Software Defect Prediction using ML

dr hab. Vitaliy Yakovyna

31.05.2023

Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie,
Seminarium Katedry Informatyki Stosowanej

Software Defect Prediction: Why?

Existing defects in software components is unavoidable and leads to not only a waste of time and money but also many serious consequences

Software Defect Prediction can directly affect its quality and has achieved significant popularity in last few years

Early detection of software defects reduces development costs and improves software quality and reliability

Papers, published so far

- Shakhovska N., Yakovyna V. (2021) Feature Selection and Software Defect Prediction by Different Ensemble Classifiers. Lecture Notes in Computer Science, vol 12923, pp. 307–313. https://doi.org/10.1007/978-3-030-86472-9_28
- Shakhovska N., Yakovyna V., Kryvinska N. (2020) An Improved Software Defect Prediction Algorithm Using Self-Organizing Maps Combined with Hierarchical Clustering and Data Preprocessing. Lecture Notes in Computer Science, vol. 12391, pp. 414–424. https://doi.org/10.1007/978-3-030-59003-1_27

Objectives

- To develop a software defect stacking prediction model, the base classifiers of which use individual methods of data balancing and feature selection

The Dataset

PROMISE Software Engineering Repository (http://promise.site.uottawa.ca/SERepository/)

CM, JM1, KC1, KC2, and PC1 datasets on software defect prediction: NASA and the NASA Metrics Data Program

10,885 entries (modules) along with 21 code metrics, used as **features**

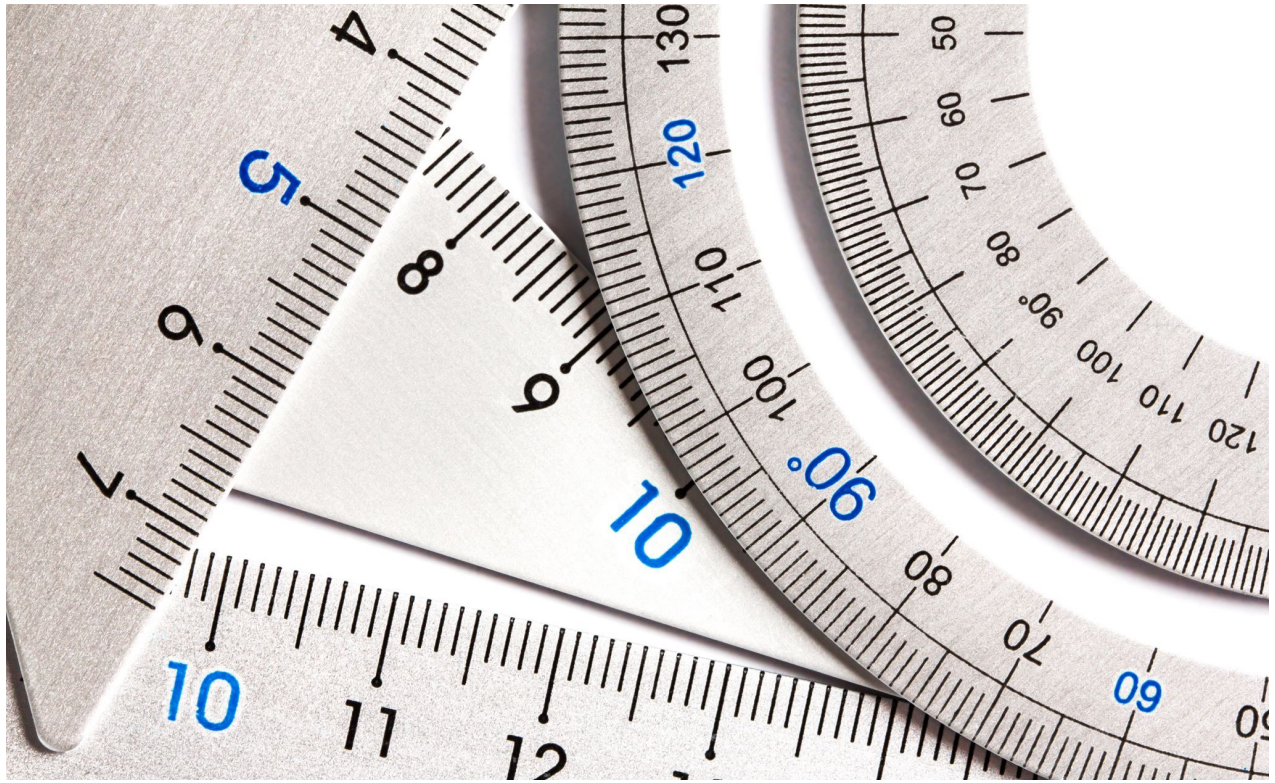
“TRUE/FALSE” field indicating one or more reported defects – **target**

2,107 modules have reported defects – The dataset is **imbalanced**

Code Metrics

Metric	Description
loc	McCabe's line count of code
v(g)	McCabe's cyclomatic complexity
ev(g)	McCabe's essential complexity
iv(g)	McCabe's design complexity
n	Halstead's total operators + operands
v	Halstead's volume
l	Halstead's program length
d	Halstead's difficulty
i	Halstead's intelligence
e	Halstead's effort
b	Halstead's delivered bugs
t	Halstead's time estimator
LOCode	Halstead's line count
LOComment	Halstead's count of lines of comments
LOBlank	Halstead's count of blank lines
LOCodeAnd Comment	Halstead's Count of lines of code that also contain a comment
Uniq_Op	Halstead's Unique operators
Uniq_Opnd	Halstead's Unique operands
Total_Op	Halstead's Total operators
Total_Opnd	Halstead's Total operands
BranchCount	Number of branches in the flow graph

Efficiency Metrics



$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall}$$

Single Classifiers

	Accuracy	Precision	Recall	F-score
Logistic regression	0.806	0.833	0.956	0.890
Decision Tree	0.721	0.845	0.810	0.827
SVM	0.825	0.826	0.997	0.904
Naïve Bayes	0.817	0.843	0.956	0.897
k-nn	0.778	0.838	0.906	0.871
Random Forest	0.810	0.841	0.950	0.892

Feature Selection

- ANOVA F-value (**SelectKBest**)
- Recursive Feature Elimination (**RFE**)
- Principal component analysis (**PCA**)

- Evaluation algorithm: RF
- Efficiency metric: F-score

Feature Selection: Results

Number of Features	SelectKBest	RFE	PCA
5	0.898	0.895	0.898
6	0.899	0.896	0.900
7	0.900	0.897	0.900
8	0.902	0.900	0.901
9	0.902	0.899	0.901
10	0.903	0.901	0.902
11	0.902	0.900	0.902
12	0.901	0.899	0.903
13	0.902	0.900	0.904
14	0.903	0.900	0.905
15	0.904	0.901	0.905
16	0.905	0.902	0.905

Data Balancing: Algorithms

Oversampling algorithms:

Data Balancing: Results

DATA BALANCING ALGORITHM	F-SCORE
KMeansSMOTE	0.904
TomekLinks	0.902
RandomOverSampler	0.889
SVMSMOTE	0.889
SMOTE	0.887
SMOTETomek	0.885
EditedNearestNeighbours	0.851
SMOTEENN	0.804
RandomUnderSampler	0.766
NearMiss	0.422

Combination of Feature Selection and Data Balancing

DATA BALANCING ALGORITHM	FEATURE SELECTION ALGORITHM	NUMBER OF FEATURES	F-SCORE
KMeansSMOTE	SelectKBest	8	0.902
KMeansSMOTE	SelectKBest	7	0.901
KMeansSMOTE	PCA	9	0.900
KMeansSMOTE	SelectKBest	9	0.900
KMeansSMOTE	PCA	7	0.900
KMeansSMOTE	PCA	8	0.900
KMeansSMOTE	RFE	9	0.900
KMeansSMOTE	RFE	8	0.898
TomekLinks	PCA	9	0.898
TomekLinks	SelectKBest	8	0.897

Hyperparameters Optimization: Logistic Regression

Hyperparameter			Data Balancing Algorithm	Feature Selection	Number of Features	F-score
solver	penalty	C				
liblinear	l1	0.1	TomekLinks	SelectKBest	7	0.906
saga	l1	0.1	TomekLinks	SelectKBest	8	0.906
newton-cg	l2	0.1	TomekLinks	SelectKBest	9	0.906
sag	l2	0.1	TomekLinks	RFE	7	0.904
newton-cg	l2	0.01	TomekLinks	PCA	9	0.904

Hyperparameters Optimization: Decision Tree

Hyperparameter		Data Balancing Algorithm	Feature Selection	Number of Features	F-score
criterion	max_depth				
entropy	6	TomekLinks	SelectKBest	7	0.905
gini	4	KMeansSMOTE	SelectKBest	8	0.906
entropy	6	TomekLinks	SelectKBest	8	0.904
entropy	6	TomekLinks	SelectKBest	9	0.904
entropy	2	KMeansSMOTE	RFE	9	0.903

Hyperparameters Optimization: SVM

Hyperparameter		Data Balancing Algorithm	Feature Selection	Number of Features	F-score
kernel	C				
rbf	50	TomekLinks	SelectKBest	7	0.907
rbf	50	KMeansSMOTE	SelectKBest	9	0.906
rbf	1.0	TomekLinks	SelectKBest	9	0.905
poly	10	KMeansSMOTE	RFE	7	0.905
rbf	50	TomekLinks	RFE	7	0.905

Hyperparameters Optimization: Naïve Bayes

HYPERPARAMETER (VAR_SMOOTHING)	DATA BALANCING ALGORITHM	FEATURE SELECTION	NUMBER OF FEATURES	F-SCORE
1.0	TomekLinks	SelectKBest	7	0.902
1.0	TomekLinks	SelectKBest	8	0.903
1.0	TomekLinks	RFE	7	0.903
0.023	KMeansSMOTE	PCA	7	0.905
1.0	TomekLinks	PCA	9	0.905

Hyperparameters Optimization: k-nn

Hyperparameter			Data Balancing Algorithm	Feature Selection	Number of Features	F-score
weights	metric	n_neighbors				
uniform	euclidean	19	KMeansSMOTE	SelectKBest	7	0.904
uniform	euclidean	19	KMeansSMOTE	SelectKBest	8	0.905
distance	euclidean	17	TomekLinks	SelectKBest	9	0.903
uniform	euclidean	19	KMeansSMOTE	RFE	8	0.904
distance	euclidean	19	TomekLinks	PCA	9	0.902

Hyperparameters Optimization: RF

Hyperparameter				Data Balancing Algorithm	Feature Selection	Number of Features	F-score
max_depth	max_features	min_samples_leaf	n_estimators				
37	sqrt	2	100	TomekLinks	SelectKBest	8	0.905
10	sqrt	2	500	KMeansSMOTE	SelectKBest	9	0.907
-	sqrt	2	500	TomekLinks	RFE	8	0.905
10	sqrt	2	500	KMeansSMOTE	PCA	7	0.906
10	sqrt	1	1000	KMeansSMOTE	RFE	8	0.907

Ensembling: Voting

DATA BALANCING ALGORITHM	FEATURE SELECTION	NUMBER OF FEATURES	ACCURACY	F-SCORE
KMeansSMOTE	SelectKBest	8	0.830	0.905
TomekLinks	SelectKBest	8	0.833	0.905
KMeansSMOTE	Variable for each basic classifier	-	0.829	0.905
TomekLinks	Variable for each basic classifier	-	0.832	0.905
Variable for each basic classifier	Variable for each basic classifier	-	0.833	0.906

Ensembling: Stacking (Logistic Regression)

DATA BALANCING ALGORITHM	FEATURE SELECTION	NUMBER OF FEATURES	PASSTHROUGH	ACCURACY	F-SCORE
KMeansSMOTE	SelectKBest	8	False	0.828	0.905
KMeansSMOTE	SelectKBest	8	True	0.827	0.905
TomekLinks	SelectKBest	8	False	0.834	0.905
TomekLinks	SelectKBest	8	True	0.835	0.906
KMeansSMOTE	Variable for each basic classifier	-	False	0.829	0.906
KMeansSMOTE	Variable for each basic classifier	-	True	0.828	0.905
TomekLinks	Variable for each basic classifier	-	False	0.837	0.907
TomekLinks	Variable for each basic classifier	-	True	0.838	0.908
Variable for each basic classifier	Variable for each basic classifier	-	False	0.835	0.907
Variable for each basic classifier	Variable for each basic classifier	-	True	0.837	0.908

Ensembling: Stacking

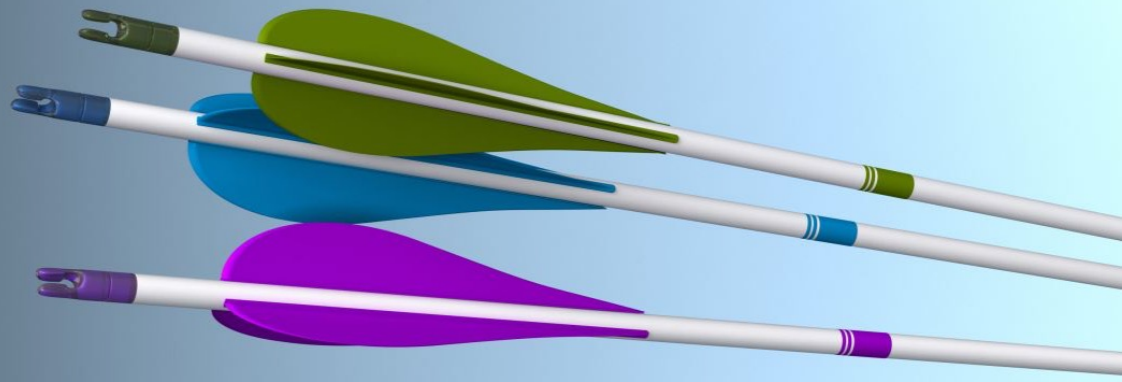
```
svc = Pipeline([
    ('feature_selection', RFE(LogisticRegression(), n_features_to_select=7)),
    ('sampling', KMeansSMOTE()),
    ('clf', SVC(C=10, kernel='poly'))])

nb = Pipeline([
    ('feature_selection', SelectKBest(k=8)),
    ('sampling', TomekLinks()),
    ('clf', GaussianNB(var_smoothing=1.0))])

knn = Pipeline([
    ('feature_selection', SelectKBest(k=8)),
    ('sampling', KMeansSMOTE()),
    ('clf', KNeighborsClassifier(n_neighbors=19, weights='uniform', metric='euclidean'))])

rf = Pipeline([
    ('feature_selection', SelectKBest(k=8)),
    ('sampling', TomekLinks()),
    ('clf', RandomForestClassifier(n_estimators=100, min_samples_leaf=2, max_features='sqrt', max_depth=37))])


estimators = [('logistic_regression', logistic_regression),
              ('cart', cart),
              ('svc', svc),
              ('knn', knn),
              ('nb', nb),
              ('rf', rf)]

final_estimator = RandomForestClassifier()

clf = StackingClassifier(estimators=estimators, final_estimator=final_estimator, passthrough=True)
```

Ensembling: Stacking (RF)

DATA BALANCING ALGORITHM	FEATURE SELECTION	NUMBER OF FEATURES	PASSTHROUGH	ACCURACY	F-SCORE
KMeansSMOTE	SelectKBest	8	False	0.830	0.906
KMeansSMOTE	SelectKBest	8	True	0.828	0.906
TomekLinks	SelectKBest	8	False	0.829	0.901
TomekLinks	SelectKBest	8	True	0.834	0.905
KMeansSMOTE	Variable for each basic classifier	-	False	0.827	0.905
KMeansSMOTE	Variable for each basic classifier	-	True	0.829	0.906
TomekLinks	Variable for each basic classifier	-	False	0.834	0.905
TomekLinks	Variable for each basic classifier	-	True	0.838	0.907
Variable for each basic classifier	Variable for each basic classifier	-	False	0.839	0.909
Variable for each basic classifier	Variable for each basic classifier	-	True	0.838	0.909

Conclusions

- Six single classifiers (default hyperparameters, no data balancing, no feature selection):
  - average: accuracy = 0.793, F-score = 0.880
  - best: accuracy = 0.825, F-score = 0.904 (SVM), accuracy = 0.812, F-score = 0.897 (Naïve Bayes)
- Feature selection: F-score up to 0.900±0.020 for 7 to 9 features
- Data balancing:
  F-score up to 0.904 (KMeansSMOTE) and 0.902 (Tomek Links)
- Hyperparameters optimization (+ data balancing + feature selection):
  F-score up to 0.905 (RF) and 0.907 (SVM)
- Ensembling (Stacking with RF): Accuracy up to 0.839, F-score up to 0.909



Thank you for attention!