ASSIGNMENT 4 MOM

Asam Mahmood

Florida Atlantic University

PROF. TAGHI M KOSHGOFTAR

Course CAP 6673 Data Mining and Machine Learning

Introduction

In this experiment we will apply 2 linear regression models to our fit data set, that will predict number of faults for the dependent variable faults. These are our candidate independent variables which are eight in total variables: NUMUORS, NUMUANDS, TOTOTORS, VG, NLOGIC, LOC, ELOC.

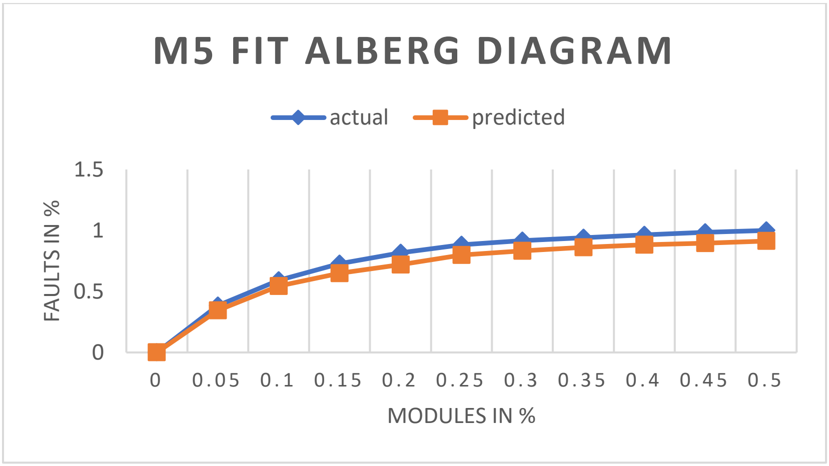
Input

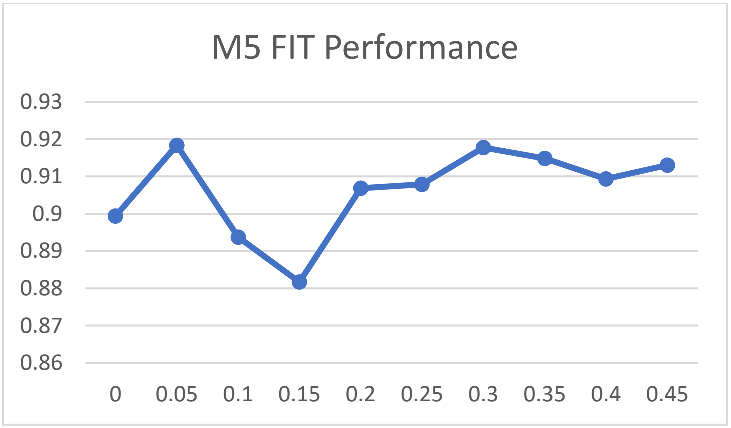
Our data FIT and Test data sets were given to us as a requirement to use. There are 9 attributes in and 188 instances in FIT data set. The Test data set has 94 instances and 9 attributes. Class attribute in each set has a requirement when attribute is less than 2 faults are considered nfp, and modules with 2 or more faults are considered fp .It was used in our previous assignment 1a where we engineered the input. I used Microsoft excel as a tool to run the algorithm to apply to our fit data set to calculate the predicted values and also the remaining values. I referenced docs for formulas and instructions to resolve. Below are instruction set that I followed for high level procedural steps for table input. After going through reference documents, it helps having high level instructions.

1. You will need to determine the **perfect ranking of modules** in the fit dataset, by ordering modules according to the ***actual*** FAULTS: [1]
2. You will then determine the perfect ranking of modules in the fit dataset, by ordering modules according to ***predicted faults. [1]***
3. While you apply a module-order model, modules are selected for reliability enhancement predicted order, beginning with the most fault prone. A cutoff point c is the percentile of the last module enhanced. [1]
   Calculate the sum of the actual number of faults $G(c)$, in modules above the cut off for R
   Calculate the sum of the actual number of faults $\hat{G}(c)$, in modules above the cut off for $\hat{R}$
4. Calculate the percentage of faults accounted for by each ranking,
   $G(c)/F_{tot}$ and $\hat{G}(c)/F_{tot}$. [1]
5. Calculate the measure of model performance that indicated how closely the faults accounted for by the model ranking match those of the perfect ranking. $\emptyset(c) = \hat{G}(c)/G(c)$ [1][2].

M5 FIT

c	$G(c)$	$\hat{G}(c)$	$\frac{G(c)}{F_{tot}}$	$\frac{\hat{G}(c)}{F_{tot}}$	$\varphi(c)$	Modules %
0.5	414	378	1	0.913043	0.913043	0.5
0.55	408	371	0.985507	0.896135	0.909314	0.45
0.6	399	365	0.963768	0.881643	0.914787	0.4
0.65	389	357	0.939614	0.862319	0.917738	0.35
0.7	380	345	0.917874	0.833333	0.907895	0.3
0.75	365	331	0.881643	0.799517	0.906849	0.25
0.8	338	298	0.816425	0.719807	0.881657	0.2
0.85	301	269	0.727053	0.649758	0.893688	0.15
0.9	245	225	0.591787	0.543478	0.918367	0.1
0.95	159	143	0.384058	0.345411	0.899371	0.05

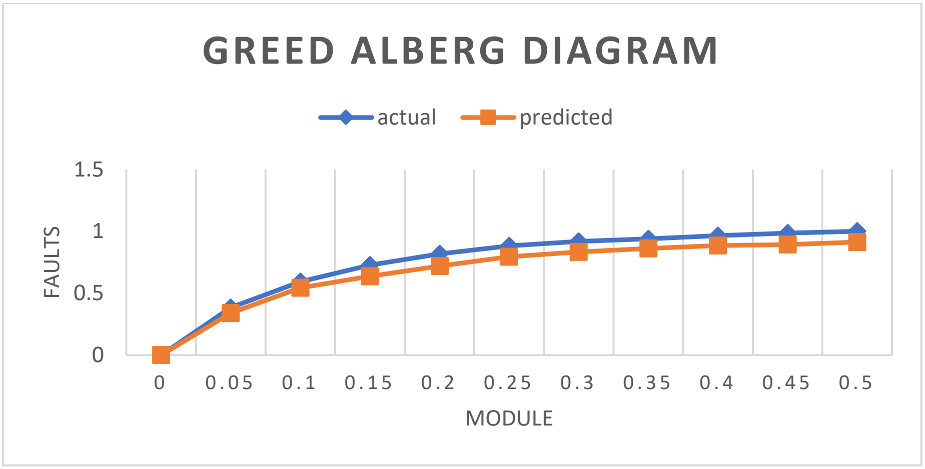


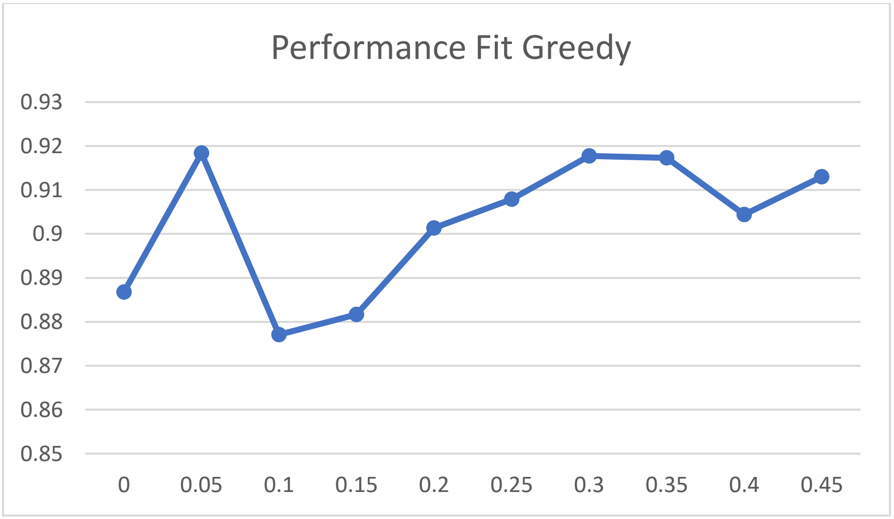


The model performance $\emptyset(c)$ is about the same for the range of c of interest; we consider the model very robust. [1]

Greedy FIT

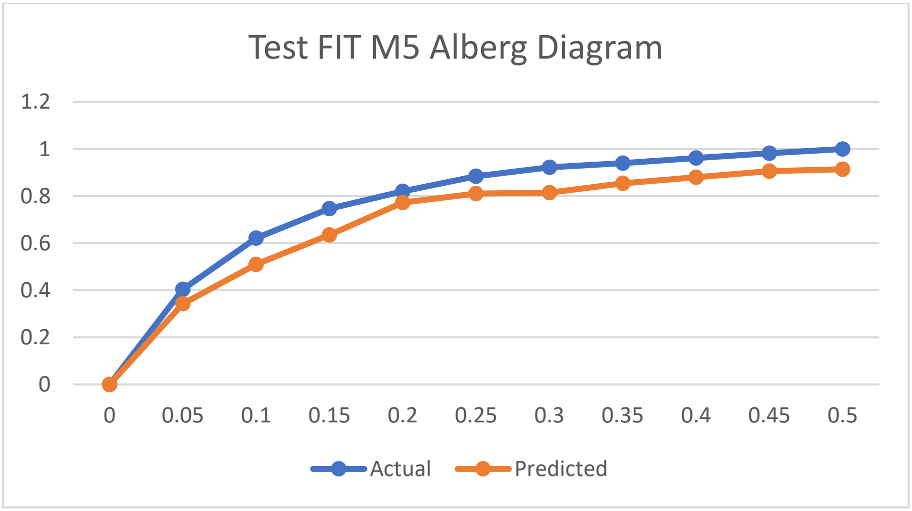
	$G(c)$	$\hat{G}(c)$	$\frac{G(c)}{F_{tot}}$	$\frac{\hat{G}(c)}{F_{tot}}$	$\varphi(c)$	**Modules %**
0.5	414	378	1	0.913043	0.913043	0.5
0.55	408	369	0.985507	0.891304	0.904412	0.45
0.6	399	366	0.963768	0.884058	0.917293	0.4
0.65	389	357	0.939614	0.862319	0.917738	0.35
0.7	380	345	0.917874	0.833333	0.907895	0.3
0.75	365	329	0.881643	0.794686	0.90137	0.25
0.8	338	298	0.816425	0.719807	0.881657	0.2
0.85	301	264	0.727053	0.637681	0.877076	0.15
0.9	245	225	0.591787	0.543478	0.918367	0.1
0.95	159	141	0.384058	0.34058	0.886792	0.05

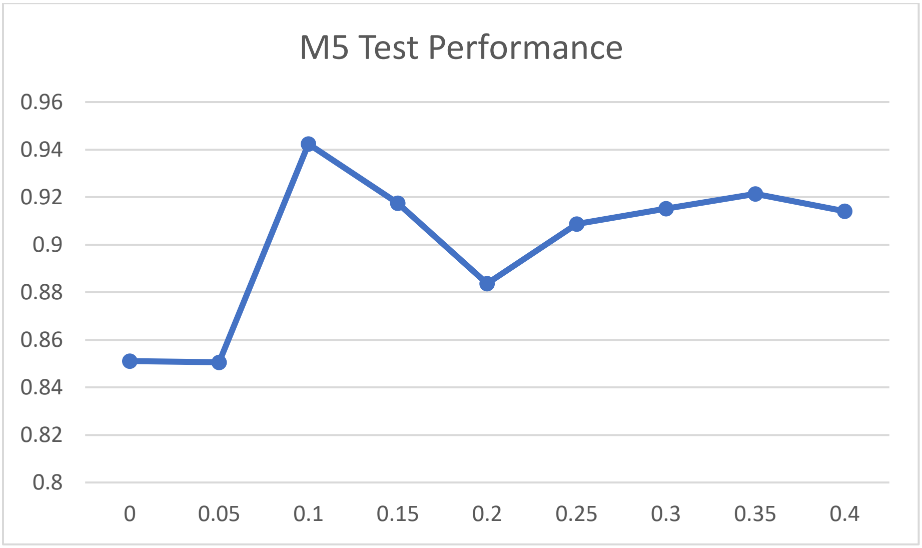




M5 Fit Test

	$G(c)$	$\hat{G}(c)$	$\frac{G(c)}{F_{tot}}$	$\frac{\hat{G}(c)}{F_{tot}}$	$\varphi(c)$	**Modules %**
0.5	233	213	1	0.914163	0.914163	0.5
0.55	229	211	0.982833	0.905579	0.921397	0.45
0.6	224	205	0.961373	0.879828	0.915179	0.4
0.65	219	199	0.939914	0.854077	0.908676	0.35
0.7	215	190	0.922747	0.815451	0.883721	0.3
0.75	206	189	0.88412	0.811159	0.917476	0.25
0.8	191	180	0.819742	0.772532	0.942408	0.2
0.85	174	148	0.746781	0.635193	0.850575	0.15
0.9	145	119	0.622318	0.51073	0.82069	0.1
0.95	94	80	0.403433	0.343348	0.851064	0.05

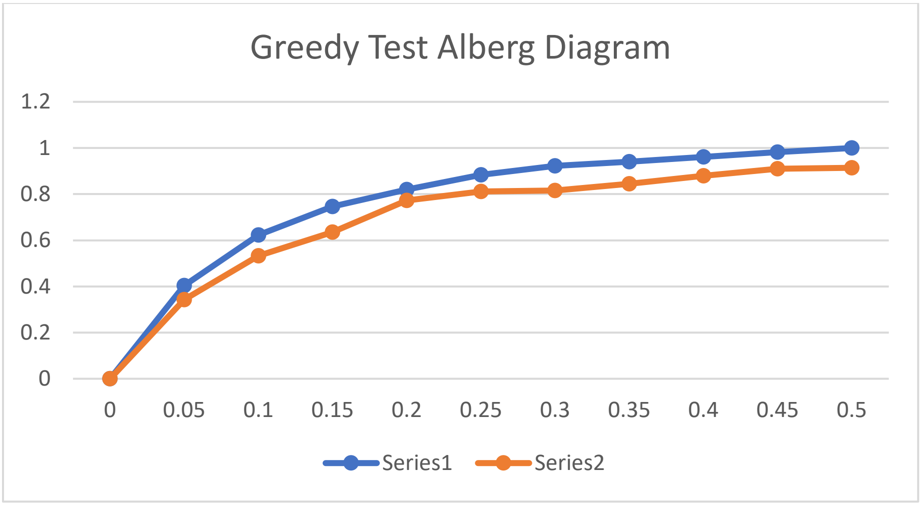


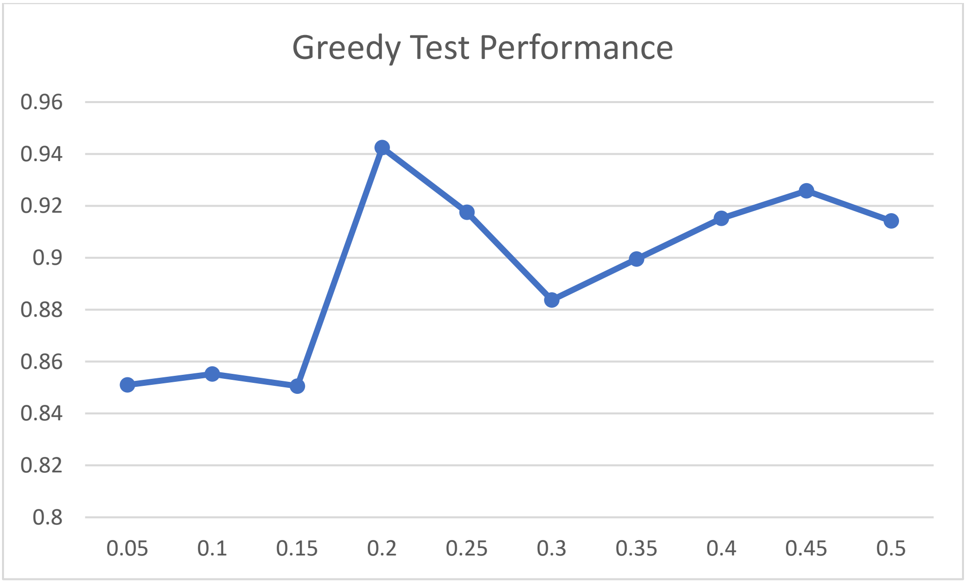


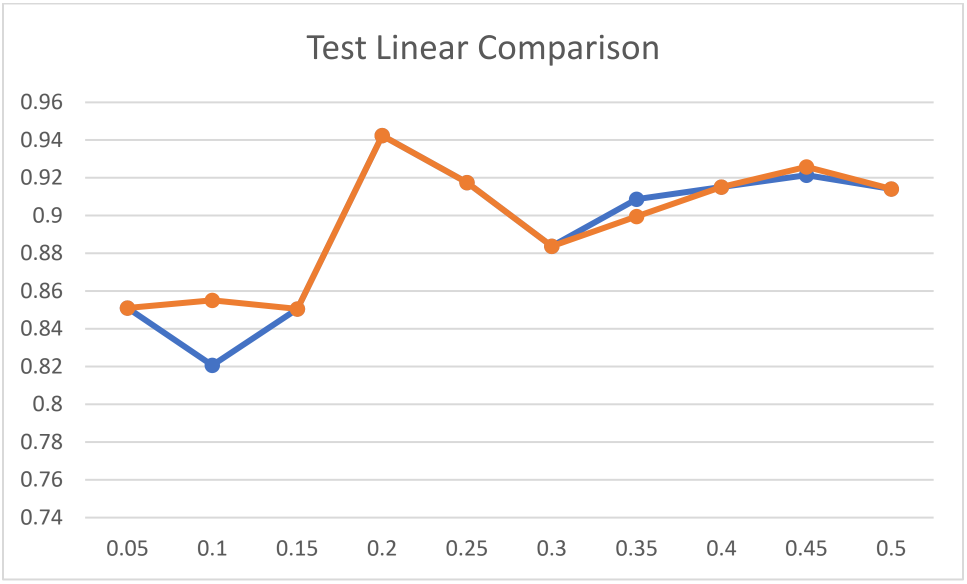
The model performance $\emptyset(c)$ is about the same for the range of c of interest; we consider the model very robust.[1]

Greedy Test

	$G(c)$	$\hat{G}(c)$	$\frac{G(c)}{F_{tot}}$	$\frac{\hat{G}(c)}{F_{tot}}$	$\varphi(c)$	**Modules %**
0.5	233	213	1	0.914163	0.914163	0.5
0.55	229	212	0.982833	0.909871	0.925764	0.45
0.6	224	205	0.961373	0.879828	0.915179	0.4
0.65	219	197	0.939914	0.845494	0.899543	0.35
0.7	215	190	0.922747	0.815451	0.883721	0.3
0.75	206	189	0.88412	0.811159	0.917476	0.25
0.8	191	180	0.819742	0.772532	0.942408	0.2
0.85	174	148	0.746781	0.635193	0.850575	0.15
0.9	145	124	0.622318	0.532189	0.855172	0.1
0.95	94	80	0.403433	0.343348	0.851064	0.05







Evaluation:

The median faults were substantially below the mean. The explanation for this can be found outlined in both papers were the same as here[1]. This is the reason in my MOM, I did not analyze the model for modules below the median $c < 50$ The results for each model calculations done in excel are above. Also, each page represents the MOM based on the specific data and linear regression model specified in the title.

Metrics

On each page there is a table with columns c our percentile, module % which specifies the index where each cut off would occur, $\boldsymbol{G(c)}$ is the number of actual faults occurring above the percentile cutoff based on the perfect ranking [1][2]., $\boldsymbol{\hat{G}(c)}$ is the number of actual faults occurring above the percentile cutoff based on the ordering of modules by predicted fault using the specified linear regression model [1][2]., $\boldsymbol{\varphi(c)}$ is the performance metric and is equal to the ratio of $\boldsymbol{\hat{G}(c)}$ **to** $\boldsymbol{G(c)}$[1], and remaining two columns are the ratio of $\boldsymbol{G(c)}$ and $\boldsymbol{\hat{G}(c)}$ to the number of total faults in the data set $\boldsymbol{F_{tot}}$.

Graphs

Second are the graphs below the tables the first graph is Alberg Diagram where again the x-axis is the % of modules at each c and the two lines are the predicted number of faults and actual number of faults (that is, it is the plot of the two columns of the table, $\boldsymbol{G(c)/F_{tot}}$ **and** $\boldsymbol{\hat{G}(c)/F_{tot}}$) [1][2]. I was able to find an example in the references from page on professor site to be able to create from my metrics calculated. The second graph on each page is the performance of the model ordering model. That graph is just the plot of $\boldsymbol{\varphi(c)}$ against the % of modules on the x-axis[1]. The performance graphs demonstrate to us how each model came to a perfect ordering of the modules for each data set[1][2].

FIT Comparison

When I compare the Linear Regression Model with M5 Method and Greedy Method of Attribute Selection running on **fit** dataset, I can say safely based on the performance graphs and tables above that the linear regression using M5 performed equal or better for all values of c except for $c = 0.6$ than greedy. In this instances M5 was still quite close at these values of c and the lower thresholds of c ($c = 0.6$) are less likely in practice as companies would have to expend significantly more resources to look at this many modules and according to Pareto's Law[1], one expects 20 % of the modules will contain 80 % of the faults. On the fit linear regression models using M5 for attribute selection is preferred to Greedy method for the module-ordering model since it performance better for almost every value except of $c\ at\ .6$.

Test Comparison

When I compare the Linear Regression Model with M5 Method and Greedy Method of Attribute Selection running on **test** dataset, I can safely say with tables as evidence that the $\boldsymbol{G(c)/F_{tot}}$ are almost the same. Both methods could make the predicted curve close to the perfect curve and the rank order modal is accurate. the performance graphs of Greedy and M5 were about the same for all values of c with the exception of c = 0.65. The model performance $\boldsymbol{\emptyset(c)}$ is about the same for the range of c. we consider the ***two model very robust***. [1][2]. When consider the most fault-prone of the modules that the model recommended for reliability enhanced (c = 0.95)[1][2].The small variation **in** $\boldsymbol{\emptyset(c)}$ indicates that the module order model was robust for different cutoff percentiles with good accuracy. [1][2].

Finally, the Alberg Diagrams, support the theories above based on the performance charts. The Alberg Diagram plot the percentage of faults in the percentage of modules (1-c) for both the actual (perfect ordering) and the predict (ordering based on model $R\hat{}$) [1][2].. If these two graphs are similar or look very close for a value of c that means the model did well on that value of *c[1].* This is reflected in the performance graphs having a high performance (y-value) for that same c (x-value). Looking at the Alberg Diagram's for the fit data set we see that the curves are relatively closer for all values of c for the M5 graph when compared to the Greedy graph. And on the test data set, we have the opposite in that the curves of the model using Greedy are closer throughout[1][2]. confirming what was describe above analysis of the performance graphs. I showed this on the last table linear test comparisons as a visual, I show this where both tests are on top of each other on the same x – scale then the performance are a given value the c we are interest in and it is reflected immediately in the graph.

References:

[1] Taghi M., K & Edward B., A 2003, 'Ordering Fault-Prone Software Modules', Software Quality Journal,no.1,p.19,

[2] Taghi M., K, Xiaojing, Y, Edward B., A, Wendell D., J & John P., H 2002, 'Uncertain Classification of Fault-Prone Software Modules', Empirical Software Engineering: an International Journal, no. 4,p.297,

[3] Witten, Ian H., et al. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2016.