SOFTWARE DEFECT PREDICTION

NIDA ZAKI (Z1938561)
YUE MA (Z1934458)

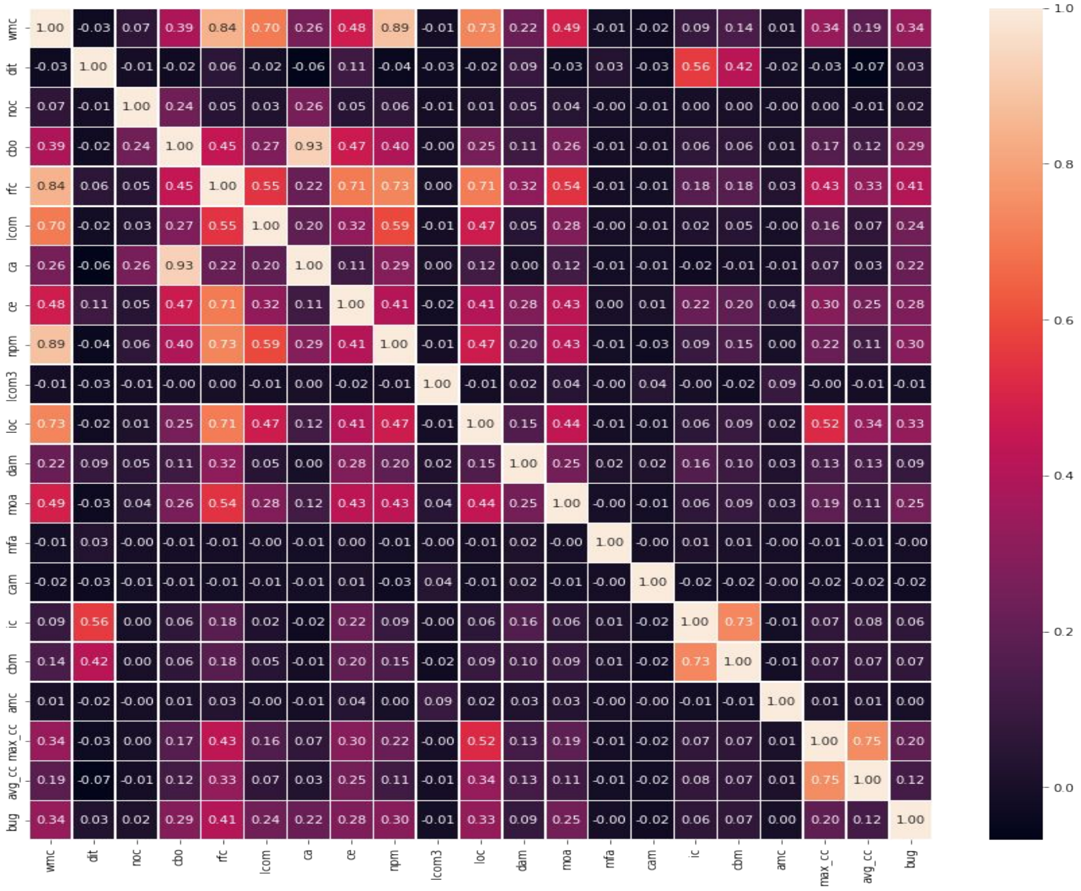
MOTIVATION

- ➢ Software Defect Prediction which predicts defective code regions, can help developers find bugs and prioritize their testing efforts.
- ➢ We utilize classifier models and artificial neural networks as a means to detect and predict defects in softwares.
- ➢ These models work by learning patterns between attributes of the software and a corresponding binary label that indicates whether or not a defect exists.
- ➢ These models can also help to better understand the underlying software features and how they affect the defect rate.
- ➢ We perform feature importance measures to see which features are more important in the models prediction performance.
- ➢ All this information can be extremely useful, and may suggest an area of focus for software design companies.
- ➢ Previous work only tested with limited models, most of researchers chose ensemble learning [1] [2] [3]. We will contribute to a more comprehensive comparisons which include most of the popular classification machine learning methods, as well as artificial neural networks.

DATASET (ATTRIBUTES)

name: a brief name
version: version of the software
name: a detailed version name for the software
wmc: weighted methods per class
dit: depth of inheritance tree
noc: number of children
cbo: coupling between objects
rfc: response for class
lcom: lack of cohesion of methods
ca: afferent couplings
ce: efferent couplings
npm: number of public methods
lcom3: lack of cohesion in methods
loc: lines of code
dam: data access metric
moa: measure of aggregation
mfa: multi-factor authentication
cam: cohesion among methods
ic: continuous integration
cbm: coupling between methods
amc: average method complexity
max_cc: maximum cyclomatic complexity
avg_cc: average cyclomatic complexity
bug: number of bugs

ATTRIBUTE CORRELATION



Correlation refers to the strength of the relationship between two attributes.

Here, we can see that the correlation between the attributes is very low (indicated by the dark color in the heat map), meaning that the attributes/ features are hardly related.

FEATURE SELECTION

Features with low correlation are less linearly dependent and hence they contribute individually to the prediction model.

Hybrid Feature Selection method is used to automate the process of finding the minimal and optimal set of features which reduces the computational and time complexity of the model.

Correlation Based Feature Selection: From the correlation matrix and depending on the threshold value, it is decided if certain features are to be removed from the dataset. The features that have correlation greater than 0.95 with respect to other features are to be removed from the dataset.

```
threshold = 0.95
correlation(df.iloc[:,:-1],threshold)
```

```
set()
```

```
threshold = 0.8
correlation(df.iloc[:,:-1],threshold)
```

```
{'ca', 'npm', 'rfc'}
```

As shown here, we got an empty set, so we did not remove any feature.

But if the threshold value is 0.8, then features that have correlation greater than 0.8 with respect to other features would be removed, which would be ca, npm and rfc.

RESEARCH QUESTIONS - SUPERVISED LEARNING

RESEARCH QUESTIONS: Which classification method has the best performance in predicting the defect (bug) in a software?

DEPENDENT VARIABLE: “bug”

“bug”>=1 → has defects (1)

“bug”=0 → no defect (0)

INDEPENDENT VARIABLES: wmc, dit, noc, cbo, rfc, lcom, ca, ce, npm, lcom3, loc, dam, moa, mfa, cam, ic, cbm, amc, max_cc, avg_cc

METHODS / MODELS

1. Decision Tree
2. Random Forest
3. K Nearest Neighbors
4. Support Vector Machines (SVM)
5. Ensemble

    a. Bagging

    b. Boosting

        -Adaboost

        -Gradient Boosting

        - XGBoost

6. Naive Bayes Classifier

    a.Gaussian Naive Bayes

    b.Bernoulli Naive Bayes

7. Neural Networks

    a.Sequential Neural Networks

    b.Multi Layer Perceptron

8. Cross Validation

9. Undersampling

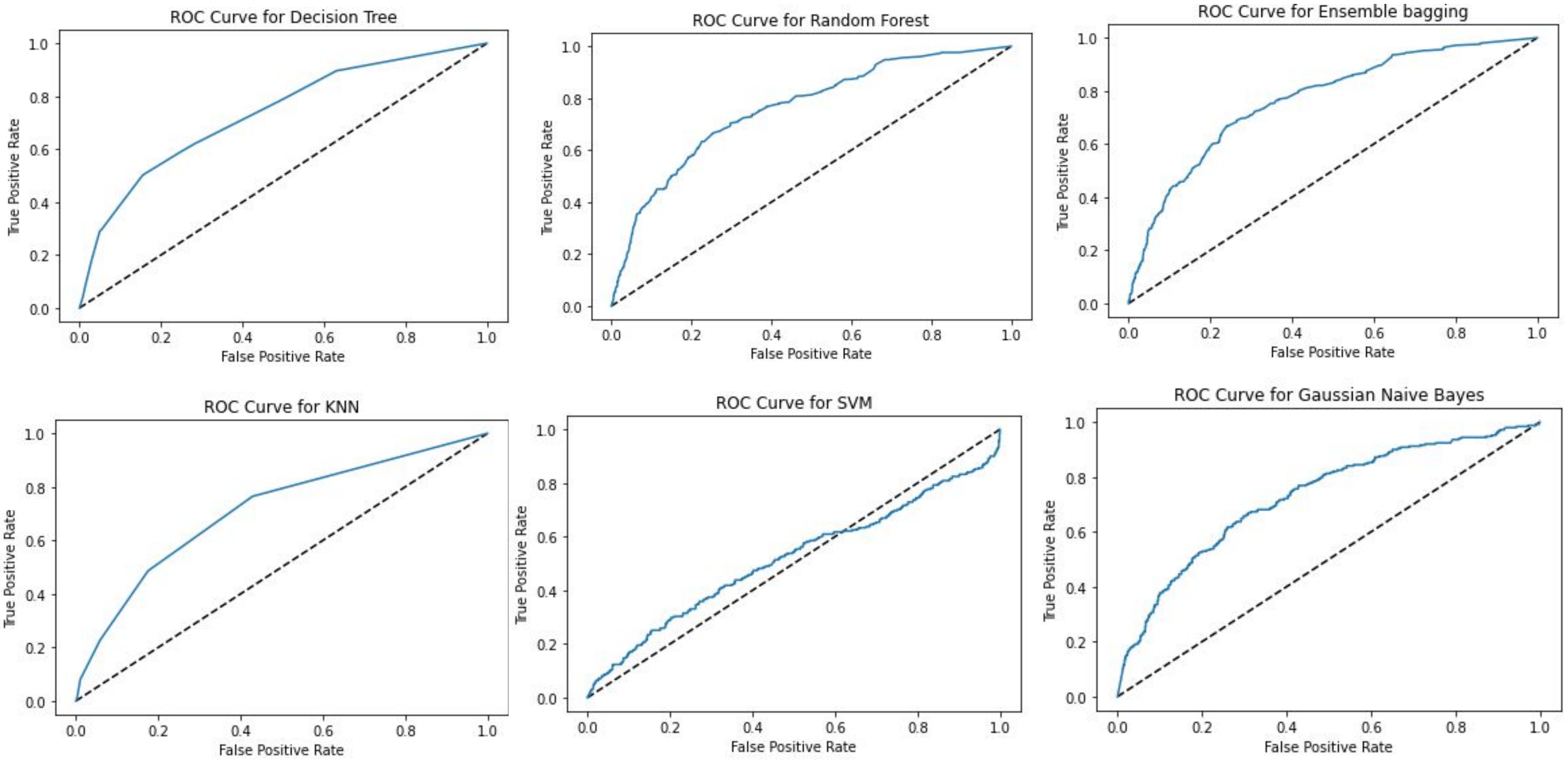
*5,6,8,9 are for dealing with imbalanced dataset

RESULTS - ACCURACY
(Best: Ensemble Boosting- Gradient Boosting)

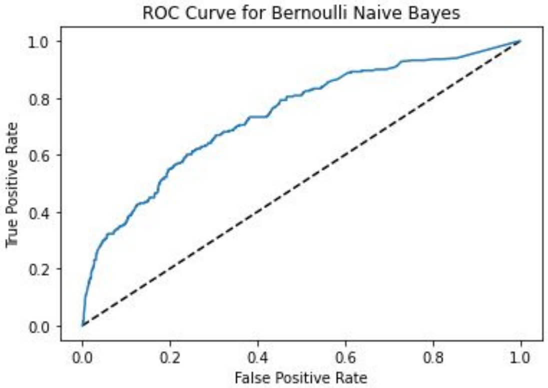
Decision Tree	0.823
Random Forest	0.827
Ensemble - Bagging	0.828
Boosting - Adaboost	0.81
Boosting - Gradient Boosting	0.84
Boosting - XGBoost	0.83
K Nearest Neighbors	0.813

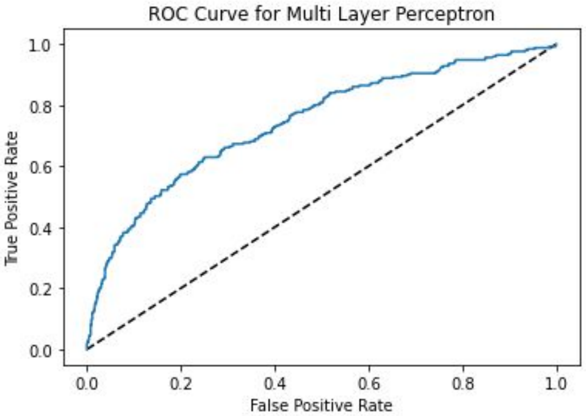
SVM	0.73
Naive Bayes- Gaussian Bayes	0.374
Naive Bayes- Bernoulli Naive Bayes	0.737
Neural Networks- Sequential NN	0.83
Neural Networks- Multi Layer Perceptron	0.83

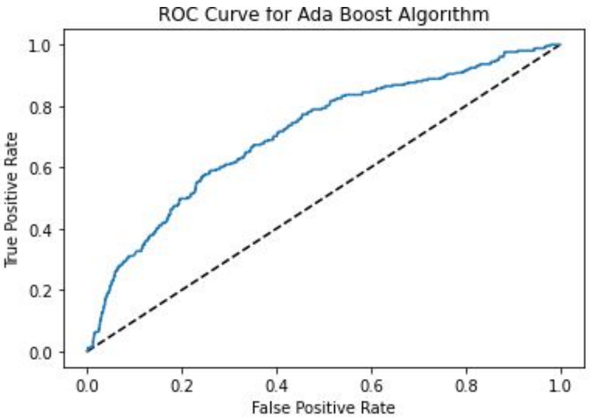
RESULTS- ROC CURVE (PART - 1)

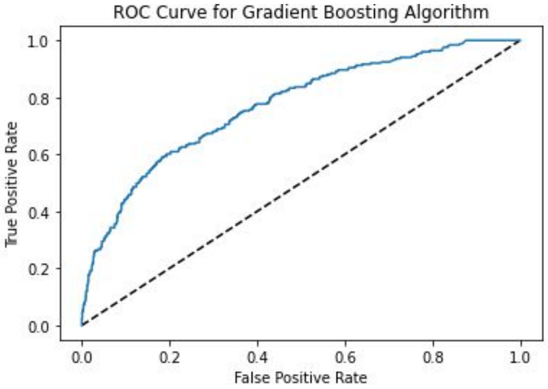


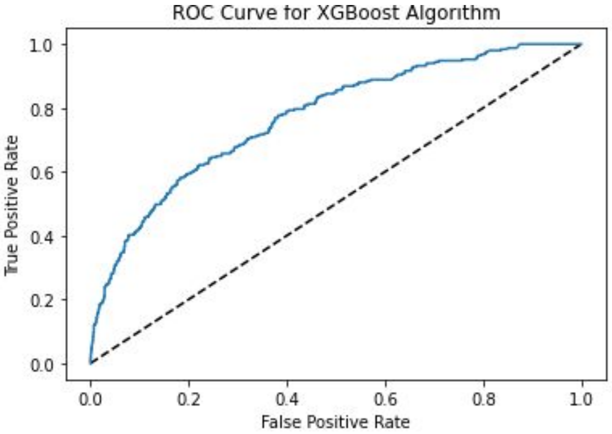
RESULTS- ROC CURVE (PART - 2)











In conclusion, SVM has the worst performance for the ROC test result.

RESULTS - CLASSIFICATION REPORT (PART - 1)

Decision Tree Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.83	0.99	0.90	1141
1	0.56	0.08	0.13	251
accuracy			0.82	1392
macro avg	0.69	0.53	0.52	1392
weighted avg	0.78	0.82	0.76	1392

Random Forest Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.86	0.95	0.90	1141
1	0.54	0.28	0.37	251
accuracy			0.83	1392
macro avg	0.70	0.61	0.63	1392
weighted avg	0.80	0.83	0.80	1392

Ensemble Bagging Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.86	0.95	0.90	1141
1	0.54	0.28	0.37	251
accuracy			0.83	1392
macro avg	0.70	0.62	0.64	1392
weighted avg	0.80	0.83	0.80	1392

K Nearest Neighbor Algorithm
Classification report :

	precision	recall	f1-score	support
0	0.85	0.94	0.89	1141
1	0.46	0.23	0.30	251
accuracy			0.81	1392
macro avg	0.65	0.58	0.60	1392
weighted avg	0.78	0.81	0.79	1392

RESULTS - CLASSIFICATION REPORT (PART - 2)

Support Vector Machine (SVM) Algorithm
Classification report :

	precision	recall	f1-score	support
0	0.84	0.84	0.84	1141
1	0.26	0.25	0.25	251
accuracy			0.73	1392
macro avg	0.55	0.54	0.55	1392
weighted avg	0.73	0.73	0.73	1392

Gaussian Naive Bayes Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.94	0.25	0.40	1141
1	0.21	0.92	0.35	251
accuracy			0.37	1392
macro avg	0.57	0.59	0.37	1392
weighted avg	0.81	0.37	0.39	1392

Bernoulli Naive Bayes Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.89	0.77	0.83	1141
1	0.36	0.57	0.44	251
accuracy			0.74	1392
macro avg	0.62	0.67	0.63	1392
weighted avg	0.80	0.74	0.76	1392

Multi Layer Perceptron Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.85	0.97	0.90	1141
1	0.56	0.20	0.29	251
accuracy			0.83	1392
macro avg	0.70	0.58	0.60	1392
weighted avg	0.79	0.83	0.79	1392

RESULTS - CLASSIFICATION REPORT (PART - 3)

Ada Boost Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.85	0.92	0.88	1141
1	0.43	0.28	0.34	251
accuracy			0.80	1392
macro avg	0.64	0.60	0.61	1392
weighted avg	0.78	0.80	0.79	1392

Gradient Boosting Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.85	0.98	0.91	1141
1	0.70	0.20	0.31	251
accuracy			0.84	1392
macro avg	0.78	0.59	0.61	1392
weighted avg	0.82	0.84	0.80	1392

XG Boost Algorithm
Classification Report :

	precision	recall	f1-score	support
0	0.85	0.98	0.91	1141
1	0.64	0.19	0.29	251
accuracy			0.83	1392
macro avg	0.74	0.58	0.60	1392
weighted avg	0.81	0.83	0.79	1392

Result Changes After Applying Undersampling Method

SUMMARY OF CLASSIFICATION REPORTS OF CLASSIFIERS

	Label	DT	RF	Bagging	AdaBoost	GB	XGB	KNN	SVM	GaussianNB	BernoulliNB	MLP
Precision	0	0.83	0.86	0.86	0.85	0.85	0.85	0.85	0.84	0.94	0.89	0.85
	1	0.56	0.54	0.54	0.43	0.70	0.64	0.46	0.26	0.21	0.36	0.56
Recall	0	0.99	0.95	0.95	0.92	0.98	0.98	0.94	0.84	0.25	0.77	0.97
	1	0.08	0.28	0.28	0.28	0.20	0.19	0.23	0.25	0.92	0.57	0.20
F1-score	0	0.90	0.90	0.90	0.88	0.91	0.91	0.89	0.84	0.40	0.83	0.90
	1	0.13	0.37	0.37	0.34	0.31	0.29	0.30	0.25	0.35	0.44	0.29

THE CLASSIFICATION REPORT AFTER APPLYING UNDER SAMPLING METHOD

	Label	DT	RF	Bagging	Adaboost	GB	XGBoost	KNN	SVM	GaussianNB	BernoulliNB	MLP
Precision	0	0.61	0.67	0.68	0.69	0.67	0.67	0.63	0.55	0.57	0.63	0.67
	1	0.73	0.69	0.69	0.68	0.71	0.71	0.66	0.55	0.73	0.70	0.70
Recall	0	0.83	0.69	0.70	0.67	0.75	0.75	0.70	0.53	0.87	0.75	0.73
	1	0.46	0.67	0.67	0.71	0.63	0.63	0.60	0.56	0.34	0.56	0.63
F1-score	0	0.70	0.68	0.69	0.68	0.70	0.71	0.66	0.54	0.69	0.69	0.70
	1	0.57	0.68	0.68	0.69	0.67	0.67	0.63	0.55	0.46	0.62	0.67

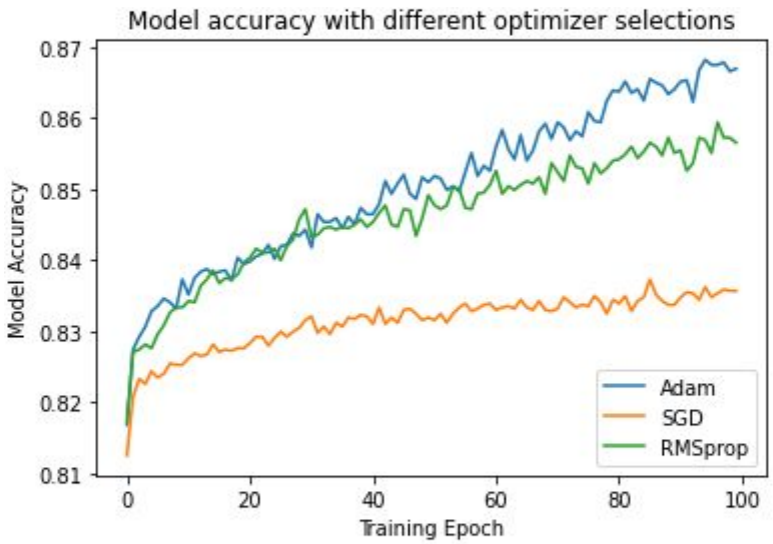
MAE Score After Applying Cross Validation

MAE SCORE OF CROSS VALIDATION ON MULTIPLE CLASSIFIERS

	DT	RF	Bagging	Adaboost	GB	XGB	KNN	SVM	GaussianNB	BernoulliNB
Fold-1	0.175	0.181	0.180	0.187	0.173	0.171	0.186	0.267	0.810	0.244
Fold-2	0.175	0.177	0.176	0.189	0.180	0.172	0.183	0.265	0.590	0.250
Fold-3	0.185	0.184	0.187	0.198	0.186	0.194	0.197	0.263	0.633	0.246
Fold-4	0.182	0.185	0.189	0.209	0.182	0.189	0.197	0.261	0.705	0.279
Fold-5	0.174	0.176	0.179	0.195	0.173	0.167	0.166	0.261	0.266	0.253
Average	0.178	0.181	0.182	0.196	0.179	0.179	0.186	0.264	0.601	0.254

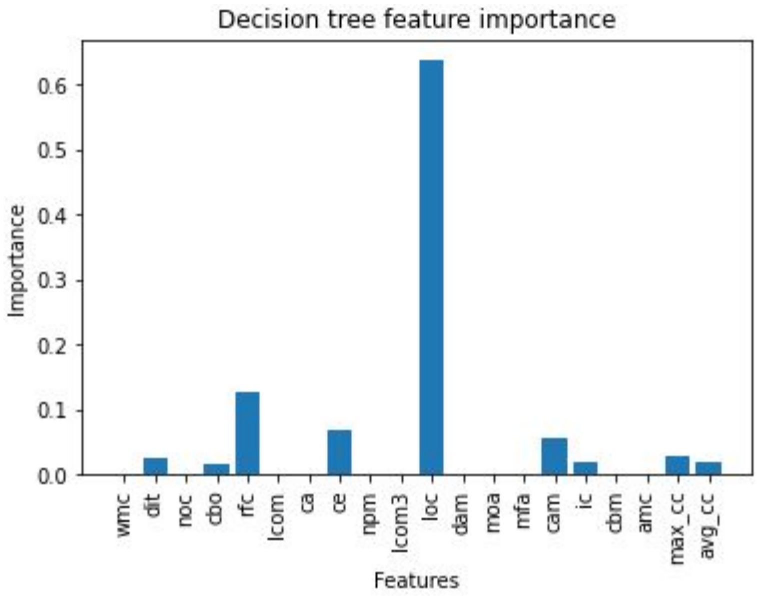
Artificial Neural Network or Sequential Neural Network

Input shape	Hidden layer 1	Hidden layer 2	Output layer
(20,)	size = 32	size=32	Size= 1

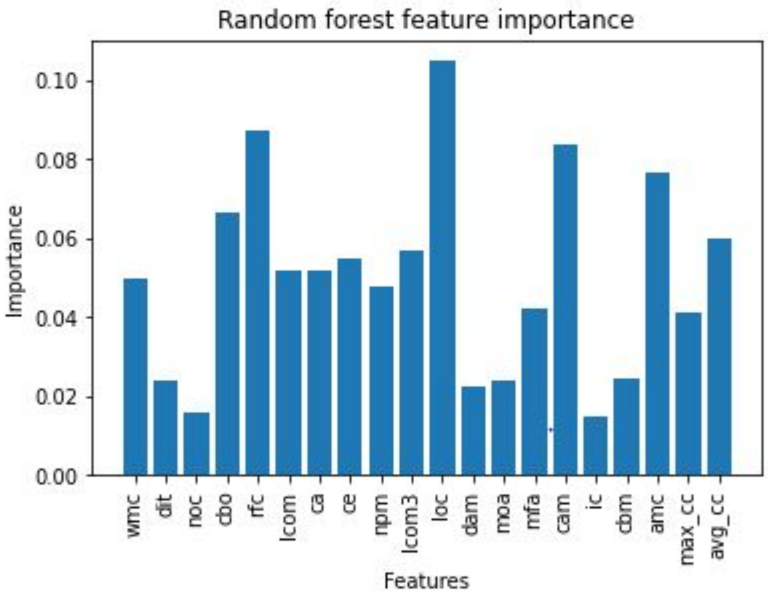


Testing result (Accuracy)	
Model with Adam	0.8305
Model with SGD	0.8297
Model with RMSprop	0.8254

FEATURE IMPORTANCE (PART - 1)

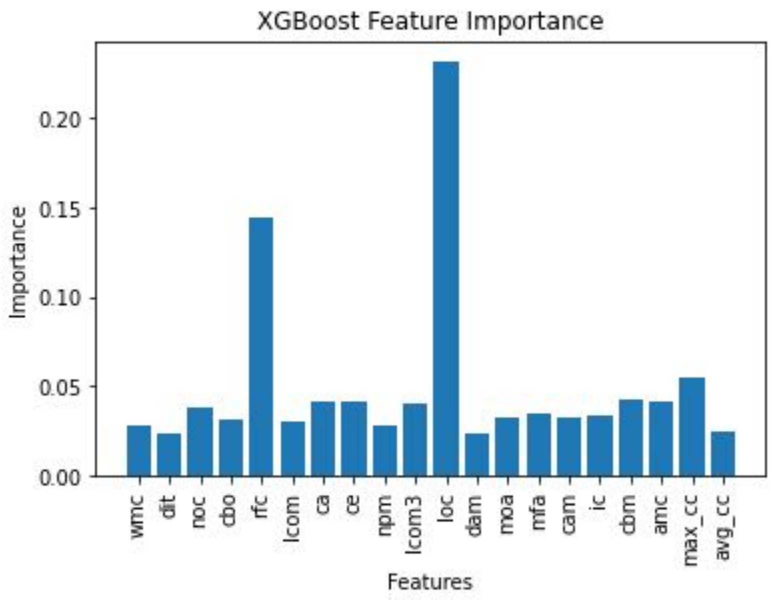


Decision Tree Feature Importance



Random Forest Feature Importance

FEATURE IMPORTANCE (PART - 2)



XGBoost Feature Importance

Top 3 important features:

loc: lines of code
rfc: response for class
cam: methods of class

REFERENCES

[1] A., Mabayoje, Abdullateef Balogun, Amos Bajeh, and Badamasi Musa. "SOFTWARE DEFECT PREDICTION: EFFECT OF FEATURE SELECTION AND ENSEMBLE METHODS" 3 (September 10, 2018): 518–22.

[2] Sun, Zhongbin, Qinbao Song, and Xiaoyan Zhu. "Using Coding-Based Ensemble Learning to Improve Software Defect Prediction." *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42, no. 6 (November 2012): 1806–17. https://doi.org/10.1109/TSMCC.2012.2226152.

[3] Balogun, Abdullateef, Amos Bajeh, Victor Orie, and Ayisat Yusuf-Asaju. "Software Defect Prediction Using Ensemble Learning: An ANP Based Evaluation Method." *FUOYE Journal of Engineering and Technology* 3 (September 1, 2018). https://doi.org/10.46792/fuoyejet.v3i2.200.