

(A)i)

$$\text{Gini}(\text{parent}) = 1 - (8/16)^2 - (8/16)^2 = 1 - (1/2)^2 - (1/2)^2 = 1 - (1/4) - (1/4) = (4/4) - (2/4) = 2/4 = 0.5$$

Colour:

$$\text{Gini}(\text{Perpule}) = 1 - (4/6)^2 - (2/6)^2 = 1 - (2/3)^2 - (1/3)^2 = 1 - (4/9) - (1/9) = (9/9) - (5/9) = 4/9 = 0.44$$

$$\text{Gini}(\text{Red}) = 1 - (4/4)^2 = 0$$

$$\text{Gini}(\text{Blue}) = 1 - (4/6)^2 - (2/6)^2 = 4/9 = 0.44$$

$$\text{Gini}(\text{Colour\_split}) = (6/16 * 0.44) + (4/16 * 0) + (6/16 * 0.44) = 2 * (6/16 * 0.44) = 2 * 0.165 = 0.33$$

Height:

$$\text{Gini}(\text{Tall}) = 1 - (4/9)^2 - (5/9)^2 = 1 - (16/81) - (25/81) = (81/81) - (41/81) = 40/81 = 0.49$$

$$\text{Gini}(\text{Short}) = 1 - (4/7)^2 - (3/7)^2 = 1 - (16/49) - (9/49) = (49/49) - (25/49) = (24/49) = 0.489 = 0.49$$

$$\text{Gini}(\text{Height\_split}) = (9/16 * 0.49) + (7/16 * 0.49) = 0.28 + 0.21 = 0.49$$

Stripes:

$$\text{Gini}(\text{Yes}) = 1 - (7/9)^2 - (2/9)^2 = 1 - (49/81) - (4/81) = (81/81) - (53/81) = 28/81 = 0.35$$

$$\text{Gini}(\text{No}) = 1 - (6/7)^2 - (1/7)^2 = 1 - (36/49) - (1/49) = (49/49) - (37/49) = 12/49 = 0.24$$

$$\text{Gini}(\text{Stripes\_split}) = (9/16 * 0.35) + (7/16 * 0.24) = 0.2 + 0.11 = 0.31$$

Texture:

$$\text{Gini}(\text{Rough}) = 1 - (2/3)^2 - (1/3)^2 = 1 - (4/9) - (1/9) = (9/9) - (5/9) = 4/9 = 0.44$$

$$\text{Gini}(\text{Smooth}) = 1 - (3/6)^2 - (3/6)^2 = 0.5$$

$$\text{Gini}(\text{Hairy}) = 1 - (3/7)^2 - (4/7)^2 = 1 - (9/49) - (16/49) = (49/49) - (25/49) = 24/49 = 0.49$$

$$\text{Gini}(\text{Texture\_split}) = (3/16 * 0.44) + (6/16 * 0.5) + (7/16 * 0.49) = 0.083 + 0.19 + 0.21 = 0.483 = 0.48$$

Επειδή το  $\text{Gini}(\text{Stripes\_split})$  έχει το χαμηλότερο Gini θα έχει και το υψηλότερο GAIN. Συνεπώς, το χαρακτηριστικό Stripes θα αποτελέσει τη ρίζα του δέντρου.

$$\text{GAIN}(\text{Stripes\_split}) = 0.5 - 0.31 = 0.19 \text{ το μεγαλύτερο.}$$

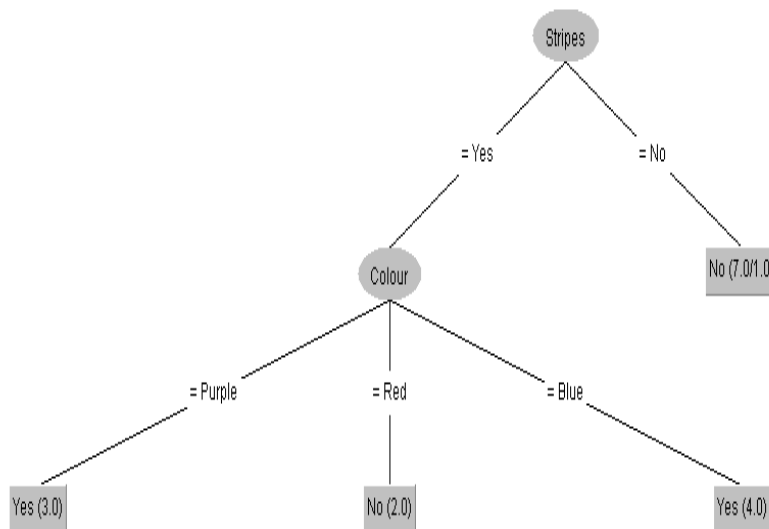
```

      |---yes (2/7) = yes
      |
Stripes|
      |
      |---no (1/6) = no

```

Άρα κατηγοριοποιούνται 3 εγγραφές λάθος (Στην περίπτωση Yes όπου 2 Yes κατηγοριοποιούνται ως No και στην περίπτωση No όπου 1 No κατηγοριοποιείται σε Yes). Συνεπώς, πετυχαίνουμε  $(16-3)/16 = 81.25\%$  ακρίβεια.

ii) ο αλγόριθμος J48 επιλέγει ως ρίζα του δέντρου το χαρακτηριστικό Stripes και πετυχαίνει ακρίβεια 93.75%.



iii) Για το συγκεκριμένο test dataset, το δέντρο που δημιούργησα πετυχαίνει ακρίβεια 50 %. Αυτό συμβαίνει γιατί κατηγοριοποιεί 1 Yes ως No και 1 No ως Yes  $\Rightarrow (4-2)/4 = 2/4 = 50\%$  ακρίβεια.

Αντίστοιχα το δέντρο απόφασης του WEKA πετυχαίνει 75% ακρίβεια με το μόνο λάθος να κατηγοριοποιεί 1 Yes σε No.

Παρακάτω φαίνονται τα αποτελέσματα που προέκυψαν από το δέντρο απόφασης του WEKA στο testing dataset.

Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Classifier

Choose

J48 -C 0.25 -M 2

Test options

☐ Use training set

☒ Supplied test set

☐ Cross-validation

☐ Percentage split

Set...

Folds 10

% 66

More options...

(Nom) Poisonous

Start

Stop

Result list (right-click for options)

18:03:53 - trees.J48

Classifier output

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances	3	75	%
Incorrectly Classified Instances	1	25	%
Kappa statistic	0.5		
Mean absolute error	0.25		
Root mean squared error	0.4345		
Relative absolute error	50	%	
Root relative squared error	86.8966	%	
Total Number of Instances	4		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,500	0,000	1,000	0,500	0,667	0,577	0,875	0,833	Yes
	1,000	0,500	0,667	1,000	0,800	0,577	0,875	0,833	No
Weighted Avg.	0,750	0,250	0,833	0,750	0,733	0,577	0,875	0,833	

=== Confusion Matrix ===

a b <-- classified as


1 1 | a = Yes

0 2 | b = No

Status

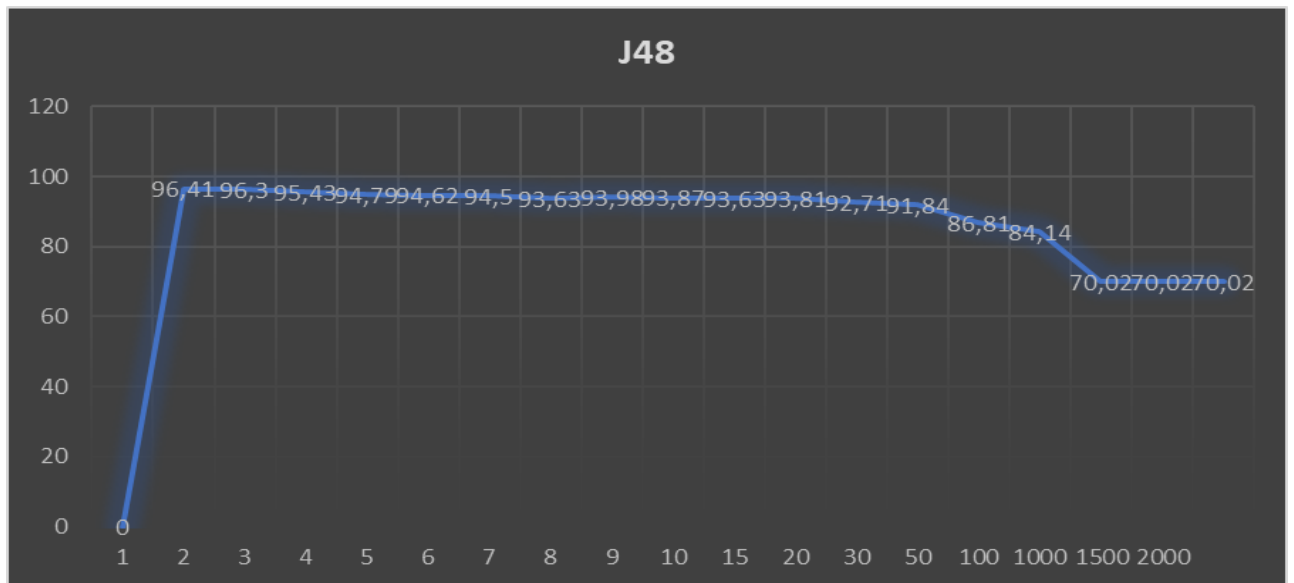
OK

Log

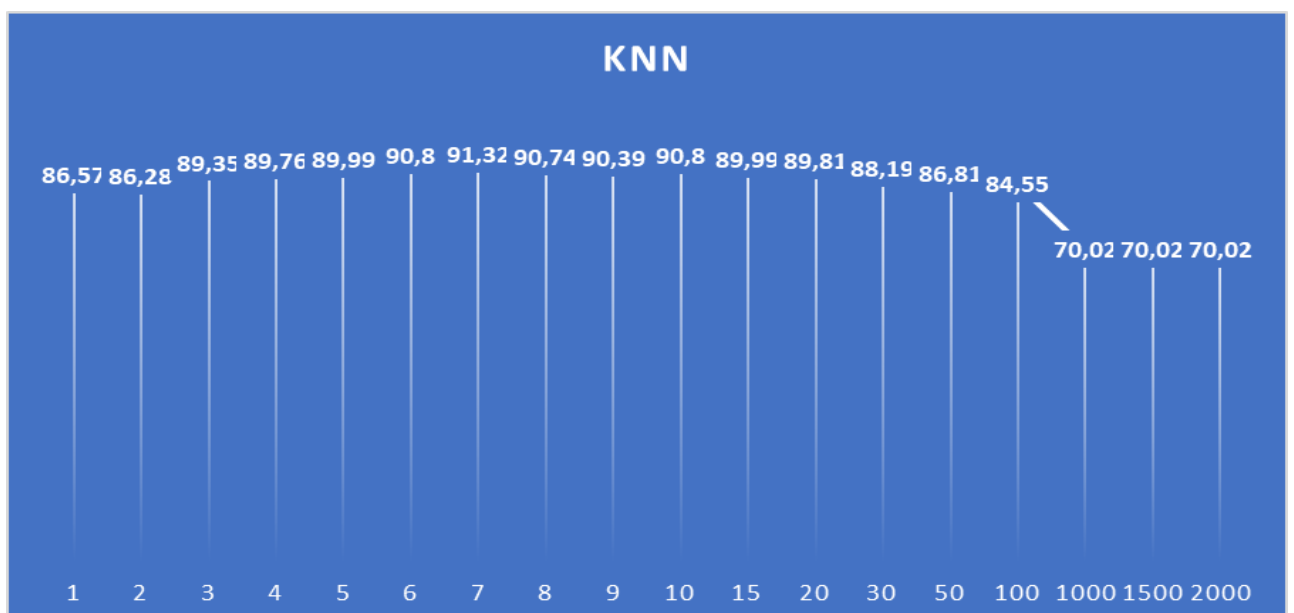
 x 0

(B)Παρακάτω φαίνονται σε μορφή γραφήματος οι πειραματισμοί μου με τις διάφορες τιμές των minNumObj και το k.

Αυτό που παρατηρούμε είναι ότι στο συγκεκριμένο αρχείο όσο περισσότερο κλαδεύουμε το δέντρο το ποσοστό ακρίβειας μειώνεται αρχικά με μικρό ρυθμό ,στη συνέχεια μειώνεται με μεγαλύτερο ρυθμό και κάποια στιγμή όπως παρατηρείται σταματάει να μειώνεται. Το καλύτερο δέντρο το βρήκα για minNumObj=1 διότι πετυχαίνει τη μεγαλύτερη ακρίβεια(96,41%).



Από την άλλη ο kNN στα αρχικά στάδια αύξησης του k όπως φαίνεται παρακάτω αυξάνεται το ποσοστό ακρίβειας και στη συνέχεια όσο το k γίνεται μεγαλύτερο μειώνεται περισσότερο το ποσοστό ακρίβειας μέχρι που σταθεροποιείται στο 70,02.Στην περίπτωση αυτή το καλύτερο k=7 διότι στο συγκεκριμένο αρχείο με k=7 πετυχαίνουμε τη μέγιστη ακρίβεια(91,32%).



Επίσης, αυτό που παρατηρούμε για το συγκεκριμένο αρχείο (car.arff) είναι πως τα δέντρα απόφασης είναι καλύτερα από το kNN. Γενικότερα, άλλοτε τα δέντρα απόφασης θα είναι καλύτερα από το kNN σε κάποια δεδομένα και άλλοτε ο kNN θα είναι καλύτερος από τα δέντρα απόφασης.

Ο πίνακας σύγχυσης για minNumObj=1 του J48:

=== Confusion Matrix ===

a	b	c	d	<-- classified as
1189	18	3	0	a = unacc
7	364	10	3	b = acc
0	10	58	1	c = good
0	8	2	55	d = vgood

(a=unacc): Από τις 1210 καταχωρίσεις κατηγοριοποιεί τις 21 λάθος (Στην περίπτωση που 18 unacc κατηγοριοποιούνται ως acc και 3 unacc ως good). Ποσοστό πρόβλεψης της κατηγορίας unacc είναι  $(1189/1210 * 100) = 98,3 \%$ . Προβλέπεται πολύ ικανοποιητικά η συγκεκριμένη κατηγορία.

(b=acc): Από τις 384 καταχωρίσεις κατηγοριοποιεί τις 20 λάθος (Στη περίπτωση που 7 acc κατηγοριοποιούνται ως unacc, 10 acc κατηγοριοποιούνται ως good και 3 acc κατηγοριοποιούνται ως vgood). Ποσοστό πρόβλεψης της κατηγορίας acc:  $364/384 * 100 = 94,8 \%$ . Η συγκεκριμένη κατηγορία προβλέπεται αρκετά ικανοποιητικά.

(c=good): Από τις 69 καταχωρίσεις κατηγοριοποιεί λάθος τις 11 (Στην περίπτωση που 10 good κατηγοριοποιούνται ως acc και 1 good ως vgood). Ποσοστό πρόβλεψης της κατηγορίας good:  $58/69 * 100 = 84,1\%$ . Η συγκεκριμένη κατηγορία δεν προβλέπεται αρκετά ικανοποιητικά από το συγκεκριμένο μοντέλο.

(d=vgood): Από τις 65 καταχωρίσεις κατηγοριοποιεί λάθος τις 10 (Στην περίπτωση που 8 vgood κατηγοριοποιούνται ως acc και 2 vgood κατηγοριοποιούνται ως good). Ποσοστό πρόβλεψης της κατηγορίας vgood:  $55/65 * 100 = 84,62\%$ . Η συγκεκριμένη κατηγορία δεν προβλέπεται αρκετά ικανοποιητικά από το συγκεκριμένο μοντέλο.

Ο πίνακας σύγχυσης για k=7 του IBk:

=== Confusion Matrix ===

a	b	c	d	<-- classified as
1144	66	0	0	a = unacc
21	358	1	4	b = acc
1	39	29	0	c = good
0	18	0	47	d = vgood

(a=unacc): Από τις 1210 καταχωρίσεις κατηγοριοποιεί τις 66 λάθος (Στην περίπτωση που 66 unacc κατηγοριοποιούνται ως acc). Ποσοστό πρόβλεψης της κατηγορίας unacc είναι  $(1144/1210 * 100) = 94,5 \%$ . Προβλέπεται αρκετά ικανοποιητικά η συγκεκριμένη κατηγορία.

(b=acc): Από τις 384 καταχωρίσεις κατηγοριοποιεί τις 26 λάθος (Στη περίπτωση που 21 acc κατηγοριοποιούνται ως unacc, 1 acc κατηγοριοποιείται ως good και 4 acc κατηγοριοποιούνται ως vgood). Ποσοστό πρόβλεψης της κατηγορίας acc:  $358/384 * 100 = 93,23 \%$ . Η συγκεκριμένη κατηγορία προβλέπεται αρκετά ικανοποιητικά.

(c=good): Από τις 69 καταχωρίσεις κατηγοριοποιεί λάθος τις 40 (Στην περίπτωση που 1 good κατηγοριοποιείται ως unacc, 39 good ως acc). Ποσοστό πρόβλεψης της κατηγορίας good:  $29/69 * 100 = 42,03 \%$ . Η συγκεκριμένη κατηγορία δεν προβλέπεται καθόλου ικανοποιητικά στο συγκεκριμένο μοντέλο.

(d=vgood): Από τις 65 καταχωρίσεις κατηγοριοποιεί λάθος τις 18 (Στην περίπτωση που 18 vgood κατηγοριοποιούνται ως acc). Ποσοστό πρόβλεψης της κατηγορίας vgood:  $47/65 * 100 = 72,31 \%$ . Η συγκεκριμένη κατηγορία δεν προβλέπεται αρκετά ικανοποιητικά από το συγκεκριμένο μοντέλο.

Αυτό που παρατηρούμε είναι ότι ο J48 έχει καλύτερα ποσοστά πρόβλεψης των κατηγοριών από ότι ο kNN βέβαια αυτό ήταν αναμενόμενο διότι ξέραμε εξ αρχής πως ο J48 πετυχαίνει καλύτερη ακρίβεια στο συγκεκριμένο μοντέλο.

Καταγραφή κανόνων για κάθε κατηγορία:

unacc:

if person <= 2 then class = unacc

acc:

if person > 2 and safety > 1 and buying > 24.47 and maint <= 3.76 and lug\_boot > 343 and maint <= 3.14 and safety > 2 then class = acc

good:

if person > 2 and safety > 1 and buying <= 24.47 and maint <= 2.78 and safety <= 2 and lug\_boot > 363 and buying <= 12.5 then class = good

vgood:

if person > 2 and safety > 1 and buying <= 24.47 and maint <= 2.78 and safety > 2 and lug\_boot > 278 and doors > 3 then class = vgood