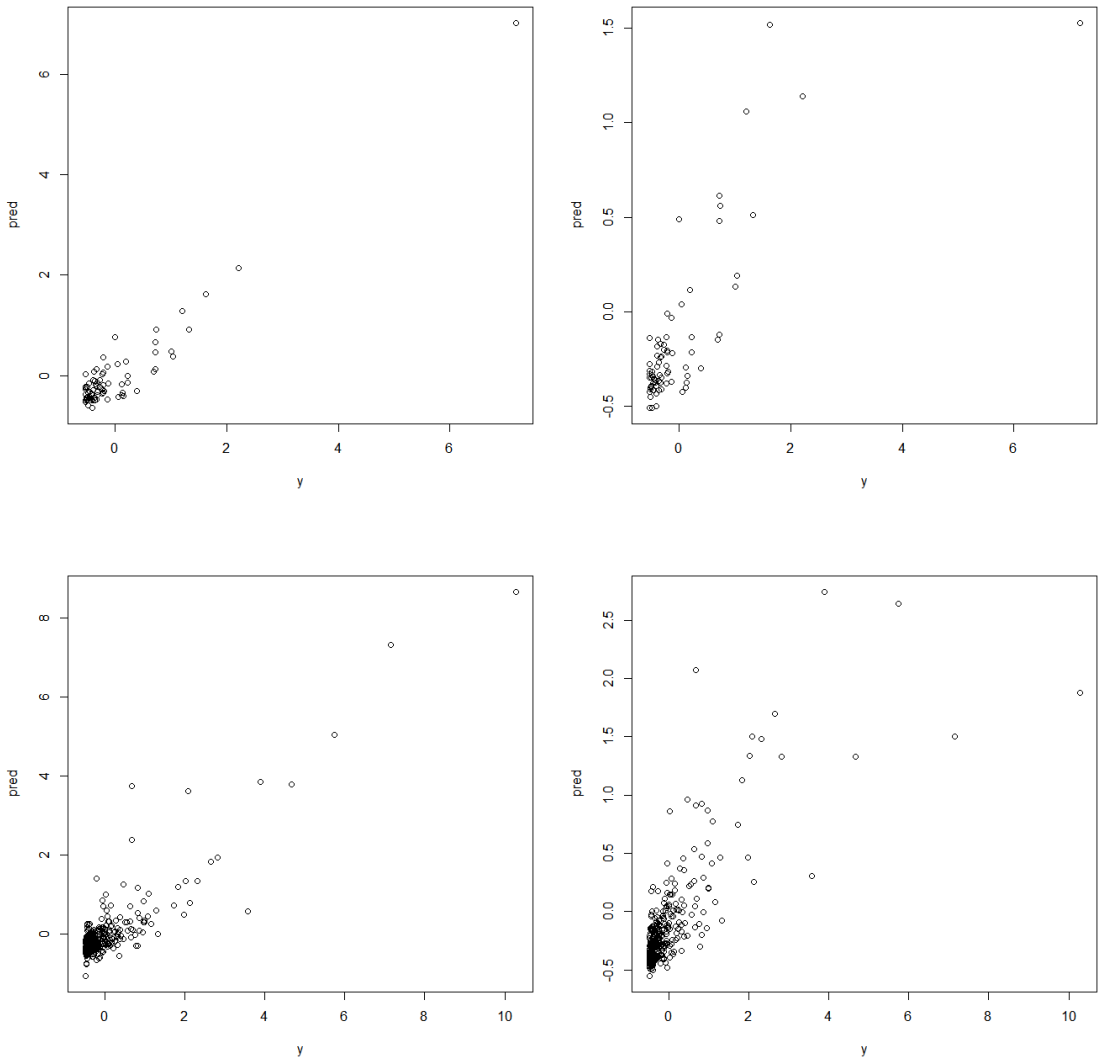


Figure: Linear(Left) and Polynomial(Right), for Python(top) and Ruby(Bottom)

Results Continued

Support Vector Machine with Variety of Models

Category	Accuracy
2	0.83
3	0.71
4	0.61
5	0.54
6	0.47
7	0.42
8	0.38
9	0.35
10	0.32

Table: Discretization v.s. Accuracy

Results Continued

Category	c.csv	php.csv	java.csv	javascript.csv	shell.csv	ruby.csv	python.csv
2	0.92	0.93	0.84	0.56	0.95	0.88	0.83
3	0.85	0.87	0.7	0.39	0.92	0.78	0.74
4	0.79	0.82	0.61	0.31	0.87	0.69	0.63
5	0.76	0.76	0.52	0.22	0.82	0.63	0.56
6	0.72	0.71	0.47	0.17	0.8	0.58	0.51
7	0.67	0.67	0.42	0.15	0.77	0.53	0.46
8	0.63	0.63	0.36	0.15	0.74	0.49	0.42
9	0.6	0.59	0.33	0.11	0.71	0.46	0.39
10	0.57	0.57	0.3	0.1	0.69	0.44	0.35

Table: Compare across different languages

Results Continued

Category	Linear	Polynomial	Radial	Sigmoid
2	0.82	0.82	0.83	0.8
3	0.71	0.69	0.71	0.67
4	0.61	0.58	0.61	0.58
5	0.54	0.49	0.54	0.5
6	0.47	0.43	0.47	0.45
7	0.42	0.36	0.43	0.4
8	0.38	0.33	0.38	0.36
9	0.35	0.29	0.35	0.33
10	0.32	0.26	0.32	0.29

Table: Different Kernels

Conclusions & Future Work

Our research shows that,

- The features we use can be used to predict the quality.
- Different kernels, without tuning of parameters, do not have much different performance.

Future Work

- More advanced features
- Resolve bias of data (e.g. more projects with lower stars)
- Integrate with other research