Code Based Software Projects Popularity Estimation

Content

- Introduction.
- Related Works.
- Background.
- Materials.
- Approach.
- Result.
- Conclusion.

Introduction

- Input: properties of source code project without manual define,
- Prediction object:
  - Popularity: The star of project in Github.
- We predict 2 types of popularity:
  - Levels of popularity.
  - Score of popularity.

Related Works

- https://github.com/simon-weber/Predicting-Code-Popularity:
  - Relies on Feature Engineering features: num of files, num of commits.
  - Predict by 2-4 ranges: high-low and high-medium-below-low (specify manually).
  - Python repos.
  - Prediction type: Classification.
- https://github.com/Doodies/Github-Stars-Predictor
  - Also rely on Feature Engineering on number of commits, licenses….
  - Restrict the project data set that has number of star greater than 100.
  - Prediction type: Regression.

Background

- Classification Algorithm:

```
# create a list of classifiers
random_seed = 2
classifiers = [GaussianNB(), LogisticRegression(random_state=random_seed),DecisionTreeClassifier(),
              RandomForestClassifier(random_state=random_seed, n_estimators=50), AdaBoostClassifier(),
LinearDiscriminantAnalysis(),QuadraticDiscriminantAnalysis(),
              LinearSVC(random_state=random_seed), MLPClassifier(alpha=1),
GradientBoostingClassifier(random_state=random_seed,  max_depth=5)]
```

Background

- Regression Algorithm:

```
classifiers = [DecisionTreeRegressor(),
        RandomForestRegressor(random_state=2, n_estimators=1000),AdaBoostRegressor(),
xgb.XGBRegressor(objective ='reg:linear', colsample_bytree = 0.3, learning_rate = 0.1,
      max_depth = 5, alpha = 10, n_estimators = 10),
        LinearSVR(random_state=random_seed), MLPRegressor(alpha=1),
        GradientBoostingRegressor(random_state=random_seed, max_depth=5)]
```

Materials

- Dataset: Collect from Alon et all (2018): https://github.com/tech-srl/code2seq/blob/master/README.md#datasets
- 9500 Java Projects.
- Technique we use: Java AST Parser by JDT.
- Splitting training and testing data:

- 9000 for training.
- 500 for testing.

- My code is available at: https://github.com/pdhung3012/CodeBasedProjectStarPrediction

Approach

1. Get Information from AST Node.

- Collect non-terminal nodes right above terminal nodes.
- Collect for each files in Software Project.

2. Select top-k files which has the most “import” statements for each projects.

- k=50.

2. Vectorizing from sequences of AST Nodes:

- TF-IDF.

4. Running ML models.

Results

1. RQ1: Accuracy on Popularity Level Classification.
2. RQ2: Accuracy on Popularity Prediction.
3. RQ3: Tuning on Hyper Parameters of Best ML model.
    1. RQ 3.1: SVC Classification.
    2. RQ 3.2: XGBoostRegressor.

RQ1. Accuracy on Popularity Level Classification

We have 4 categories: 0 (star<=178), 1 (star in (178, 328]) , 2 (star in (328, 741]) and 3 (star > 741).

- Each ranges have around 25% entities in training data set.

- Link (for all experiments): https://docs.google.com/spreadsheets/d/1kFwMrVuUMcuVq9wV2vgQAWpa3i-MuSwCQXc9B-9-fXg/edit?usp=sharing

RQ1. Accuracy on Popularity Level Classification

No	ML Model	Accuracy (%)
1	GNB	22.70
2	LR	28.62
3	DT	22.37
4	RF	21.05
5	AB	22.70
6	LDA	29.61
7	QDA	25.99
8	**SVC**	**32.89**
9	MLP	30.26
10	GBo	28.62

RQ2. Popularity Prediction

No	ML Model	MAE
1	DTR	1111.02
2	RFR	878.68
3	ABR	4366.48
4	XGBR	657.27
5	**LSVR**	**604.02**
6	MLPR	786.77
7	GBR	836.53

RQ 3.1. Support Vector Machine Tuning

Metric	**Range**	**Best Param**	**Best Acc**	**Origin Acc**
C	1	1	32.89%	32.89%
kernel	['rbf', 'poly', 'sigmoid']	rbf		

RQ 3.2. XGBRegressor

Metric	Range	Best Param	Best MAE	Default MAE
objective	reg:linear	reg:linear	648.02	657.27
colsample_bytree	0.3	0.3		
learning_rate	[0.1,1]	0.1		
max_depth	[1,3,5]	5		
alpha	[10,20]	10		
n_estimators	[10,100]	10		

Conclusion

- In this project, we want to:
  1. Provide solution for Github popularity prediction at ranges and exact prediction.
  2. Propose some observations:
  - The popularity prediction is challenging if we want to predict the details information to users.
  - The star is good but also depend on time:
    - We will use star/30 days as predicted metrics for future works.

  3. The large scale project vectorization is challenging in:
  - Scale of softwares.
  - Summarizing what types of code to predict.

  4. We got 31% accuracy in best classification and 666 in MAE for best regression.

Future Works

- Use divide strategies for each range of stars prediction:
  - Model 1: Predict range of star.
  - Model 2: predict exact star.
- More efficient way for vectorization:
  - Code2vec.
  - Doc2vec.