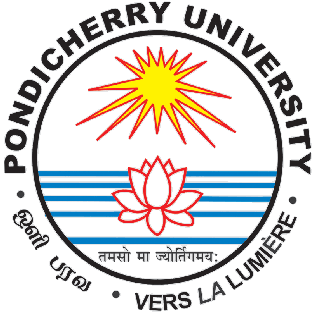
****

**PONDICHERRY UNIVERISTY**

**Department of  
Computer Science**

M.Sc. 1st Year

**Study and Working  
of Weka Tool**

**Sawon Bhattacharya**

19370045

**Rajarsi Saha**

19370037

**Aarko Datta**

19370001

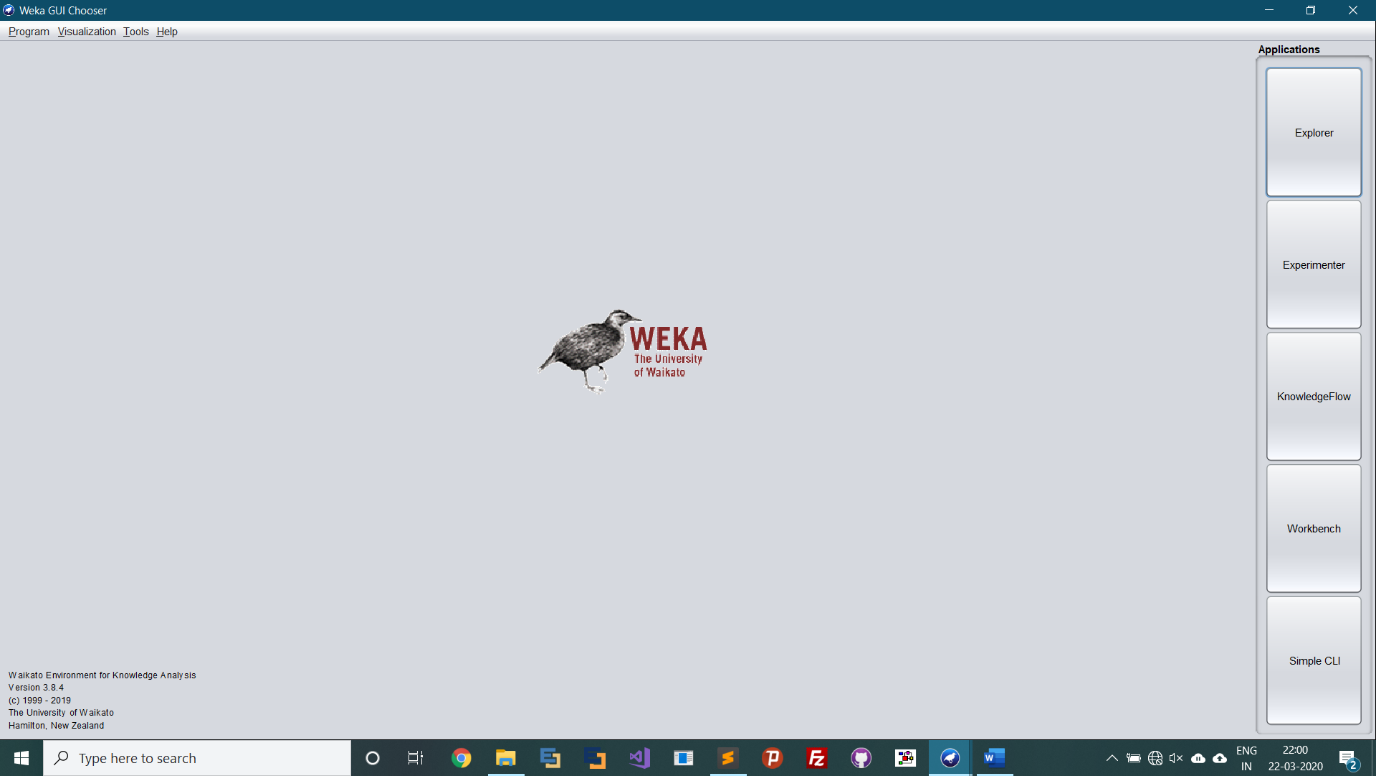
**STUDY AND WORKING OF WEKA TOOL**

**Aim:** To study and analyze data using WEKA tool.

**About:** **Waikato Environment for Knowledge Analysis (Weka),** developed at the [University of Waikato](https://en.wikipedia.org/wiki/University_of_Waikato), [New Zealand](https://en.wikipedia.org/wiki/New_Zealand). Weka supports several standard [data mining](https://en.wikipedia.org/wiki/Data_mining) tasks, more specifically, data preprocessing, [clustering](https://en.wikipedia.org/wiki/Data_clustering), [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis), visualization, and [feature selection](https://en.wikipedia.org/wiki/Feature_selection). All of Weka's techniques are predicated on the assumption that the data is available as one flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric or nominal attributes, but some other attribute types are also supported). Weka provides access to [SQL](https://en.wikipedia.org/wiki/SQL) [databases](https://en.wikipedia.org/wiki/Database) using [Java Database Connectivity](https://en.wikipedia.org/wiki/Java_Database_Connectivity) and can process the result returned by a database query. Weka provides access to [deep learning](https://en.wikipedia.org/wiki/Deep_learning) with [Deeplearning](https://en.wikipedia.org/wiki/Deeplearning4j) It is not capable of multi-relational data mining, but there is separate software for converting a collection of linked database tables into a single table that is suitable for processing using Weka.

**Procedure:**

**CLASSIFIER ALGORITHM**



**Step 1:** Open and load WEKA 3.8.4.

**Step 2:** Open the explorer window.

**Step 3:** Open notepad and create a file with .arff extension with the following attributes and values.

**Relevant Information:**

Vina conducted a comparison test of her rule-based system, BEAGLE, the nearest-neighbor algorithm, and discriminant analysis. BEAGLE is a product available through VRS Consulting, Inc.; 4676 Admiralty Way, Suite 206; Marina Del Ray, CA 90292 (213) 827-7890 and FAX: -3189. In determining whether the glass was a type of "float" glass or not, the following results were obtained (# incorrect answers):

Type of Sample Beagle NN DA

Windows that were float processed (87) 10 12 21

Windows that were not: (76) 19 16 22

The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence...if it is correctly identified!

Number of Instances: 214

Number of Attributes: 10 (including an Id#) plus the class attribute

-- all attributes are continuously valued

Attribute Information:

1. Id number: 1 to 214

2. RI: refractive index

3. Na: Sodium (unit measurement: weight percent in corresponding oxide, as are attributes 4-10)

4. Mg: Magnesium

5. Al: Aluminum

6. Si: Silicon

7. K: Potassium

8. Ca: Calcium

9. Ba: Barium

10. Fe: Iron

11. Type of glass: (class attribute)

-- 1 building\_windows\_float\_processed

-- 2 building\_windows\_non\_float\_processed

-- 3 vehicle\_windows\_float\_processed

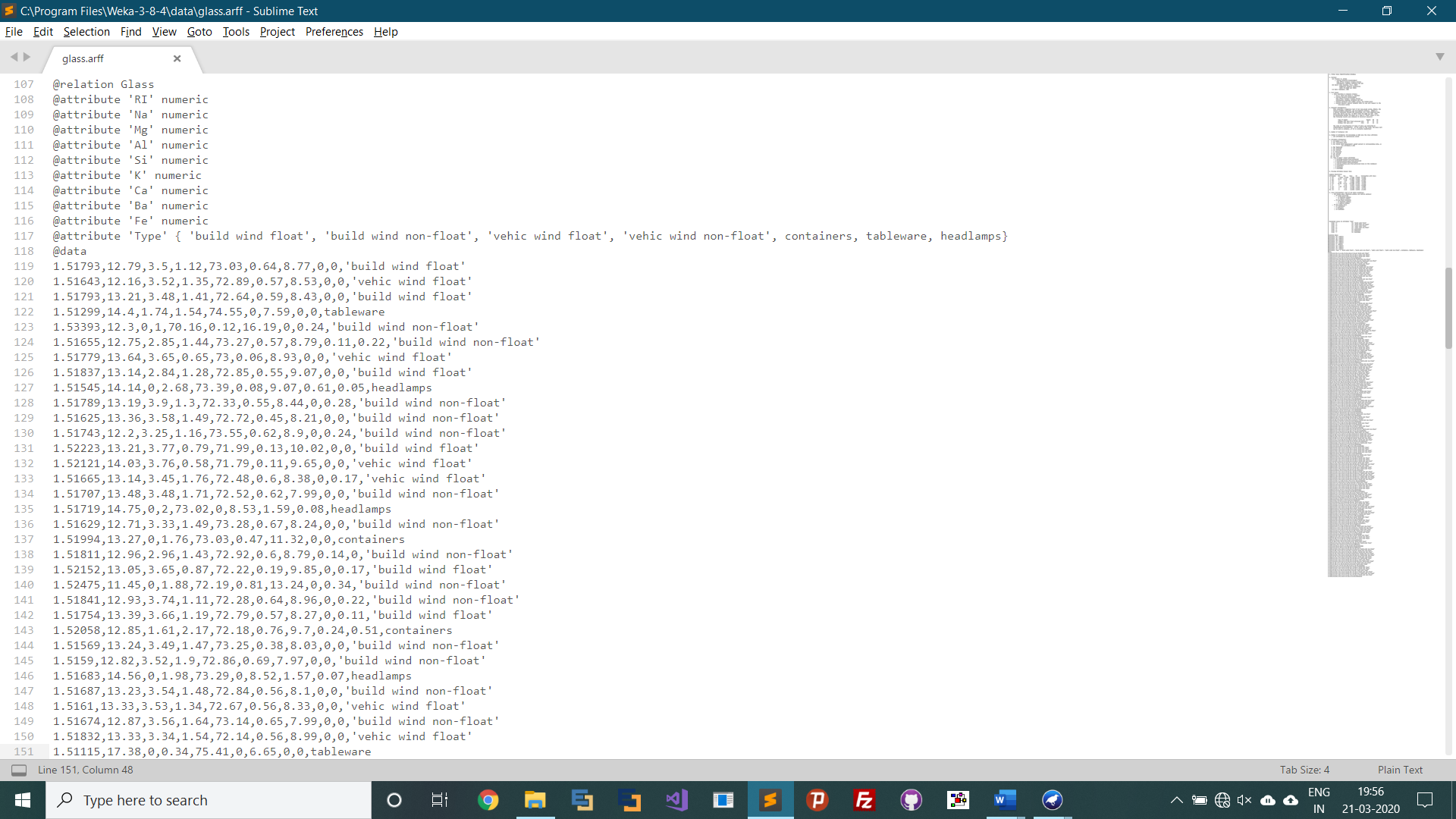
-- 4 vehicle\_windows\_non\_float\_processed (none in this database)

-- 5 containers

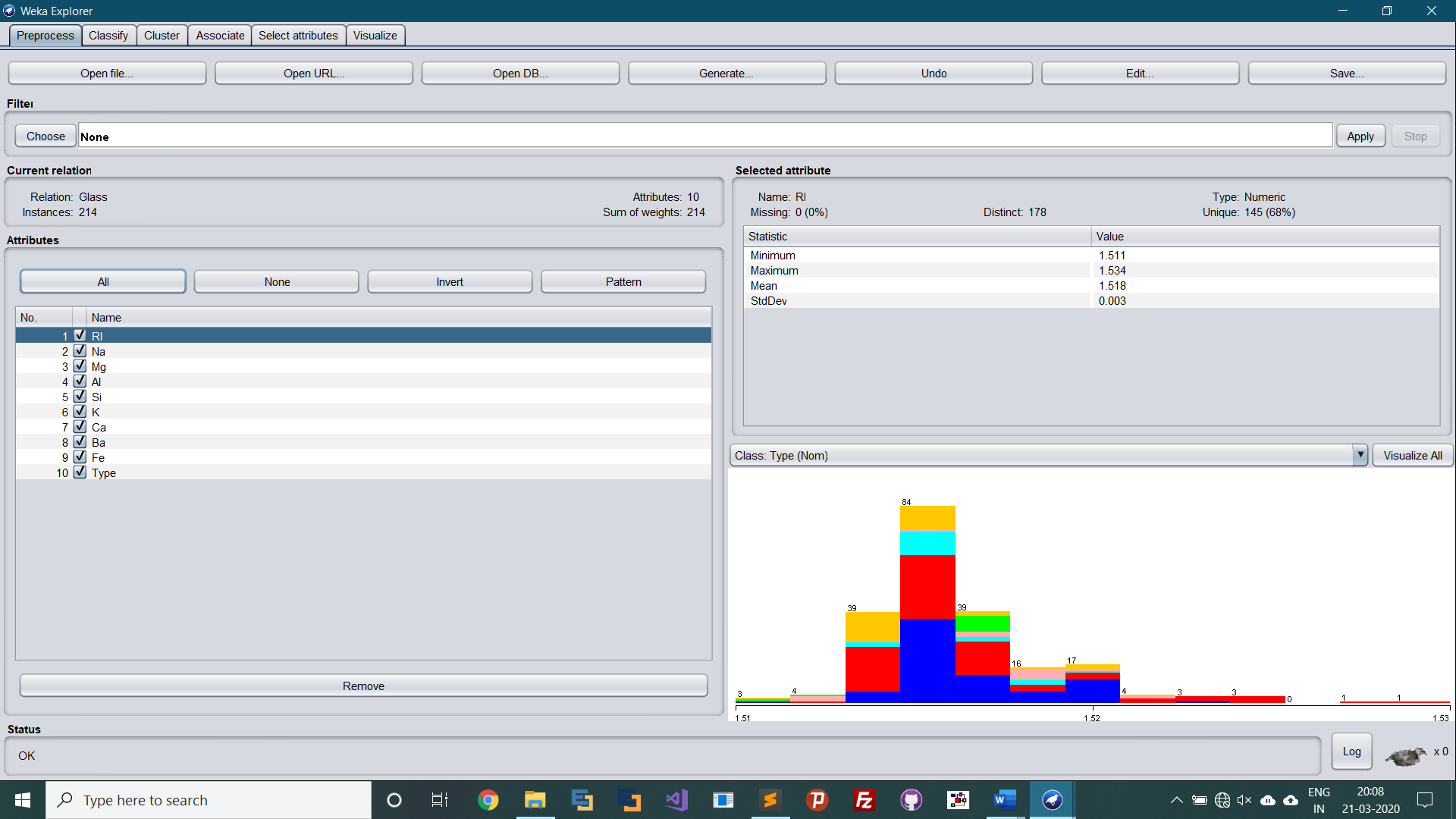
-- 6 tableware

-- 7 headlamps

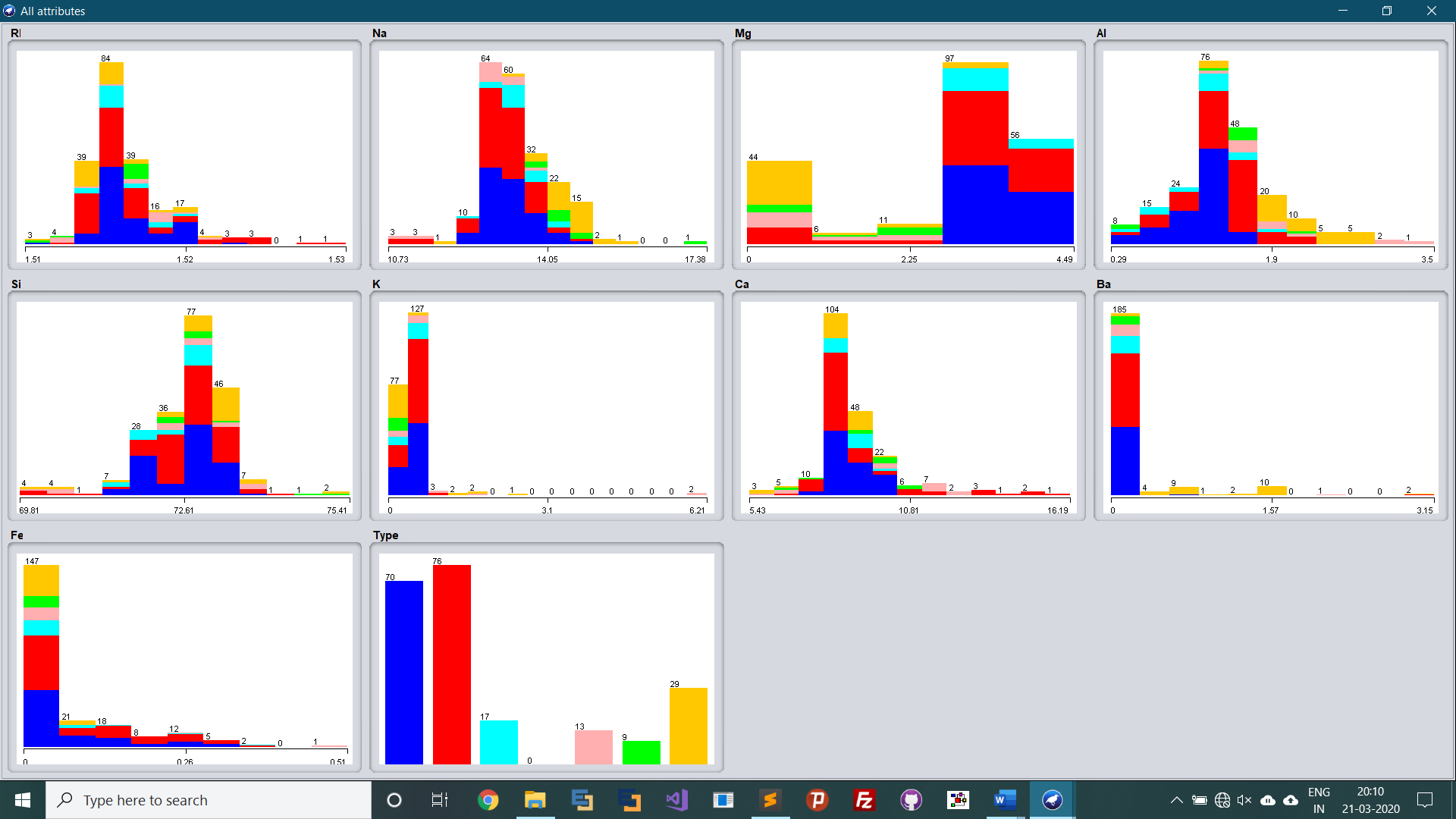
Missing Attribute Values: None



**Step 4:** After opening the explorer window select the option “Open file” and select the training dataset which we created previously.

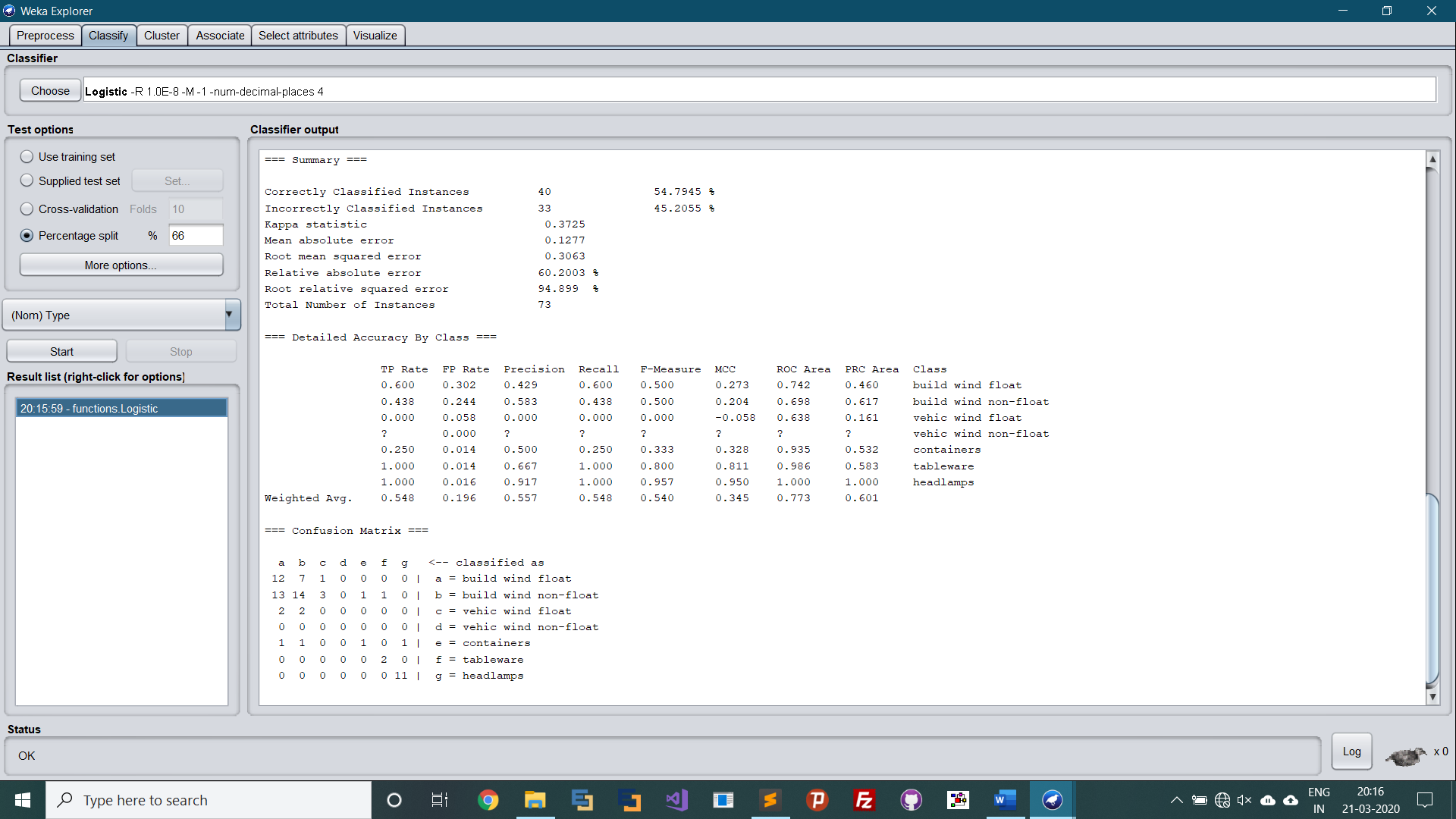


**Step 5:** Following are the visualizations of the records in the dataset.



**Step 6:** Open the “Classify” tab from the top most menu and choose Logistic algorithm from the classifier option available.

\*[Select the “percentage split” option and set it to 66% (This will divide your dataset into training and testing set with 66% and 34% respectively.)]



**Step 7:** Click on “start” to build the model.

Following is the statistical result we obtain after applying the Logistic regression upon our data.

=== Run information ===

Scheme: weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 -num-decimal-places 4

Relation: Glass

Instances: 214

Attributes: 10

RI

Na

Mg

Al

Si

K

Ca

Ba

Fe

Type

Test mode: split 66.0% train, remainder test

=== Classifier model (full training set) ===

Logistic Regression with ridge parameter of 1.0E-8

Coefficients...

Class

Variable build wind float build wind non-float vehic wind float vehic wind non-float containers tableware

===========================================================================================================================================================

RI -46030.7875 -45790.3017 -47513.4102 -10339.1857 -45731.8053 -17742.7183

Na -215.5097 -219.9798 -218.6246 -35.5832 -215.991 56.571

Mg -2.3608 -9.2984 -5.0187 16.825 -8.4554 75.0936

Al -123.6754 -124.3886 -126.2082 -37.6918 -107.4788 120.3998

Si -171.8241 -177.0562 -178.11 -12.6877 -169.2975 29.2027

K -70.3905 -75.3404 -76.9881 -5.6794 -73.2595 -348.9329

Ca 45.21 39.6415 44.4346 8.0981 43.6207 74.2154

Ba -70.4019 -77.5189 -75.1364 -46.0028 -73.8273 -334.0511

Fe 1280.3022 1282.3597 1279.2274 403.8231 1274.6903 -1452.535

Intercept 85211.8621 85361.5465 87981.1698 16953.547 84585.4682 23103.5733

Odds Ratios...

Class

Variable build wind float build wind non-float vehic wind float vehic wind non-float containers tableware

===========================================================================================================================================================

RI 0 0 0 0 0 0

Na 0 0 0 0 0 3.702369673866181E24

Mg 0.0943 0.0001 0.0066 20277603.9281 0.0002 4.09950622341485E32

Al 0 0 0 0 0 1.945187281424097E52

Si 0 0 0 0 0 4.814835754955086E12

K 0 0 0 0.0034 0 0

Ca 4.3097391577902875E19 1.64469464281863008E17 1.984708698854199E19 3288.063 8.7950184263749007E18 1.7034550420268027E32

Ba 0 0 0 0 0 0

Fe Infinity Infinity Infinity 2.38858229515231E175 Infinity 0

Time taken to build model: 0.12 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances 40 54.7945 %

Incorrectly Classified Instances 33 45.2055 %

Kappa statistic 0.3725

Mean absolute error 0.1277

Root mean squared error 0.3063

Relative absolute error 60.2003 %

Root relative squared error 94.899 %

Total Number of Instances 73

=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class

0.600 0.302 0.429 0.600 0.500 0.273 0.742 0.460 build wind float

0.438 0.244 0.583 0.438 0.500 0.204 0.698 0.617 build wind non-float

0.000 0.058 0.000 0.000 0.000 -0.058 0.638 0.161 vehic wind float

? 0.000 ? ? ? ? ? ? vehic wind non-float

0.250 0.014 0.500 0.250 0.333 0.328 0.935 0.532 containers

1.000 0.014 0.667 1.000 0.800 0.811 0.986 0.583 tableware

1.000 0.016 0.917 1.000 0.957 0.950 1.000 1.000 headlamps

Weighted Avg. 0.548 0.196 0.557 0.548 0.540 0.345 0.773 0.601

=== Confusion Matrix ===

a b c d e f g <-- classified as

12 7 1 0 0 0 0 | a = build wind float

13 14 3 0 1 1 0 | b = build wind non-float

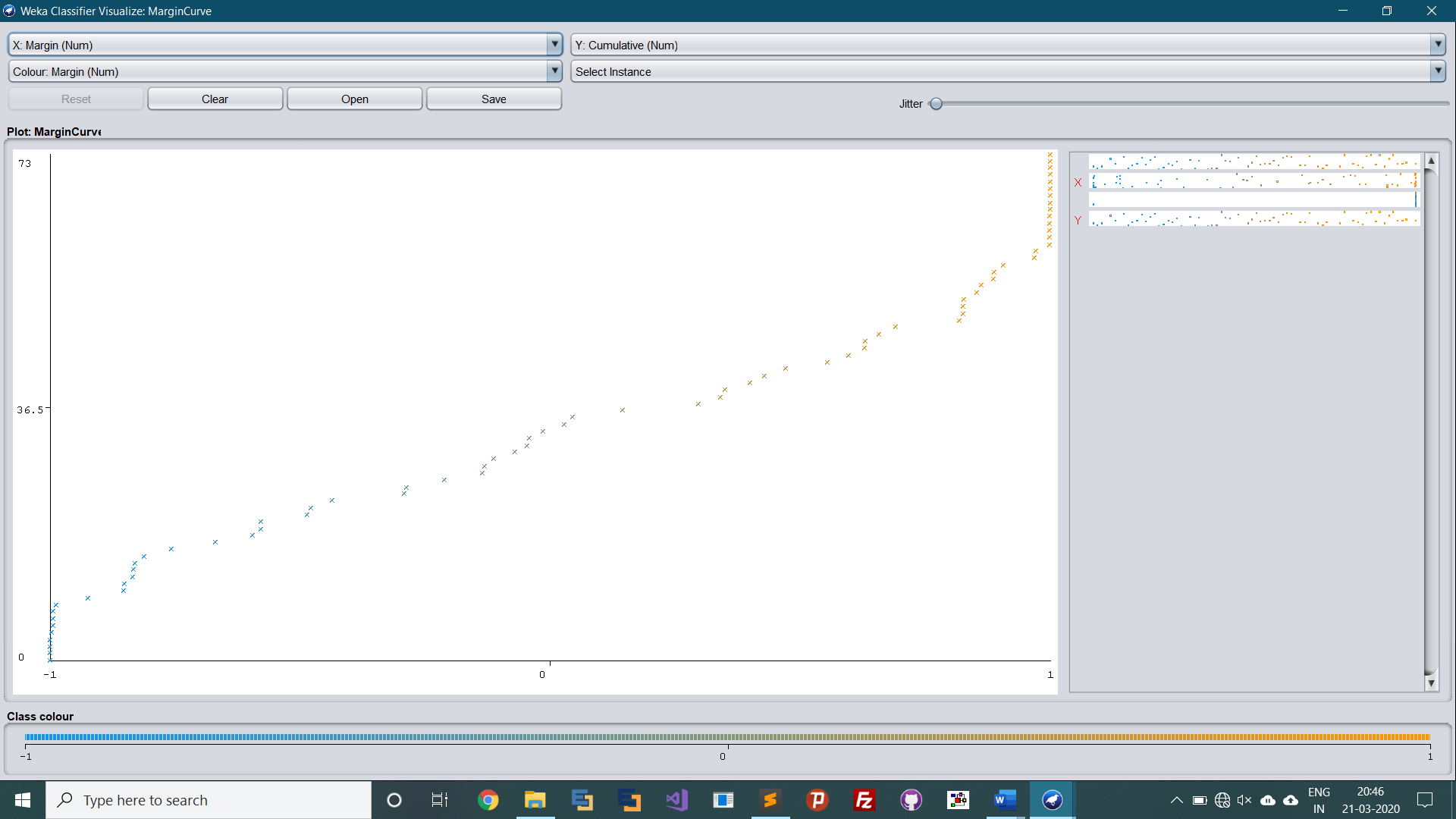
2 2 0 0 0 0 0 | c = vehic wind float

0 0 0 0 0 0 0 | d = vehic wind non-float

1 1 0 0 1 0 1 | e = containers

0 0 0 0 0 2 0 | f = tableware

0 0 0 0 0 0 11 | g = headlamps



**Results:**

From the above analysis we observed that our machine learning model trained on **66%** of data when applied to the remaining **34%** of testing data has predicted around **54.7945 %** correctly.

**CLUSTERING ALGORITHM**

**Procedure:**

**Step 1:** Open notepad and create a file with .arff extension with the following attributes and values.

**Relevant Information:**

The instances were drawn randomly from a database of 7 outdoor images. The images were handsegmented to create a classification for every pixel.

Each instance is a 3x3 region.

Number of Instances: Training data: 210 Test data: 2100

Number of Attributes: 19 continuous attributes

Attribute Information:

1. region-centroid-col: the column of the center pixel of the region.

2. region-centroid-row: the row of the center pixel of the region.

3. region-pixel-count: the number of pixels in a region = 9.

4. short-line-density-5: the results of a line extractoin algorithm that

counts how many lines of length 5 (any orientation) with

low contrast, less than or equal to 5, go through the region.

5. short-line-density-2: same as short-line-density-5 but counts lines

of high contrast, greater than 5.

6. vedge-mean: measure the contrast of horizontally

adjacent pixels in the region. There are 6, the mean and

standard deviation are given. This attribute is used as

a vertical edge detector.

7. vegde-sd: (see 6)

8. hedge-mean: measures the contrast of vertically adjacent

pixels. Used for horizontal line detection.

9. hedge-sd: (see 8).

10. intensity-mean: the average over the region of (R + G + B)/3

11. rawred-mean: the average over the region of the R value.

12. rawblue-mean: the average over the region of the B value.

13. rawgreen-mean: the average over the region of the G value.

14. exred-mean: measure the excess red: (2R - (G + B))

15. exblue-mean: measure the excess blue: (2B - (G + R))

16. exgreen-mean: measure the excess green: (2G - (R + B))

17. value-mean: 3-d nonlinear transformation

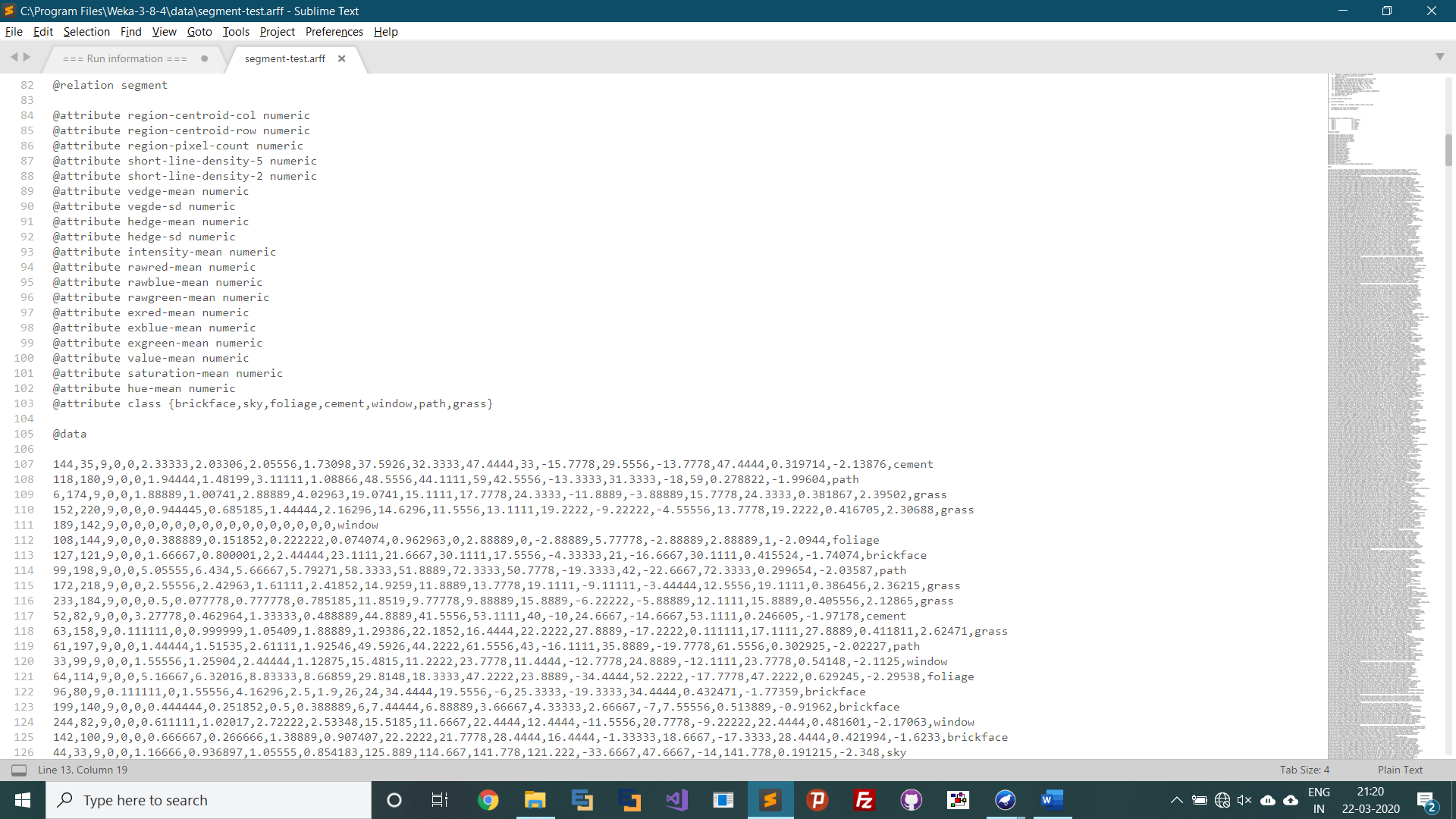
of RGB. (Algorithm can be found in Foley and VanDam, Fundamentals

of Interactive Computer Graphics)

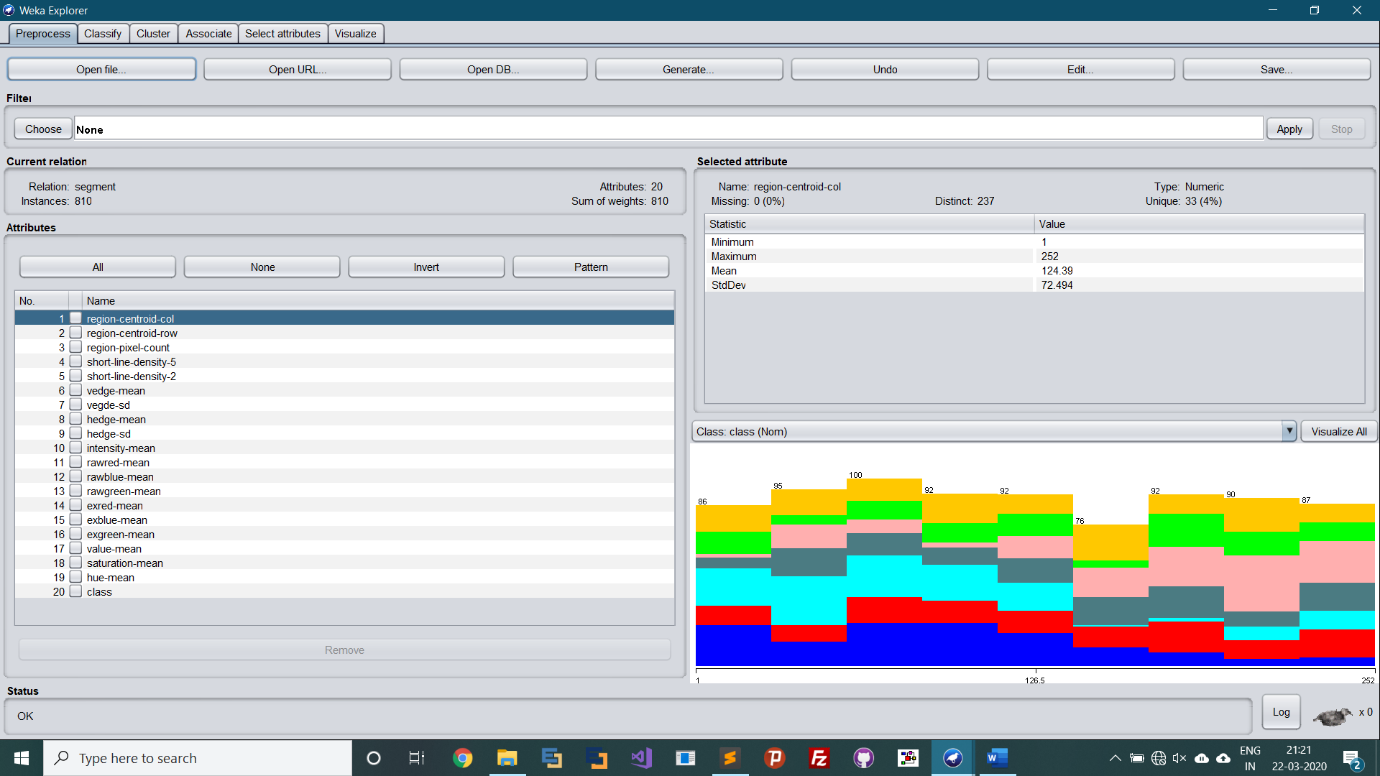
18. saturatoin-mean: (see 17)

19. hue-mean: (see 17)

Missing Attribute Values: None

****

**Step 2:** After opening the explorer window select the option “Open file” and select the training dataset which we created previously.

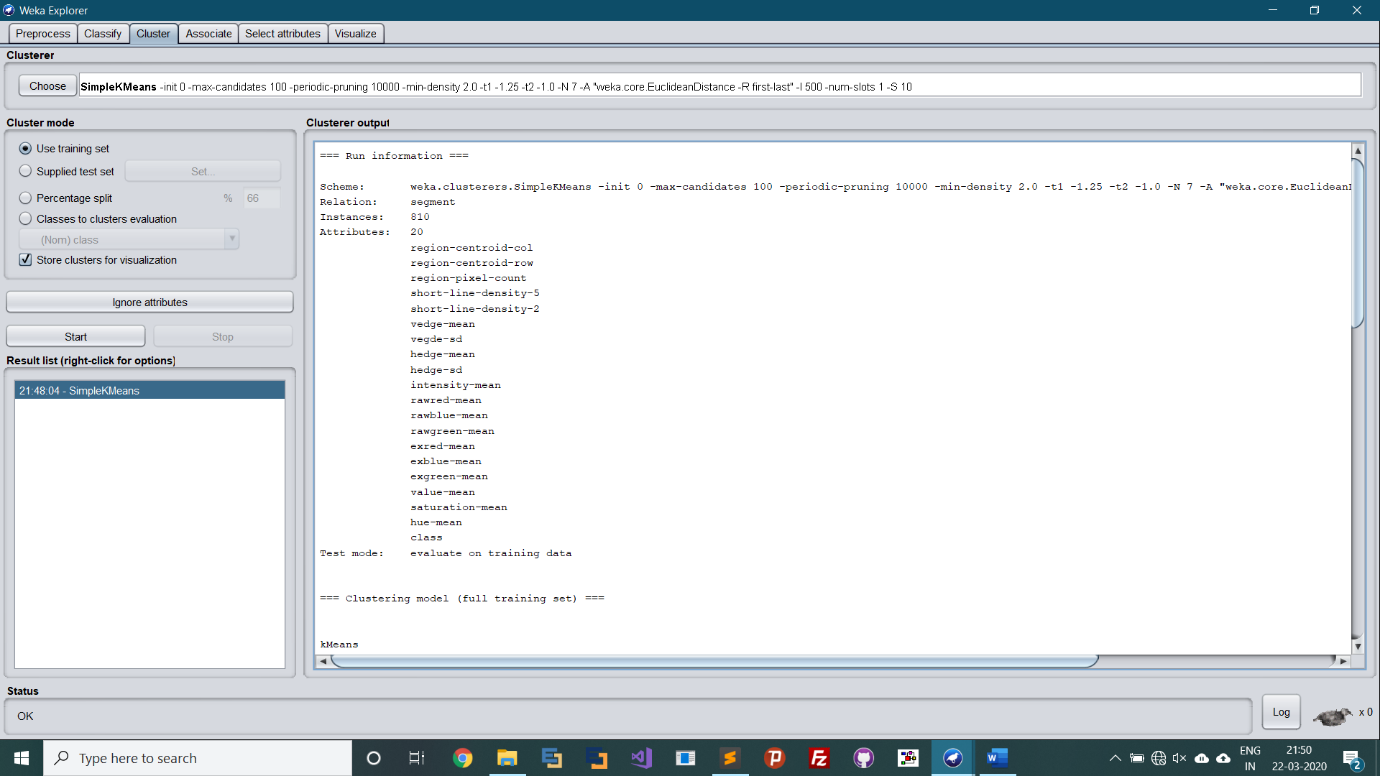
****

**Step 3:** Following are the visualizations of the records in the dataset.

****

**Step 4:** Open the “cluster” tab from the top most menu and choose SimpleKmeans algorithm from the classifier option available.

**Step 5:** Click on the algorithm and in the weka.gui.GenericObjectEditor form set numClusters to 7. (We are clustering the above data into 7 clusters viz. brickface, sky, foliage, cement, window, path, grass)



**Step 6:** Select the “use training set” option and click on “start” to build the model.

(Following is the statistical result we obtain after applying the SimpleKmeans algorithm upon our data.)

=== Run information ===

Scheme: weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 7 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10

Relation: segment

Instances: 810

Attributes: 20

region-centroid-col

region-centroid-row

region-pixel-count

short-line-density-5

short-line-density-2

vedge-mean

vegde-sd

hedge-mean

hedge-sd

intensity-mean

rawred-mean

rawblue-mean

rawgreen-mean

exred-mean

exblue-mean

exgreen-mean

value-mean

saturation-mean

hue-mean

class

Test mode: evaluate on training data

=== Clustering model (full training set) ===

kMeans

======

Number of iterations: 5

Within cluster sum of squared errors: 182.85491701731036

Initial starting points (random):

Cluster 0: 103,125,9,0,0,0.944445,0.827759,0.833333,0.691215,1.77778,0.444444,3.88889,1,-4,6.33333,-2.33333,3.88889,0.923809,-2.23756,foliage

Cluster 1: 147,182,9,0,0,2.27778,1.70511,2.94444,1.59745,43.5926,40.3333,52.5556,37.8889,-9.77778,26.8889,-17.1111,52.5556,0.278669,-1.91917,path

Cluster 2: 105,65,9,0,0,0.944443,0.611618,1.5,1.22474,123,111.667,140,117.333,-34,51,-17,140,0.202365,-2.30399,sky

Cluster 3: 202,129,9,0,0,0.222222,0.544331,3.44444,3.80448,2.2963,2.44444,2.55556,1.88889,0.444444,0.777778,-1.22222,2.66667,0.096561,-1.10139,window

Cluster 4: 11,163,9,0,0,1.05556,0.928958,2.5,1.31234,19.8148,17.4444,16.6667,25.3333,-7.11111,-9.44444,16.5556,25.3333,0.344267,2.00168,grass

Cluster 5: 6,154,9,0,0,0.111111,0.02963,0.111111,0.02963,0.074074,0,0.222222,0,-0.222222,0.444444,-0.222222,0.222222,0.222222,-2.0944,window

Cluster 6: 247,159,9,0,0,2.55556,5.05185,15.7222,168.952,35.7778,30.4444,45,31.8889,-16,27.6667,-11.6667,45.4444,0.289218,-2.54935,path

Missing values globally replaced with mean/mode

Final cluster centroids:

Cluster#

Attribute Full Data 0 1 2 3 4 5 6

(810.0) (122.0) (110.0) (110.0) (126.0) (123.0) (125.0) (94.0)

======================================================================================================

region-centroid-col 124.3901 89.3607 133.5909 134.0727 164.6587 124.3415 93.976 134.2872

region-centroid-row 122.7827 111.9016 97.5182 47.5818 110.6667 206.0081 104.072 186.6915

region-pixel-count 9 9 9 9 9 9 9 9

short-line-density-5 0.0128 0.0055 0.0152 0.0131 0.0035 0.0244 0.0196 0.0071

short-line-density-2 0.0043 0.0091 0.004 0 0.0044 0.0018 0.0009 0.0106

vedge-mean 1.7607 2.1375 2.8934 0.8379 1.4264 1.6766 1.092 2.4734

vegde-sd 3.8186 11.3117 8.0811 0.5863 1.6422 1.5839 1.0438 2.4197

hedge-mean 2.2213 2.4117 2.6056 1.2258 1.2196 2.1951 1.4849 5.0455

hedge-sd 5.3832 11.9823 5.7791 0.8182 3.23 2.3598 1.2548 14.0293

intensity-mean 34.9162 6.4581 45.5838 117.4061 10.4591 15.6745 14.3416 48.1576

rawred-mean 30.8538 3.5 40.7404 105.9909 8.0088 12.6549 14.6071 42.8995

rawblue-mean 41.7221 10.8752 56.0182 134.4859 15.1058 14.1256 18.1902 59.5544

rawgreen-mean 32.1727 4.9991 39.9929 111.7414 8.2628 20.243 10.2276 42.0189

exred-mean -12.1872 -8.8743 -14.5303 -34.2454 -7.351 -9.0587 0.7964 -15.7742

exblue-mean 20.4177 13.2514 31.303 51.2394 13.94 -4.6468 11.5458 34.1903

exgreen-mean -8.2305 -4.377 -16.7727 -16.9939 -6.5891 13.7055 -12.3422 -18.4161

value-mean 42.7288 10.9071 56.0202 134.4859 15.1464 20.364 18.4853 59.5756

saturation-mean 0.4386 0.7987 0.2985 0.2156 0.497 0.4071 0.4868 0.2953

hue-mean -1.3292 -2.2276 -2.0183 -2.3022 -1.8751 2.174 -1.3329 -2.0659

class window foliage cement sky window grass brickface path

Time taken to build model (full training data) : 0.03 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 122 ( 15%)

1 110 ( 14%)

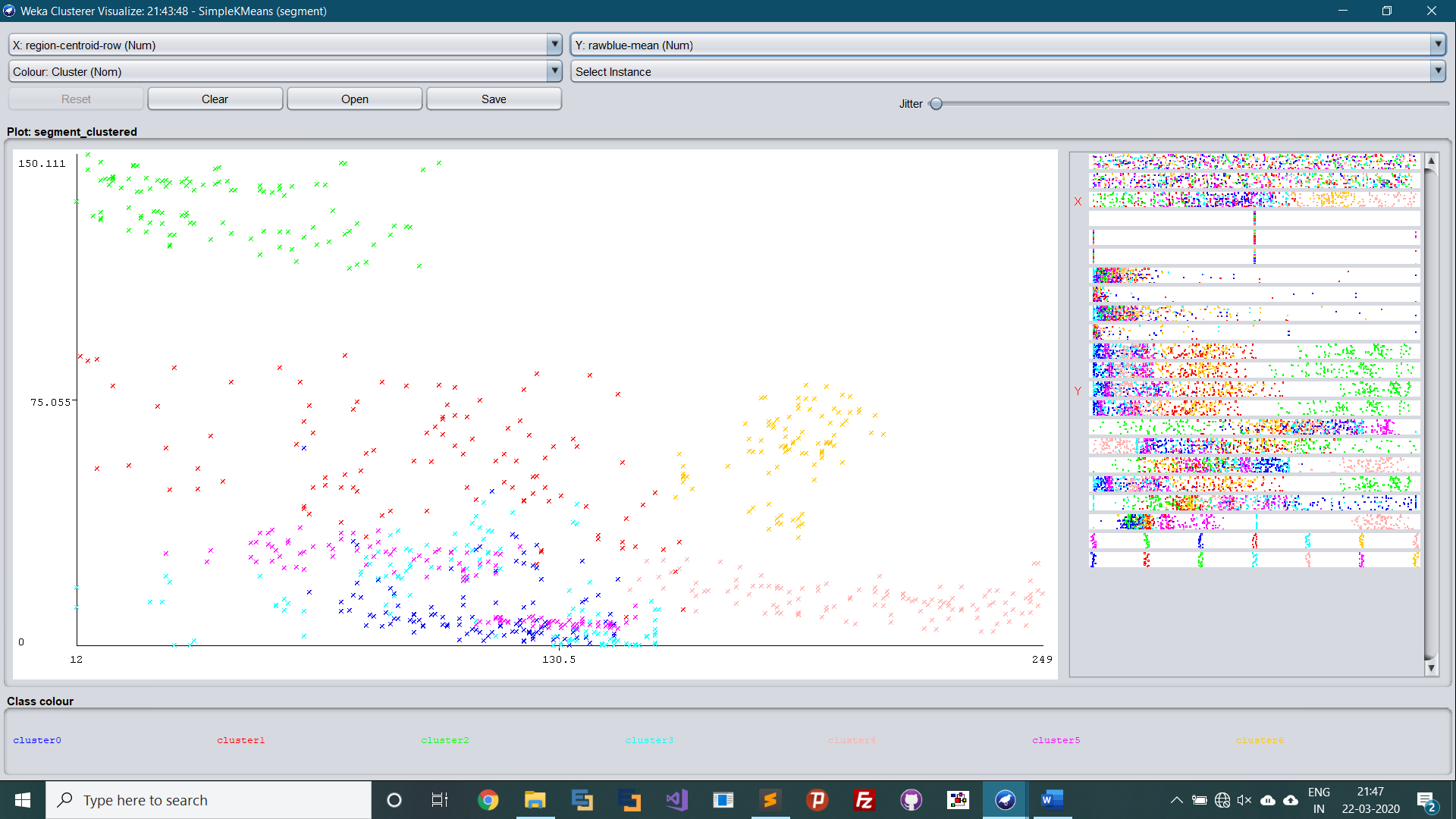
2 110 ( 14%)

3 126 ( 16%)

4 123 ( 15%)

5 125 ( 15%)

6 94 ( 12%)



**Results:**

From the above visualization we can see that the SimpleKmeans algorithm has clustered our data into 7 clusters, based on the attributes on the data i.e. brickface, sky, foliage, cement, window, path, grass. The SimpleKmeans algorithm classifies the dataset into 7 clusters based on the attributes associated with each of the rows and map it into the following graph.