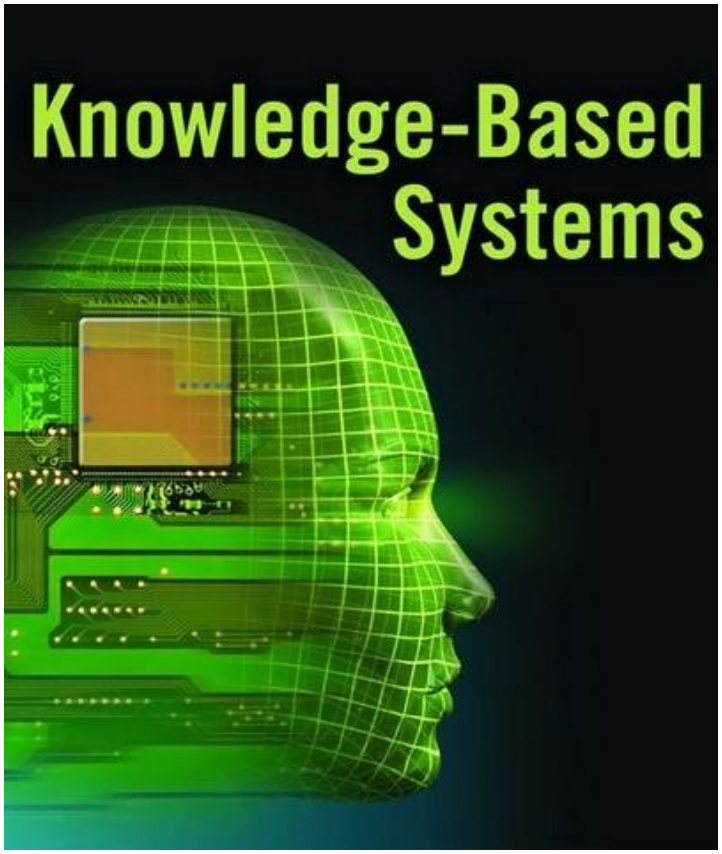
Knowledge Based Systems
ITCS 6155

Report on Classification of ‘Saatchi Art’ Paintings.



Joyta Choudhury – 800966655
Pooja Mounika Poluru – 800969916
Venkata Sai Siva Pruthvi. Ustepalle – 800957126
Vikas Dayananda – 800969865
Yousuf Sadiq Mohammed – 800959514

Project Description

The main goal of this project is to classify paintings based on price with high precision. The dataset used is the Paintings Dataset which we were supposed to extract from Saatchi art website. We are going to provide a solution to price painting based on various features of the painting.
We applied classification algorithms, J48, Naive Bayes, and Random Forest. In the first step, we extracted the dataset from the Saatchi art website, uploaded the dataset into Weka and discretized the dataset. On the resulting dataset, we applied J48, Naive Bayes, SOM algorithms and set price as the decision factor.

Data Source

Saatchi Art is the world's leading online art gallery, connecting people with art and artists they love. Saatchi Art, one of the most exciting places to discover and invest in emerging fine art artists. It offers an unparalleled selection of paintings, drawings, sculpture, and photography in a range of prices, and it provides artists from around the world with an expertly curated environment in which to exhibit and sell their work.

Based in Los Angeles, Saatchi Art is redefining the experience of buying and selling art by making it easy, convenient and welcoming for both collectors and artists.

Tools Used

Import.io and Python – To scrape data from website
R – To preprocess the data
WEKA – To classify the data

Web Scraping:

Fetching data from the website is the prior step for this project. Used Import.io to extract the visible attributes on web page (Painting title, Artist Name, # of views etc.,) of each item. For other attributes, we have designed a Python scrapper and parser using urllib and Beautifulsoup.

Addition of five new attributes namely

- Followers for each artist
- # of artworks of each artist
- Location of artist
- Painting brightness
- Education of the artist

Initial flow of the code is to fetch 20 pages of website containing paintings display to extract URLs specific to each item. This resulted in 400 items for 20 pages. Parser is built to crawl into html page of each item URL contained in a file input passed to it. This resulted in printing of each attribute required which is then copied to excel file to make it WEKA readable(.csv).

Four among five new attributes are scrapped from each artist profile page from the website.

Artist related attributes are chosen as they play a primary role to create a branding for an item. The number of followers determines popularity of artist which leads to increase in his popularity and price of the painting. **Artist Education** is chosen as it increases the value of painting, the more the education the expectation on quality of the painting is higher.

Image Brightness is chosen as it is the naïve attribute for a layman that captures his vision to buy a painting. It is calculated by a formula based on values of Red Green Blue pixels in each painting. This relies on using Image and ImageStat packages in Python. Though Image Brightness attribute is not perfectly alright in displaying the exact value but it portrays Perceived Brightness of each painting and since a same method is used to calculate the value it works relatively on all the images.

Entire Python code is attached as Scrapper.ipynb in the project package.

Preprocessing

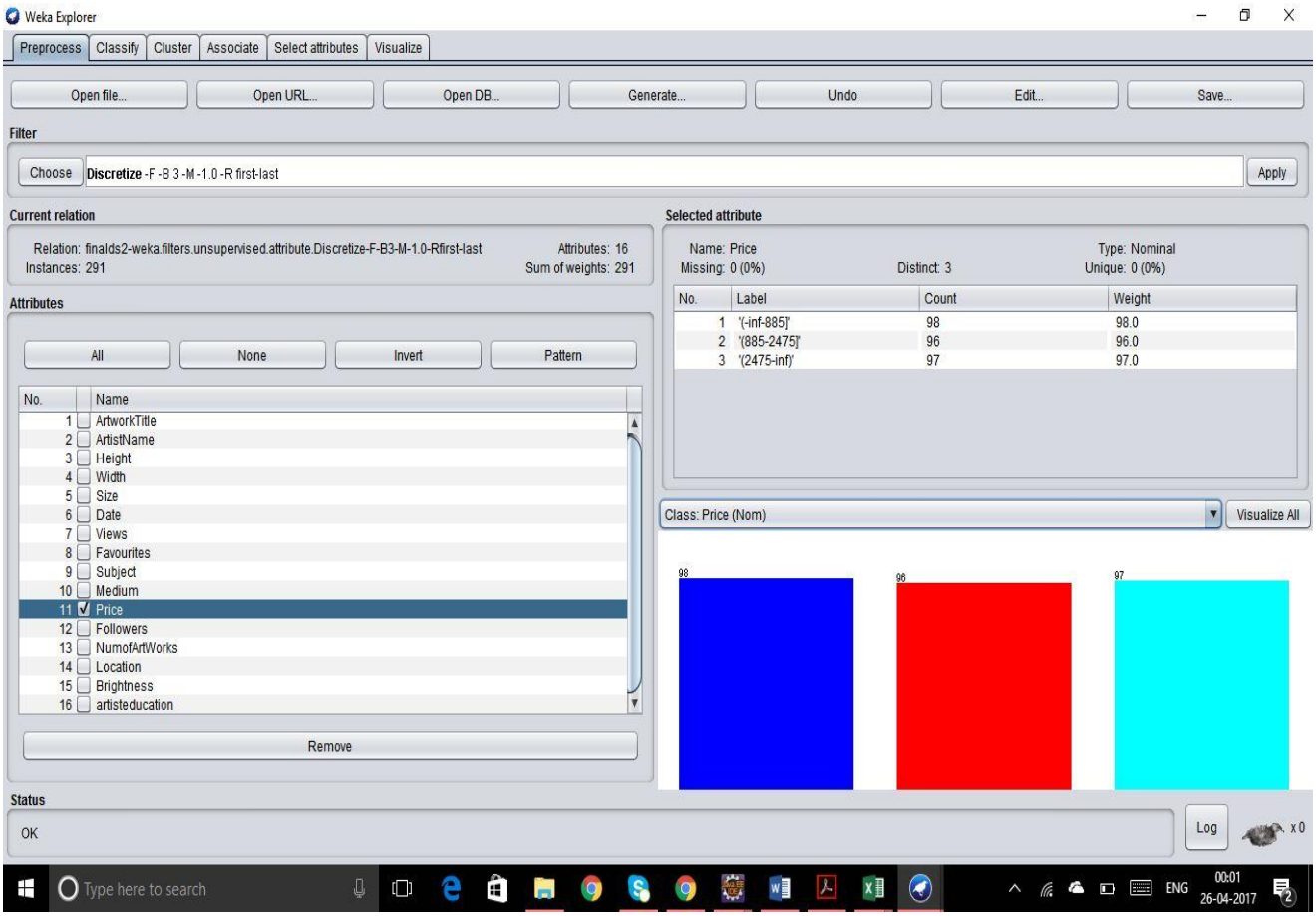
Many of the features extracted form website needed some preprocessing. Like,

- Converting the 13 digit ubuntu timestamp to readable date format.
- Removing punctuations and unnecessary characters from Price, Dimensions etc.,

All this tidying is performed by a R script and python code in the Scrapper.ipynb. Attached a R script DS_Preprocessing.R in the project package.After the preprocessing of the data we came down to 292 paintings that is a total of 292 rows in the data file.

Filter

Upload the processed data file and apply the Discretize filter and in the Discretize option set the **useEqualFrequency** to True so that we can get equal weight histograms. Price is taken as the decision feature and split it into 3 intervals.

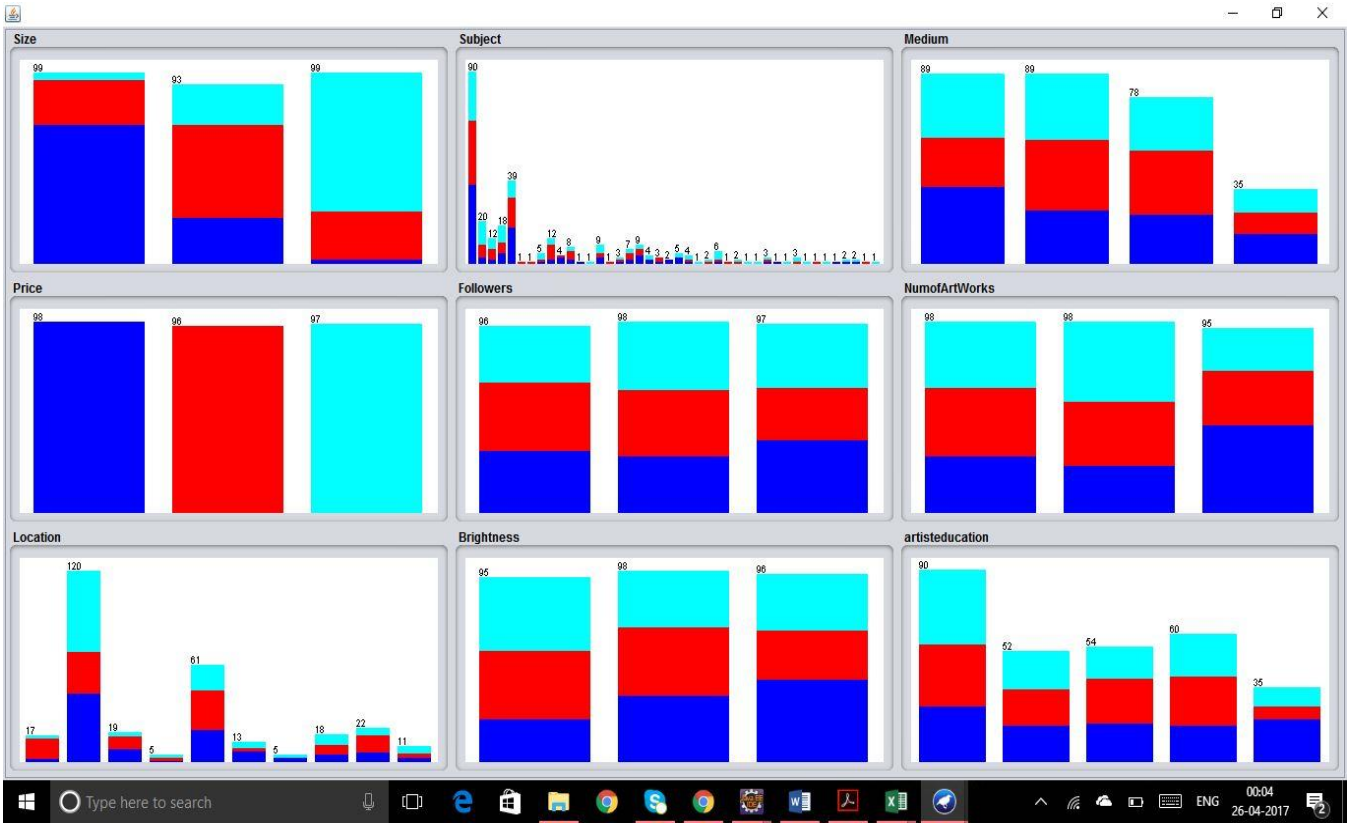


Discretize:

In the pic above we can see that the price is taken as the decision attribute and divided into just three intervals.

Next, we have removed the unwanted attributes like date, name of the artist, name of the art, and date which would not be of much help in the classification.

When we visualize the rest of the attributes it would look something like the picture below.

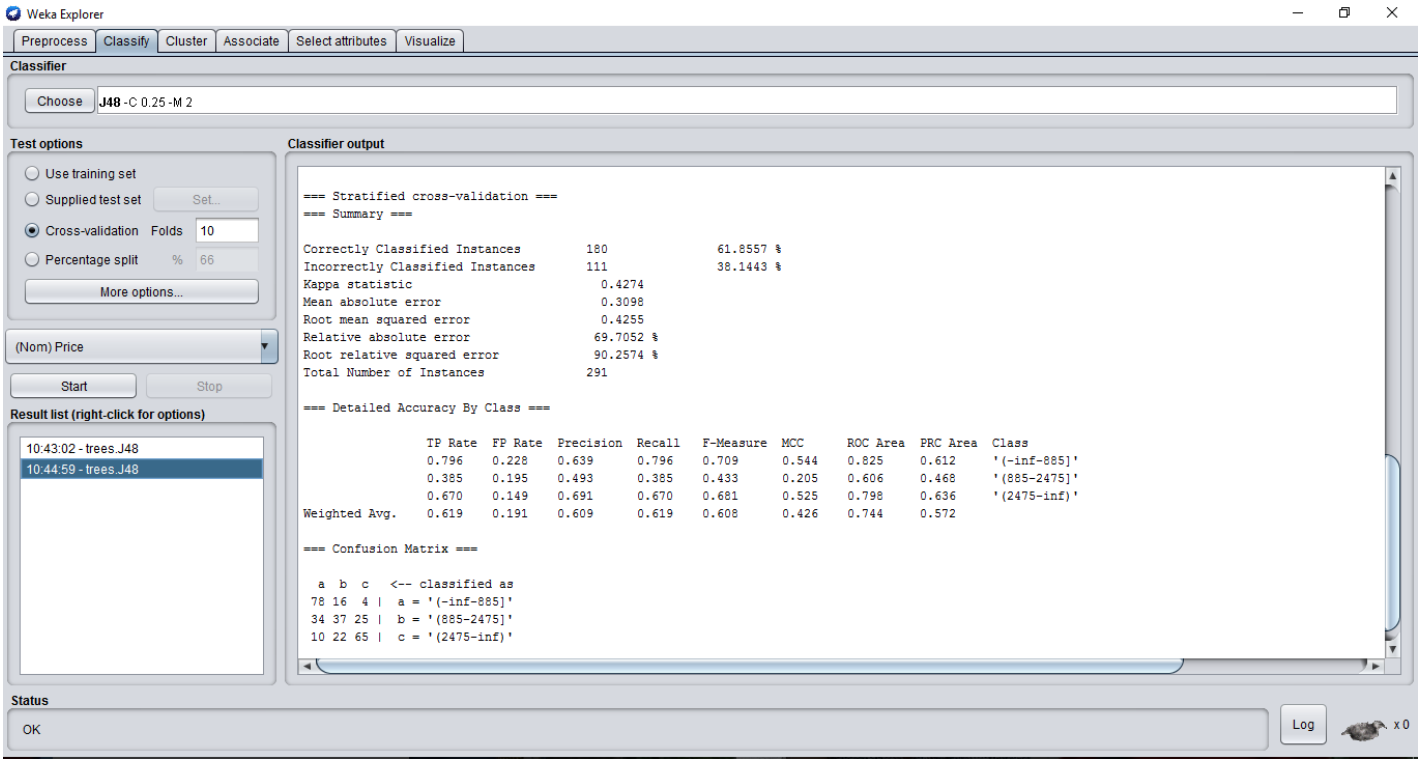


Manual discretization:

After initial set of discretization and running few algorithms like J48, Naïve Bayes we found that the price distribution '(-inf-885]', '(885-2475]', '(2475-inf)' was accurate. We tried verifying this using online art websites like artweb, and ugallery. This helped us in understanding the discretization better and cross checking our values.

Applying J48 algorithm and setting price as decision parameter:

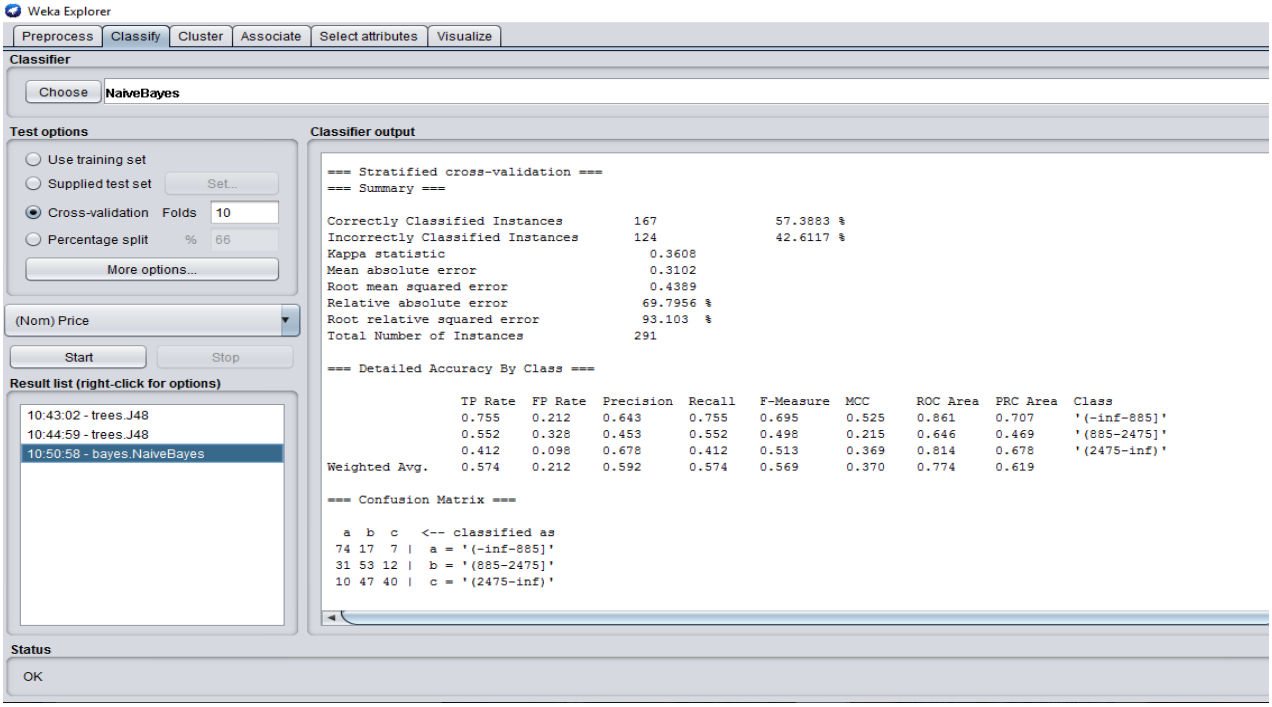
The first algorithm we implemented on the dataset is the J48 and observed the correctly and incorrectly classified instances.



From the above figure, we can see that the price is the decision parameter and the Correctly Classified Instances are 180 out of 291 that is 61.85%.

Applying Naïve Bayes Algorithm and setting price as the decision parameter:

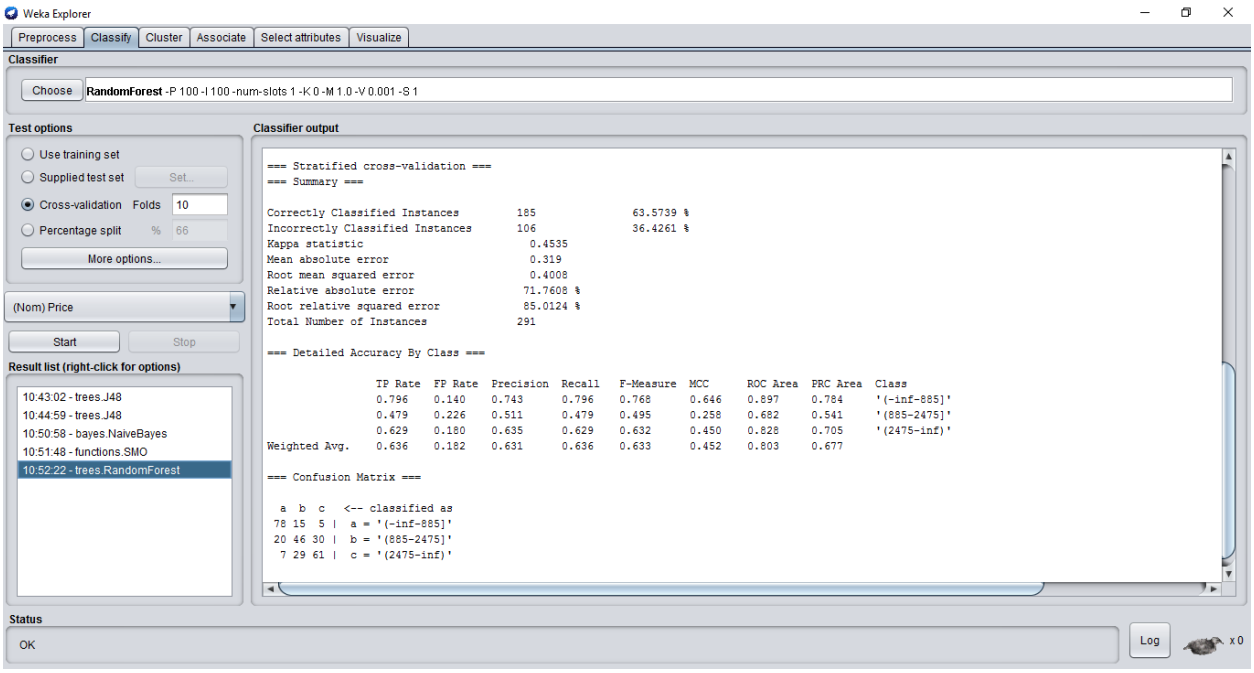
The second algorithm we implemented on the dataset is the Naïve Bayes and observed the correctly and incorrectly classified instances.



From the above figure, we can see that the price is the decision parameter and the Correctly Classified Instances are 167 out of 291 that is 57.38% of accuracy.

Applying Random Forest Algorithm and setting price as the decision parameter:

The final algorithm we implemented on the dataset is the Random Forest and observed the correctly and incorrectly classified instances.



From the above figure, we can see that the price is the decision parameter and the Correctly Classified Instances are 185 out of 291 that is 63.57% which makes it the most precise algorithm out of the three we have used.

Decision Tree:

The decision tree is obtained by running the J48 algorithm setting price, medium, artist education and followers as the main attributes. This would help us arrive draw a conclusion of the highly-priced paintings depending on both the artist education and the medium of the art.

From the above decision tree, we can infer that if the size of the painting is less than 1408, medium is Acrylic and size is less than 784 then the possible price of the painting would be in the range (2475 – inf).

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.796	0.140	0.743	0.796	0.768	0.897	'(-inf-885]'
0.479	0.226	0.511	0.479	0.495	0.682	'(885-2475]'
0.629	0.180	0.635	0.629	0.632	0.828	'(2475-inf]'

Confusion Matrix:

a	b	c
78	15	5
20	46	30
7	29	61

a = '(-inf-885]'
b = '(885-2475]'
c = '(2475-inf)'

From the above confusion matrix, the sum of row elements give the instances of each class i.e. the number of actual instances in class ‘a’ are 78+20+7 = 105.
And the sum of the column elements give the predicted instances of each class.
So out of 105 instances in class ‘a’, 78 instances are correctly classified. These are the True Positives.
And the remaining are wrongly classified (20 as class ‘b’, 7 as class ‘c’).

The number of actual instances in class ‘b’ are 15+46+29 = 90.
And the sum of the column elements give the predicted instances of each class.
So out of 90 instances in class ‘b’, 46 instances are correctly classified. These are the True Positives.
And the remaining are wrongly classified (15 as class ‘a’, 29 as class ‘c’).

The number of actual instances in class ‘c’ are 5+30+61 = 96.
And the sum of the column elements give the predicted instances of each class.
So out of 96 instances in class ‘c’, 61 instances are correctly classified. These are the True Positives.
And the remaining are wrongly classified (30 as class ‘b’, 5 as class ‘a’).

The sum of all the diagonal elements give the true predictions and the non-diagonal are the false predictions.
True prediction=78+46+61 = 185

Accuracy Percentage = True prediction/ (True + False Predictions);

= (185 / (185+106)) * 100 = 63.57%.