EXPERIMENT: 3

(2K17/SE/79 PARV GUPTA)

AIM: Consider any prediction model of your choice.

a)Analyze the dataset that is given as a input to the prediction model

b) Find out the quartiles for the used dataset.

c) Analyze the performance of a model using various performance metrics.

THEORY:

The predictive model used for this experiment is from a research paper "**Software Fault Proneness Prediction Using Support Vector Machines by Yogesh Singh, Arvinder Kaur, Ruchika Malhotra,** 2009 [Proceedings of the World Congress on Engineering 2009, London, U.K.] "

a) Analyze the dataset that is given as a input to the prediction model

DATASET USED:

Public domain KC1 NASA data set (Class level data) of Object Oriented Metrics.

DATASET LINK:

http://promise.site.uottawa.ca/SERepository/datasets/kc1-class-level-numericdefect.arff

This study makes use of the public domain data set KC1 from the NASA Metrics Data Program . The data in KC1 was collected from a storage management system for receiving/processing ground data, which was implemented in the C++ programming language. This system consists of 145 classes that comprise 2107 methods,

with 40K lines of code. KC1 provides both class-level and method-level static metrics. At the class level, values of 10 metrics are computed including seven metrics given by Chidamber and Kemerer . The seven OO metrics taken in this study are defined in the Table below.

METRICS STUDIED

Metric	Definition
Coupling between Objects (CBO)	CBO for a class is count of the number of other classes to which it is coupled and vice versa.
Lack of Cohesion (LCOM)	For each data field in a class, the percentage of the methods in the class using that data field; the percentages are averaged then subtracted from 100%.
Number of Children (NOC)	The NOC is the number of immediate subclasses of a class in a hierarchy.
Depth of Inheritance (DIT)	The depth of a class within the inheritance hierarchy is the maximum number of steps from the class node to the root of the tree and is measured by the number of ancestor classes.
Weighted Methods per Class (WMC)	A count of methods implemented within a class.
Response for a Class (RFC)	A count of methods implemented within a class plus the number of methods accessible to an object class due to inheritance.
Source Lines Of Code (SLOC)	It counts the lines of code.

Statistical Summary of dataset:

```
X.describe()
```

	CBO	DIT	LCOM	NOC	RFC	WMC	SLOC
count	2095.000000	2095.000000	2095.000000	2095.000000	2095.000000	2095.000000	2095.000000
mean	0.132220	0.950835	2.843914	2.550358	14.619570	21.375384	20.377566
std	0.704866	3.094650	3.912850	3.386475	24.241343	21.506699	29.838073
min	0.000000	0.000000	1.000000	1.000000	0.000000	0.000000	1.000000
25%	0.000000	0.000000	1.000000	1.000000	0.000000	5.330000	3.000000
50%	0.000000	0.000000	1.000000	1.000000	5.000000	14.450000	9.000000
75%	0.000000	0.000000	3.000000	3.000000	17.000000	29.890000	24.000000
max	12.000000	44.000000	45.000000	45.000000	262.000000	193.060000	288.000000

b) Find out the quartiles for the used dataset.

The quantile( ) function of NumPy is used to find quartiles of the dataset for each of the seven OO metrics used.

```
L=['CBO', 'DIT', 'LCOM', 'NOC', 'RFC', 'WMC', 'SLOC']
for feature in L:
  first_quartile = np.quantile(X[feature],0.25)
  second_quartile = np.quantile(X[feature],0.5)
  third_quartile = np.quantile(X[feature],0.75)
  print(" first_quartile for", feature, " is ", first_quartile )
  print(" second_quartile for", feature, " is ", second_quartile )
  print(" third_quartile for", feature, " is ", third_quartile )
  print("\n")
```

c) Analyze the performance of a model using various performance metrics.

The Model evaluation metrics used are:

Classification Accuracy: Overall, how often is the classifier correct?

```
[42] # use float to perform true division, not integer division
     print((TP + TN) / float(TP + TN + FP + FN))
     print(metrics.accuracy_score(y_test, y_pred))

     0.8336134453781513
     0.8336134453781513
```

Classification Error: Overall, how often is the classifier incorrect?

```
classification_error = (FP + FN) / float(TP + TN + FP + FN)

print(classification_error)
print(1 - metrics.accuracy_score(y_test, y_pred))

0.16638655462184873
0.1663865546218487
```

Sensitivity: When the actual value is positive, how often is the prediction correct?

Good model aims to maximise Sensitivity..Also known as "True Positive Rate" or "Recall" TP /TP + FN

```
sensitivity = TP / float(FN + TP)

print(sensitivity)
print(metrics.recall_score(y_test, y_pred))

0.04854368932038835
0.04854368932038835
```

Specificity: When the actual value is negative, how often is the prediction correct?

Good model aims to maximise Specificity. TN / TN + FP

```
specificity = TN / (TN + FP)

print(specificity)

0.9979674796747967
```

False Positive Rate: When the actual value is negative, how often is the prediction incorrect?

```
false_positive_rate = FP / float(TN + FP)

print(false_positive_rate)
print(1 - specificity)
```

```
0.0020325203252032522
0.002032520325203291
```

Precision: When a positive value is predicted, how often is the prediction correct?

```
[47] precision = TP / float(TP + FP)

     print(precision)
     print(metrics.precision_score(y_test, y_pred))
```

```
0.8333333333333334
0.8333333333333334
```

Completeness Score: It is the Completeness metric of a cluster labeling given a ground truth.A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster.

```
[48] print(1-completeness_score(y_test, y_pred))
```

```
0.8107354492580748
```

Below table (from paper) summaries the results of evaluating the performance of the model using various metrics:

TABLE 4
. RESULTS OF 10-CROSS VALIDATION OF MODELS

SUPPORT VECTOR MACHINE	
	Model
Cutoff point	0.45
Sensitivity	69.49
Specificity	82.55
Precision	77.24
Completeness	79.59
AUC	0.89

CONCLUSION:

Using the Support Vector Machine Predictive Model for Software Fault Proneness Prediction using OO metrics of KC1 NASA dataset,

we analysed the dataset, found out quartiles and analysed the performance of the model using various performance metrics.