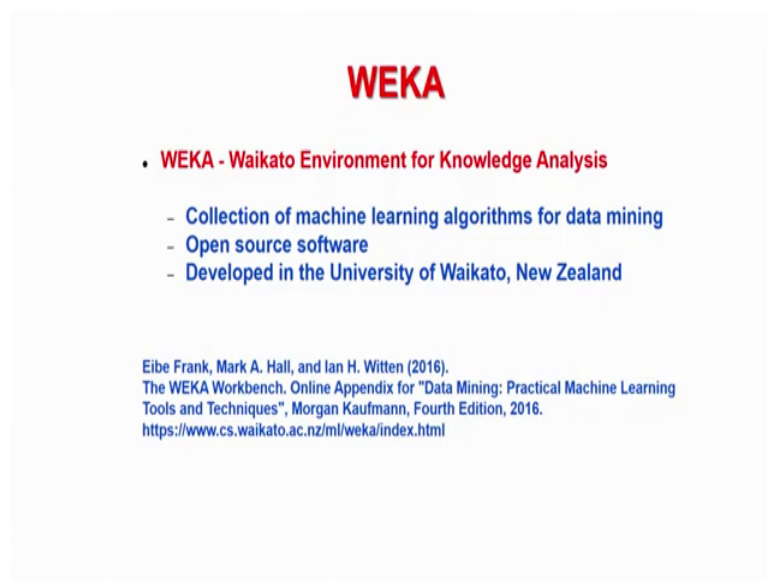


Bioinformatics
Prof. M. Michael Gromiha
Department of Biotechnology
Indian Institute of Technology, Madras

Lecture – 69
Demonstration on WEKA

Demonstration on WEKA, WEKA stands for Waikato Environment for Knowledge Analysis.

(Refer Slide Time: 00:29)



WEKA

- **WEKA - Waikato Environment for Knowledge Analysis**
 - Collection of machine learning algorithms for data mining
 - Open source software
 - Developed in the University of Waikato, New Zealand

Eibe Frank, Mark A. Hall, and Ian H. Witten (2016).
The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning
Tools and Techniques", Morgan Kaufmann, Fourth Edition, 2016.
<https://www.cs.waikato.ac.nz/ml/weka/index.html>

And it is a collection of machine learning algorithms for data mining, which is open source software, developed the university of Waikato New Zealand. So, you can get more details about WEKA and different machine learning technique and, how to prepare the input files and, how to interpret the result this book write the written by Ian Witten and, we can also refer the website for more information.

(Refer Slide Time: 00:53)

Problem statement

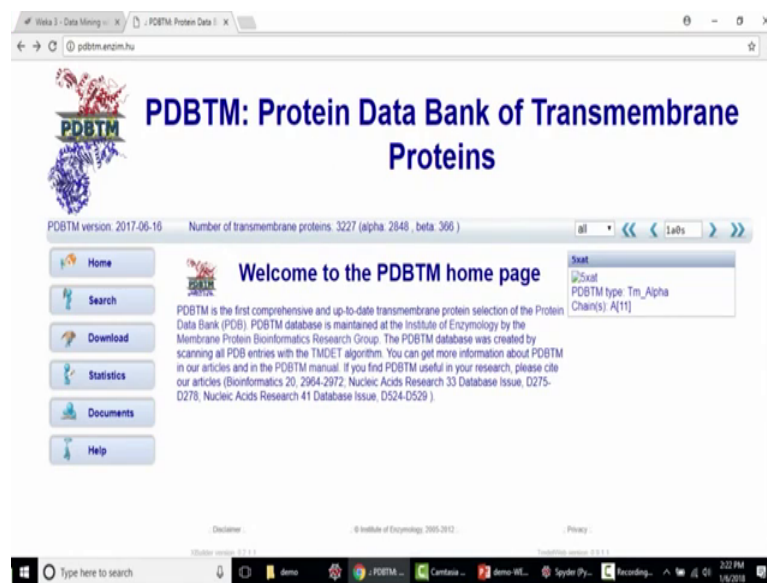
Develop a prediction model to discriminate α -helical and β -strand transmembrane proteins

- Steps to be followed:
 - Obtain transmembrane protein dataset (training and test dataset)
 - Extract features from dataset
 - Choose a machine learning algorithm and train the model
 - Perform validation and assess the model performance

So, in this demonstration we will explain how to develop a prediction model for example, to discriminate two different sets of proteins, in this case alpha helical and beta strand transmembrane proteins. So, you have to follow different steps for the discrimination, first we had to prepare dataset right in this case you have to get the sequences for the transmembrane proteins both alpha helical and beta strand proteins.

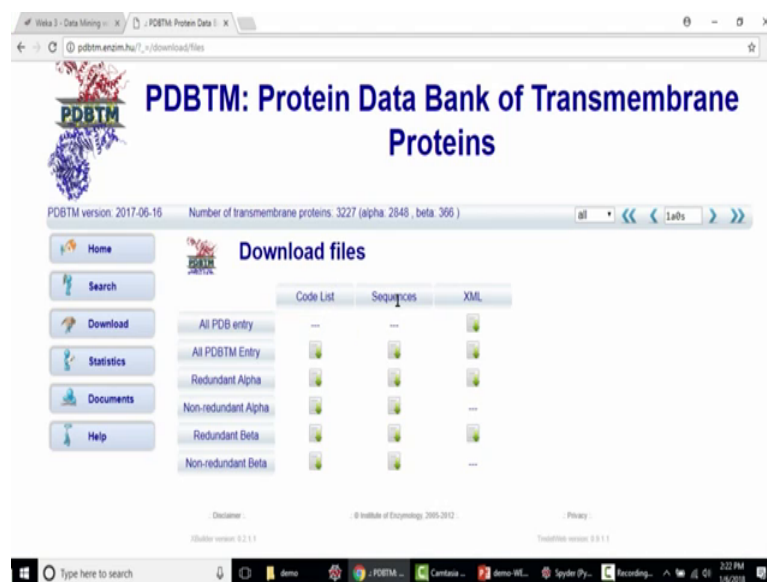
And you classified additional training and test dataset and, from this training dataset we need to extract different features, as we discussed in the class like the amino acid composition, or different properties. And we use a machine learning algorithm available in WEKA right, there are more than 50 algorithms are available and you train the model, then once you train the model you get the results and, we need to do the validation using the fiberglass validation, or the sample split and assess the performance. So, we will see the details how to create the dataset and how to get the features and how to validate any algorithm, with respect to a different assessment parameters.

(Refer Slide Time: 02:04).



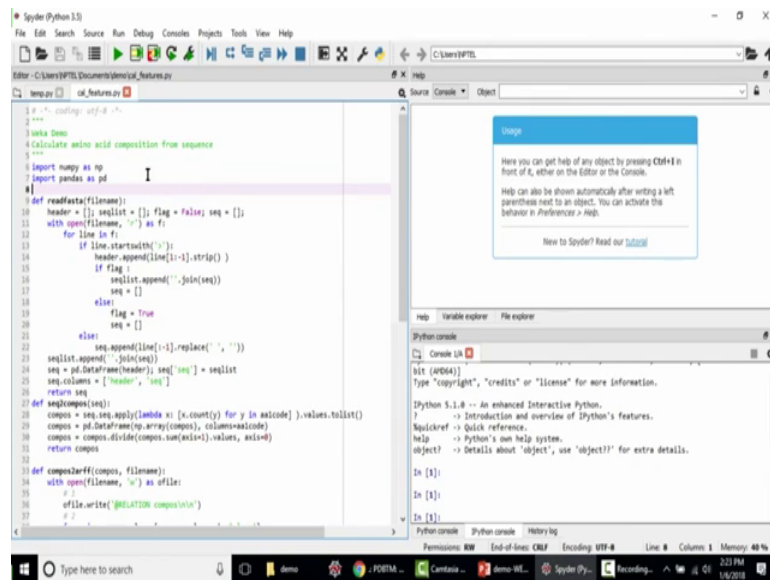
We will begin the exercise by downloading the transmembrane proteins from PDBTM go to download files.

(Refer Slide Time: 02:10)



The three categories code list sequence XML, we need their sequence to extract the features. So, under sequence download the non redundant alpha sequence and non redundant beta sequence, I would suggest you to download the redundant and, you see deed it to convert into non redundant sequence, in this demo I will save the non redundant alpha, or non redundant beta, sequence has my dataset.

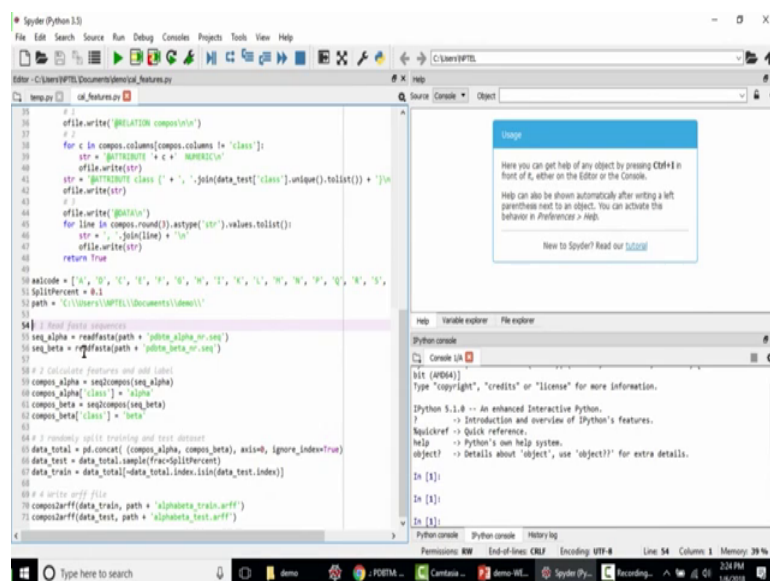
(Refer Slide Time: 03:00)



```
1 # -*- coding: utf-8 -*-
2
3 """
4 calculate amino acid composition from sequence
5 """
6 import numpy as np
7 import pandas as pd
8
9 def readfasta(filename):
10     header = []
11     seqlist = []
12     flag = False
13     seq = []
14     with open(filename, "r") as f:
15         for line in f:
16             if line.startswith(">"):
17                 header.append(line[1:].strip())
18             elif flag:
19                 seqlist.append("".join(seq))
20                 seq = []
21             else:
22                 flag = True
23                 seq = []
24             else:
25                 seq.append(line[1:].replace(" ", ""))
26                 seqlist.append("".join(seq))
27                 seq = []
28                 seq.columns = ["header", "seq"]
29                 return seq
30
31 def seq2comp(seq):
32     comp = seq.apply(lambda x: [x.count(y) for y in aacode]).values.tolist()
33     comp = pd.DataFrame(np.array(comp), columns=aacode)
34     comp = comp.divide(comp.sum(axis=1).values, axis=1)
35     return comp
36
37 def comp2arff(comp, filename):
38     with open(filename, "w") as ofile:
39         ofile.write("@RELATION comp\n")
40         for c in comp.columns:
41             ofile.write("@ATTRIBUTE %s TYPE %s\n" % (c, comp[c].dtype))
42         ofile.write("@DATA\n")
43         for line in comp.iterrows():
44             ofile.write("%s\n" % " ".join([str(v) for v in line[1]]))
45         ofile.write("@END\n")
46
47 aacode = ["A", "C", "G", "U", "V", "W", "M", "T", "Y", "K", "R", "N", "S", "D", "E", "Q", "H", "L", "P", "F", "I", "O", "X"]
48 splitPercent = 0.1
49 path = "C:\\Users\\NPTEL\\Documents\\ol_features.py"
50
51 # Read fasta sequences
52 seq_alpha = readfasta(path + "ol_data_alpha.fasta")
53 seq_beta = readfasta(path + "ol_data_beta.fasta")
54
55 # Calculate features and add label
56 comp_alpha = seq2comp(seq_alpha)
57 comp_beta = seq2comp(seq_beta)
58 comp_alpha['class'] = 'alpha'
59 comp_beta['class'] = 'beta'
60
61 # Randomly split training and test dataset
62 data_total = pd.concat([comp_alpha, comp_beta], axis=0, ignore_index=True)
63 data_test = data_total.sample(frac=splitPercent)
64 data_train = data_total[~data_total.index.isin(data_test.index)]
65
66 # Write arff file
67 comp2arff(data_train, path + "ol_data_train.arff")
68 comp2arff(data_test, path + "ol_data_test.arff")
```

To extract the features, we will use a short a simple program, will calculate composition as the features in this exercise, but you can calculate other features, which you think will be useful for the your problem, the program does three main tasks first it read the FASTA file that is sequence file, which we downloaded and convert and stored in the memory as sequence and, this sequence is converted two composition which is our feature and features are converted to ARFF file which is a input file for WEKA.

(Refer Slide Time: 03:50)

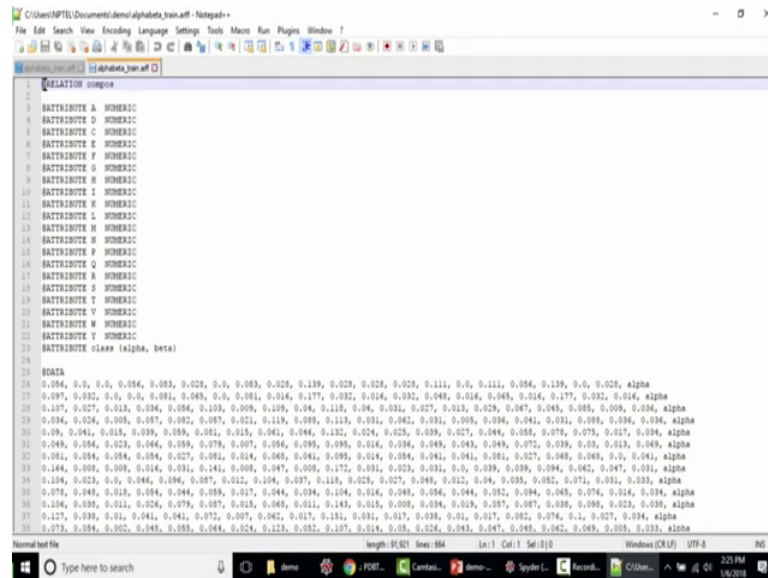


```
35 #
36 ofile.write("@RELATION comp\n")
37 #
38 for c in comp.columns:
39     ofile.write("@ATTRIBUTE %s TYPE %s\n" % (c, comp[c].dtype))
40 ofile.write("@DATA\n")
41 for line in comp.iterrows():
42     ofile.write("%s\n" % " ".join([str(v) for v in line[1]]))
43 ofile.write("@END\n")
44
45 aacode = ["A", "C", "G", "U", "V", "W", "M", "T", "Y", "K", "R", "N", "S", "D", "E", "Q", "H", "L", "P", "F", "I", "O", "X"]
46 splitPercent = 0.1
47 path = "C:\\Users\\NPTEL\\Documents\\ol_features.py"
48
49 # Read fasta sequences
50 seq_alpha = readfasta(path + "ol_data_alpha.fasta")
51 seq_beta = readfasta(path + "ol_data_beta.fasta")
52
53 # Calculate features and add label
54 comp_alpha = seq2comp(seq_alpha)
55 comp_beta = seq2comp(seq_beta)
56 comp_alpha['class'] = 'alpha'
57 comp_beta['class'] = 'beta'
58
59 # Randomly split training and test dataset
60 data_total = pd.concat([comp_alpha, comp_beta], axis=0, ignore_index=True)
61 data_test = data_total.sample(frac=splitPercent)
62 data_train = data_total[~data_total.index.isin(data_test.index)]
63
64 # Write arff file
65 comp2arff(data_train, path + "ol_data_train.arff")
66 comp2arff(data_test, path + "ol_data_test.arff")
```

So, the code is written as follows read the FASTA sequence both alpha beta calculate the features and, I am adding the label as alpha and beta to the dataset, then I am randomly splitting the dataset into training and test, 90 percent of the data will be used as training. And the remaining 10 percentage will be used as test dataset, further I am writing the split dataset training and test to a input file ARFF file, in the WEKA prescribed ARFF format.

[illegible]

(Refer Slide Time: 05:00)



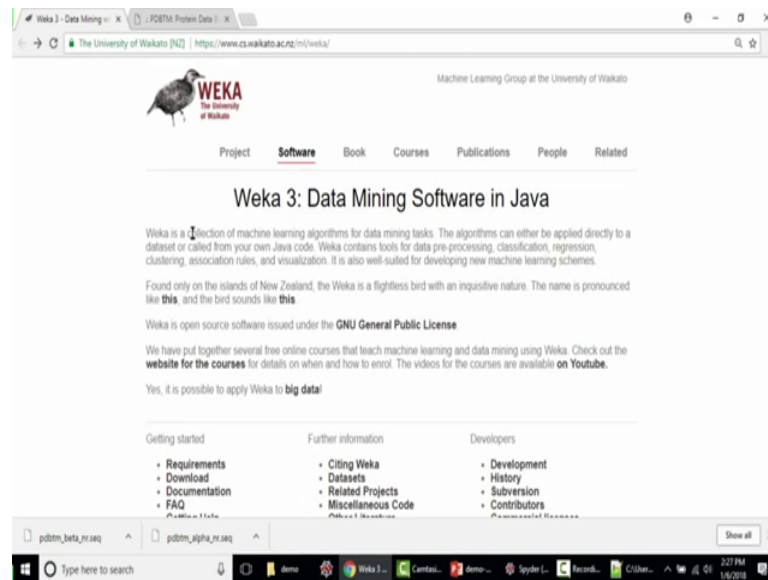
```
1 RELATION composition
2
3 ATTRIBUTE A NUMERIC
4 ATTRIBUTE C NUMERIC
5 ATTRIBUTE D NUMERIC
6 ATTRIBUTE E NUMERIC
7 ATTRIBUTE F NUMERIC
8 ATTRIBUTE G NUMERIC
9 ATTRIBUTE H NUMERIC
10 ATTRIBUTE I NUMERIC
11 ATTRIBUTE K NUMERIC
12 ATTRIBUTE L NUMERIC
13 ATTRIBUTE M NUMERIC
14 ATTRIBUTE N NUMERIC
15 ATTRIBUTE P NUMERIC
16 ATTRIBUTE Q NUMERIC
17 ATTRIBUTE R NUMERIC
18 ATTRIBUTE S NUMERIC
19 ATTRIBUTE T NUMERIC
20 ATTRIBUTE V NUMERIC
21 ATTRIBUTE W NUMERIC
22 ATTRIBUTE Y NUMERIC
23 ATTRIBUTE class {alpha, beta}
24
25 DATA
26 0.084, 0.0, 0.0, 0.084, 0.083, 0.028, 0.0, 0.083, 0.028, 0.139, 0.028, 0.028, 0.028, 0.111, 0.0, 0.111, 0.084, 0.139, 0.0, 0.028, alpha
27 0.097, 0.032, 0.0, 0.0, 0.081, 0.048, 0.0, 0.081, 0.014, 0.177, 0.032, 0.014, 0.032, 0.048, 0.014, 0.048, 0.014, 0.177, 0.032, 0.014, alpha
28 0.107, 0.027, 0.013, 0.034, 0.084, 0.103, 0.009, 0.109, 0.04, 0.118, 0.04, 0.031, 0.027, 0.013, 0.028, 0.047, 0.048, 0.088, 0.009, 0.034, alpha
29 0.034, 0.024, 0.008, 0.087, 0.082, 0.087, 0.021, 0.119, 0.088, 0.113, 0.031, 0.042, 0.031, 0.008, 0.034, 0.041, 0.031, 0.088, 0.034, 0.034, alpha
30 0.09, 0.041, 0.015, 0.039, 0.039, 0.031, 0.015, 0.041, 0.046, 0.112, 0.024, 0.025, 0.039, 0.027, 0.044, 0.038, 0.078, 0.078, 0.017, 0.034, alpha
31 0.049, 0.054, 0.023, 0.044, 0.059, 0.079, 0.007, 0.054, 0.095, 0.095, 0.014, 0.034, 0.049, 0.043, 0.049, 0.072, 0.039, 0.03, 0.013, 0.049, alpha
32 0.081, 0.054, 0.054, 0.054, 0.027, 0.081, 0.014, 0.048, 0.041, 0.095, 0.014, 0.034, 0.041, 0.041, 0.081, 0.027, 0.048, 0.048, 0.0, 0.041, alpha
33 0.144, 0.008, 0.008, 0.014, 0.031, 0.141, 0.008, 0.047, 0.008, 0.172, 0.031, 0.023, 0.031, 0.0, 0.039, 0.039, 0.094, 0.042, 0.047, 0.031, alpha
34 0.104, 0.023, 0.0, 0.044, 0.094, 0.087, 0.012, 0.104, 0.037, 0.119, 0.025, 0.027, 0.048, 0.012, 0.04, 0.038, 0.082, 0.071, 0.031, 0.033, alpha
35 0.078, 0.048, 0.018, 0.054, 0.044, 0.059, 0.017, 0.044, 0.034, 0.104, 0.014, 0.048, 0.054, 0.044, 0.082, 0.094, 0.048, 0.074, 0.014, 0.034, alpha
36 0.104, 0.038, 0.011, 0.024, 0.079, 0.087, 0.015, 0.048, 0.011, 0.143, 0.015, 0.008, 0.034, 0.019, 0.087, 0.087, 0.038, 0.098, 0.023, 0.038, alpha
37 0.117, 0.038, 0.01, 0.041, 0.041, 0.072, 0.007, 0.042, 0.017, 0.181, 0.031, 0.017, 0.038, 0.01, 0.017, 0.082, 0.074, 0.1, 0.027, 0.034, alpha
38 0.172, 0.054, 0.002, 0.048, 0.088, 0.044, 0.024, 0.113, 0.032, 0.107, 0.014, 0.05, 0.024, 0.043, 0.047, 0.048, 0.042, 0.048, 0.008, 0.031, alpha
```

Let us look into the file this is the ARFF format, as the first line starts with a relation which gives a some unique name. So, that we can refer the file later I am written composition, then there are attributes, attributes of the features or properties.

So, alanine composition which is soft type numeric, attribute can be of two type, one can be numeric the other can be nominal categorical variable for example, the last attribute which is a class is a nominal type, we have two categories alpha comma beta, which is given in the curly bracket followed by the list of attribute, we have a section called data a data, where each features are given for each data point in a (Refer Time: 05:57) separated format.

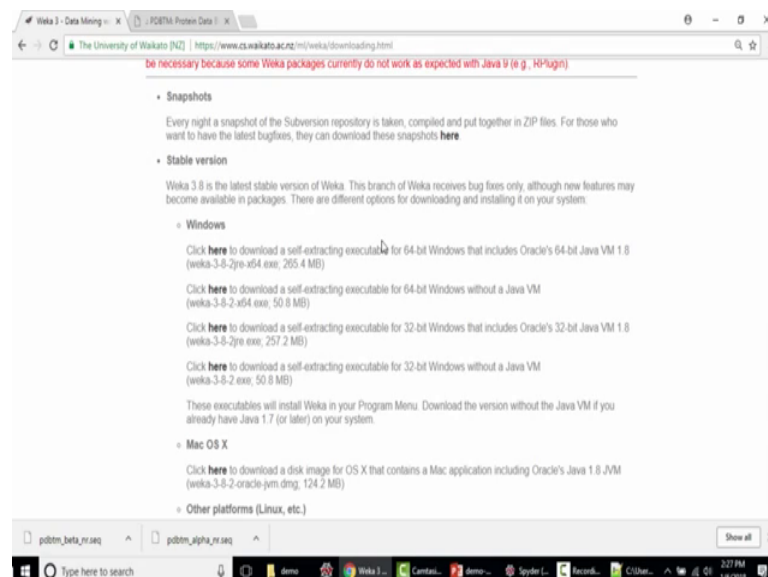
So, I have written the all are alpha that is alpha helical transmembrane protein and, calculate the composition and I listed line by line followed by the beta transmembrane proteins. So, my entire dataset for training is listed in a single file, where each row correspond to each protein. So, this is a format for WEKA. Now, that we have created the input WEKA let us move on to WEKA.

(Refer Slide Time: 06:43)



So, WEKA it is a collection of machine learning algorithm. So, it forms a wrapper a kind of a package. So, which facilitate us to access the algorithms easily. So, you can download the WEKA from WEKA website, WEKA supports three operating systems the windows Mac and Linux.

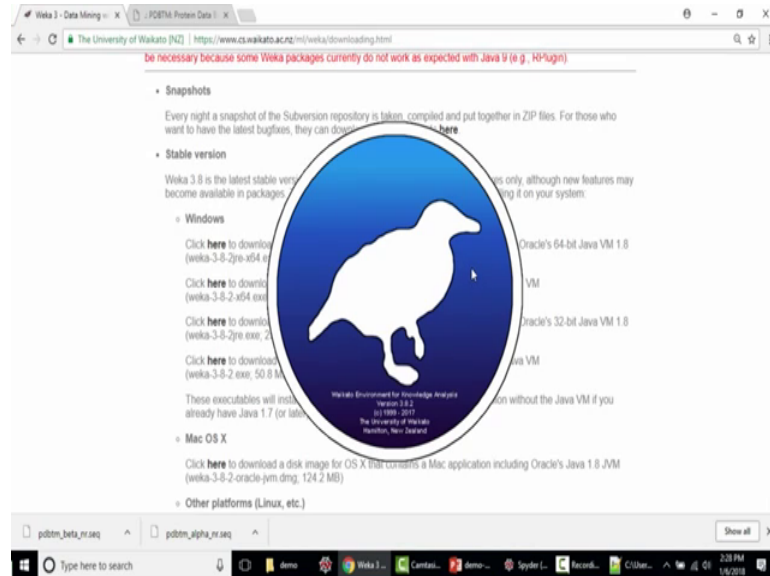
(Refer Slide Time: 07:10)



And, since WEKA is written in java you need a working java environment to use WEKA, you can install java by downloading the java VM are WEKA also comes with JRE, which you can download and directly install, it in case you face any problem using

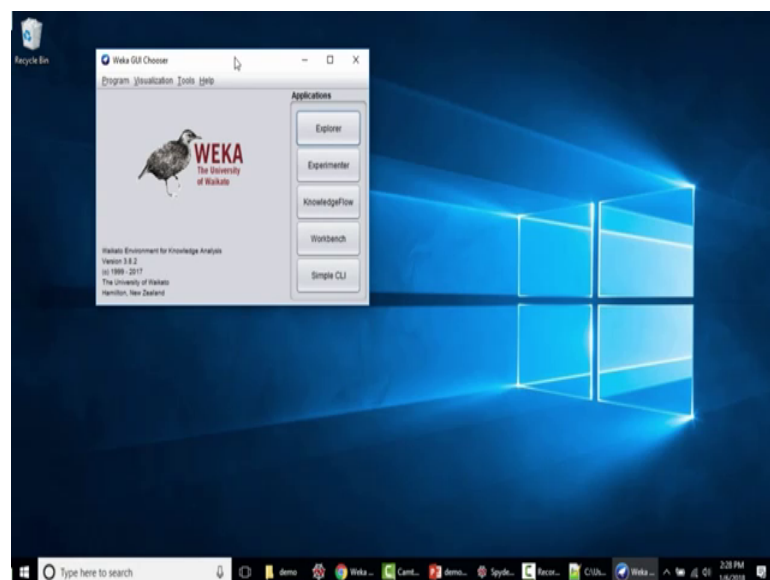
WEKA check your java and were not first, I have already used installed java in my system and WEKA, let me launch WEKA.

(Refer Slide Time: 07:44)



So, WEKA starts which the logo is a WEKA bird. So, as you see there is a small window which is a WEKA GUI.

(Refer Slide Time: 08:00)

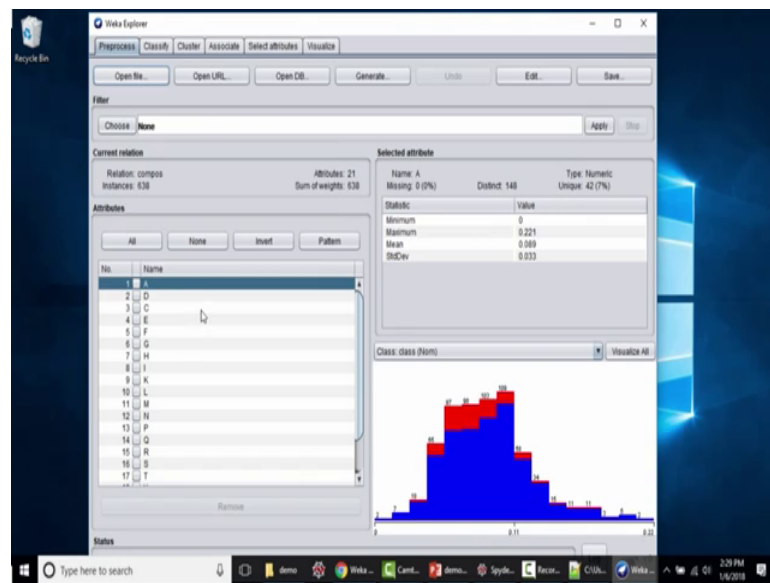


There is a menu bar which has few options to visualize tools and other things, there are five main buttons on the main window, that that is essential that the functionalities of the WEKA. The first one is the explorer which we will be discussing in this tutorial. The

next one experimenter experimenter is used to compare multiple, ah classifiers, algorithms with a different dataset for your work and knowledge flow gives a graphical interface to do this, workbench is a composite of all this and simple command line interface is a GULS WEKA access point.

So, you can access all the WEKA's functionality in the command line here. So, let us begin let us go to the explorer. So, the preprocess tab click on open file and, open the training input file the tab is now populated with the data which from the file.

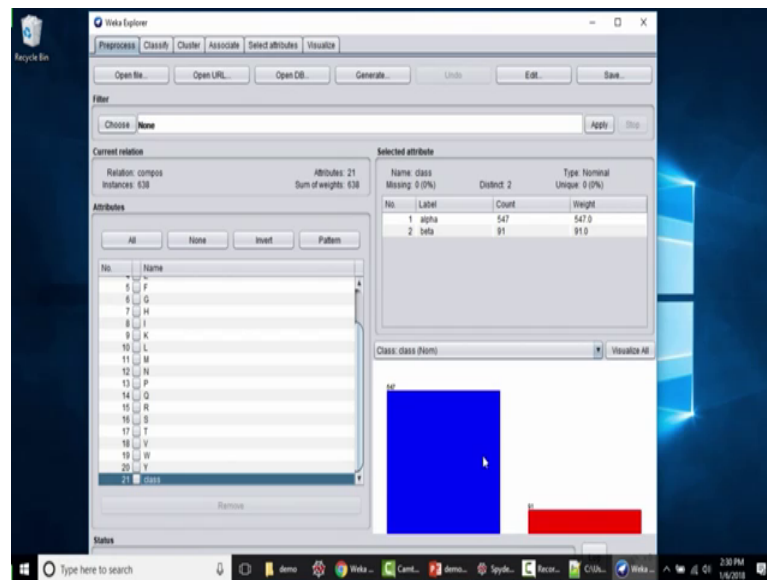
(Refer Slide Time: 09:16)



As you could see there is a filter section and, there is a attribute section and, there is a statistical section here, which displays the attribute information here and there is a graph.

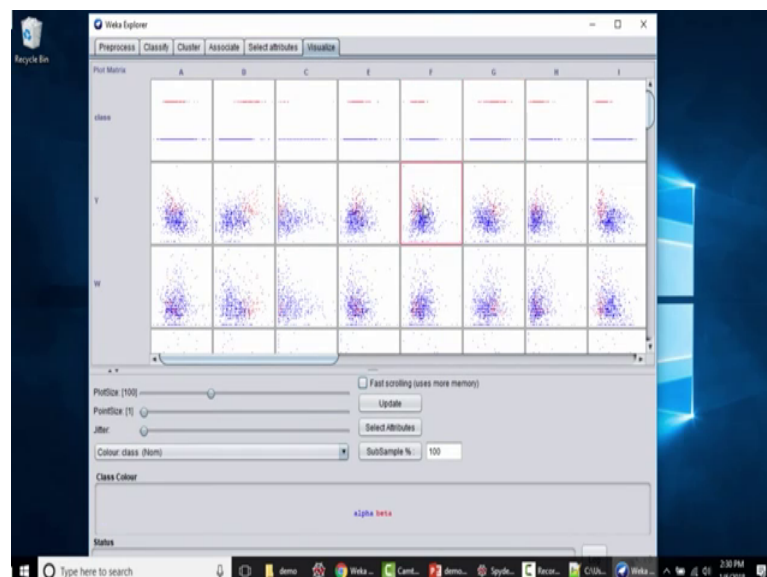
So, each attribute can be selected by checking here the checkbox, if you click on the class, you could see that blue, the sent for alpha and let for beta.

(Refer Slide Time: 09:38)



The first class is given a blue color, or the count number of proteins in each class is 547 and 91, which is displayed here and the corresponding weight. So, similarly you can get statistics of all other variable here, you want to look into much more detail the relation between the attributes you can go to visualize.

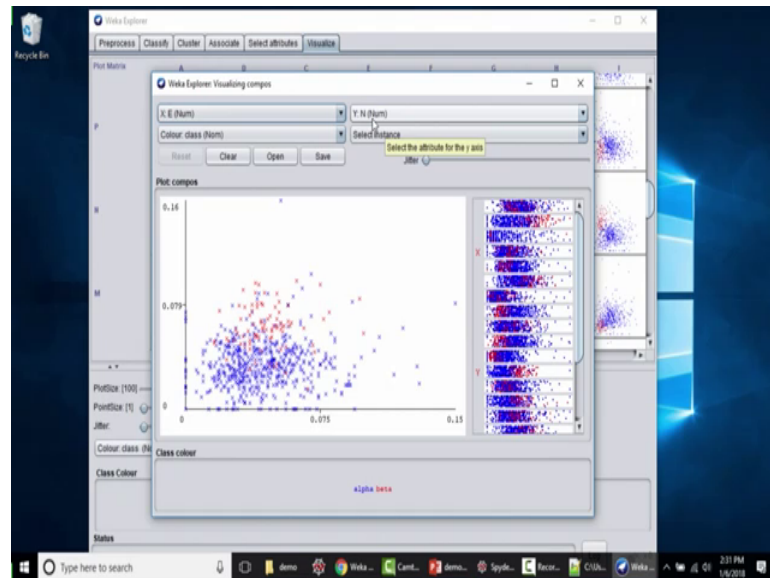
(Refer Slide Time: 10:08)



As you could see that each feature, or the attribute is listed here in X and Y axis and, the corresponding relationships are showed here this is very useful in case you are analyzing

your data for example, if I click on this plot this a plot between X that is the glutamic acid and Y that is the aspergine.

(Refer Slide Time: 10:35)



So, you could see the relationship as he could see the Y axis differentiate the red and blue, which is alpha and beta. So, it could be a potential good feature, such analysis can be done here for more complicated problems, you can choose and analyze the data. Let us go on to preprocess and the most important part of the preprocess tab is a filter.

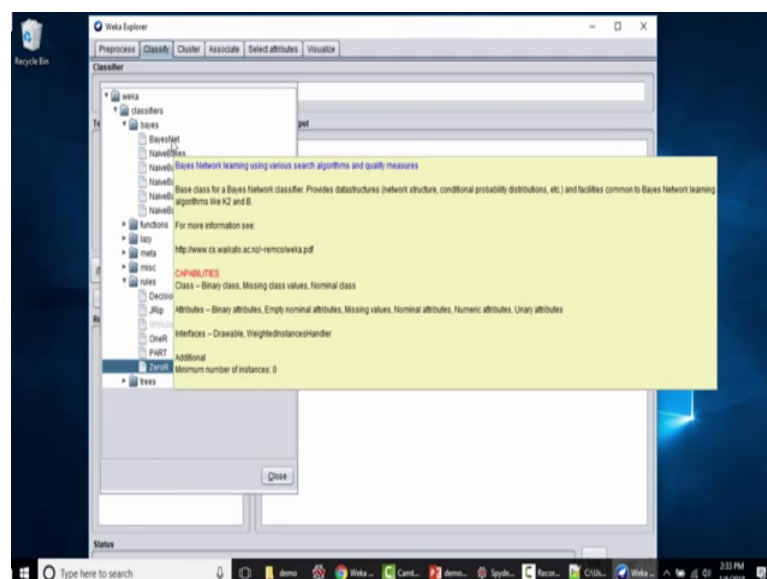
You can add filters here a lot of different the filters are available in WEKA, feel free to explore it and, there is a supervised category and unsupervised category, in supervised category there is another two categories attribute and instance. This filters for the attribute will modify the attributes, listed in the attributes will list whereas, instance will vary or modify the actual input data.

(Refer Slide Time: 11:51)

[illegible]

So, if you want to edit your data, you can directly go to edit and change modify and can save this data, but which should be avoided because this will modify original file itself. Once you are done modifying, or once you are done loading the data go to classify, here we will actually train the model, in this tab. So, there is a three different section here, as you could see classifier text option, which is a validation part and there is a classifier output where you would see the output.

(Refer Slide Time: 12:30)



So, Zero R is a by default is a classifier which is displayed here, why Zero R will come back to that later, if you click on choose you will see a list of classifiers for example, base under this folder you will see many more classifier these are the actual algorithms, which is rapidly on WEKA you can use base to train a model.

Similar arguments are grouped under single folder, for the base base to a algorithms are displayed in base folder and, tree based algorithms are displayed in trees folder and, decision rules based algorithms are displayed in rules folder. And there is a separate category called meta. Meta algorithm or kind of classifier which depends on other classifier they are modified classifier meta classifiers.

So, they are listed under separate category and the meta. So, you can learn more about this each classifier by selecting them, you will see the Bayes net is selected here and, the parameters are listed here you can click on that you can see a tab opened, you can modify the parameters here, if you want to understand what this parameter mean you can click on more look at the details.

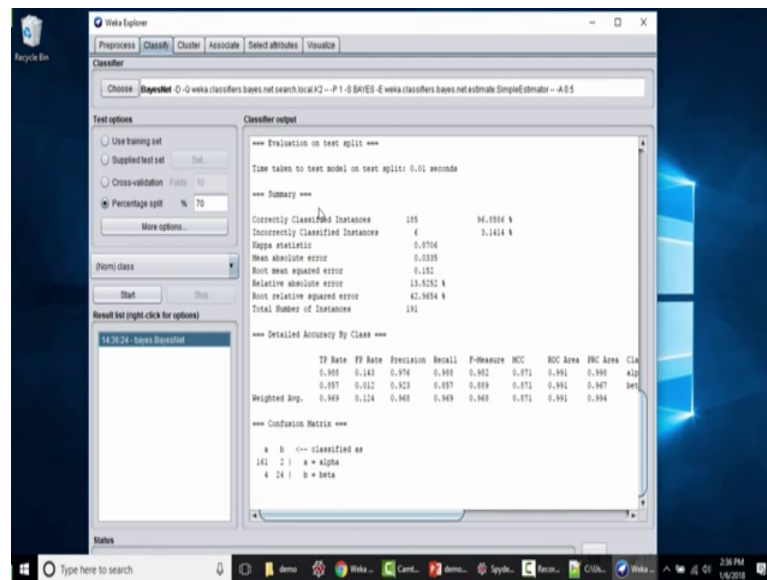
Each parameter and the method is displayed along with an difference, make sure you understand what the method is before you are using it. And you can see the capabilities what they are designed to do for example, this is a binary classifier it does not do regression does the only classification.

So, once you are chosen your classifier other algorithm, second you choose the validation method, there is a four different validation method listed here, use training set, where you could train your model using the training dataset and test its performance on the training set. Which may not be useful as a validation procedure and the second is a supplied test set, where you can be feed a blind test set, or independent test set as a for validation.

And third is a cross validation, which I will explain you and fourth one is a percentage split, in percentage split for example, 70 I will old 70 percentage of the data and data for training and remaining thirty percent will be used for testing or the validation ok. Let me run the program by default WEKA chooses a last attribute as the class.

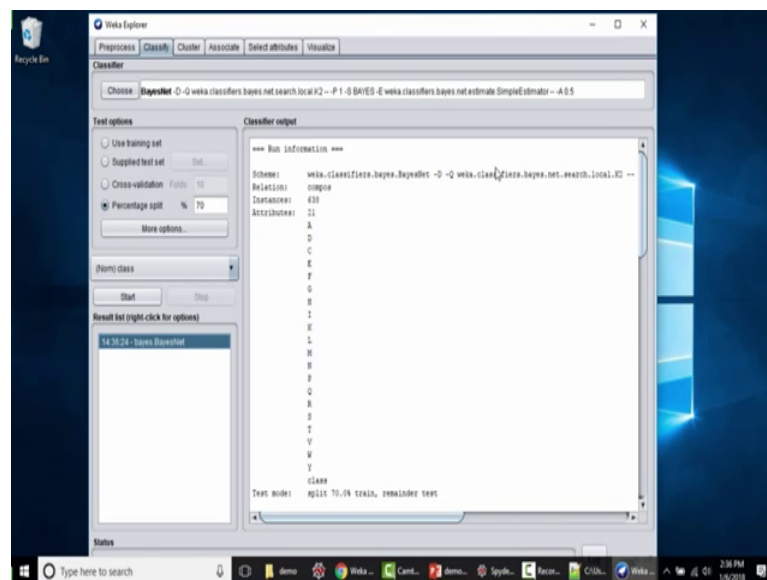
So, in case if your class variable is not the last variable you have to change it manually select the class and click on start.

(Refer Slide Time: 15:40)



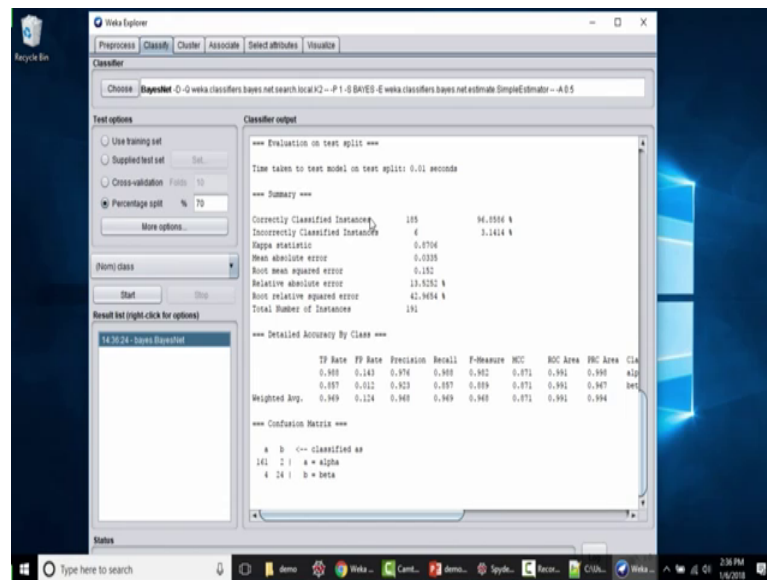
Now, the model building is done the WEKA trained your model and, the results are listed here.

(Refer Slide Time: 15:47)



Each class fare will have a different algorithm. So, they will display different information here, but the summary will be the same, which is composed by the WEKA interface.

(Refer Slide Time: 12:55)



As you could see the correctly classified instance is 185, which is a thirty percent of the total data, which we used as a test set and the percentage of the correctly classified is 96.85 which is nothing, but accuracy and incorrectly classified a 6. Similarly you can see the other details such as the TP rate, which is sensitivity and FP rate precision recall MCC ROC other details are listed here, for both alpha and beta. If you consider alpha is positive TP rate of alpha will be a sensitive and TP rate of beta will be the sensitivity and, the total weighted average will also again be the accuracy.

And below you could see the confusion matrix, where it should be read as alpha classified as alpha, a classified as a classified as b. So, a classified as a will be the TP as you learned in the class. So, you can also calculate the sensitivity other parameters from this confusion matrix. Since percentage split divides your data 70 30 randomly.

So, by changing the random seed parameter, you will get different results for example, normally if I run the algorithm again you will find the results are do not change because, WEKA does not change the random seed, in case you want to get different results go to more option, you will see a field random seed change it.

Now, if I run it again you will see the result change, this is because now different 70 and 30 percent of the data is taken as training and test. So, each time the random variable is change different section of the dataset will be taken as training and test and,

corresponding value will change. In case you are using this for your work, does do it more times.

And get the mean and variance of the each parameter, the other way of validation is a cross validation for example, in a tenfold cross validation the data is split into 10 equal parts and 9 is used for training and remaining one is used for test in first iteration and, this is repeated 10 times so, we from the 10 iteration every data will be used as training and also test.

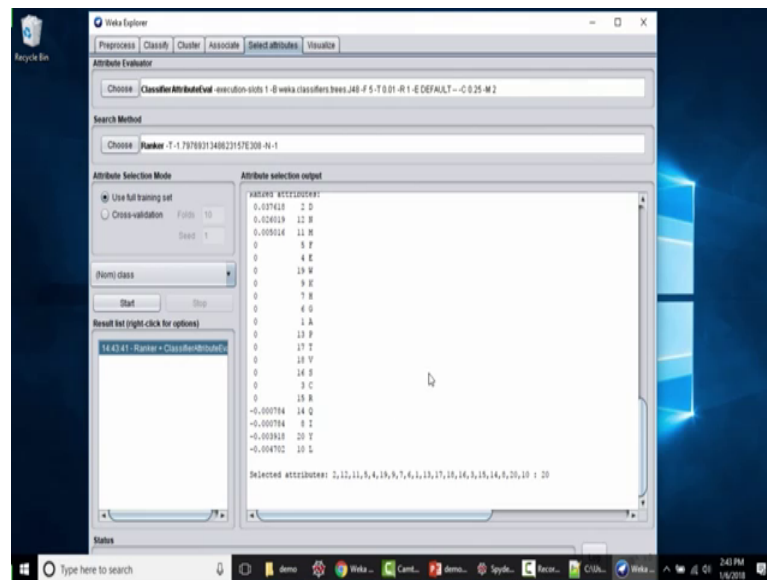
Cross validation is a better method, if you have a limited data and you need to validate. Percentage split is better, if you have a large data and, you can afford to lose some of the data for testing because, those information would not be available for your model. So, I used cross validation and run my results, you could see I get a got a 97 percent accuracy, which is pretty good, you can also try other methods other classifiers data mining is kind of exponential science. Let me choose a meta classifier there are 4 types of meta classifier one which does bagging, boosting and there is a stacking, the other type is a randomized randomization classifier.

So, stacking simply uses two algorithms and stack it one by one. So, input of one will be from the output of the other. So, let us use bagging. So, since I said these are meta classifier, they depend on other classifier. So, under the properties choose an another classifier, from your list let me use J 48 is a decision based classifier, it is a tree classifier, I am running the training the model and, here are the results.

In our case we are used 20, all the 20 amino acid composition as attributes, which is very huge you can reduce the number of attributes, because all a attribute own contribute your model performance, we can use select attributes further under select attribute, you could see the attribute evaluator the method and the search method, which is search algorithm to be used and the validation method again.

I am going to choose one of the evaluator, I am going to choose the classifier attribute evaluator and, the search method is chosen as ranker, I am going to use full training set, under the classifier attribute evaluator I am going to change the classifier.

(Refer Slide Time: 22:26)

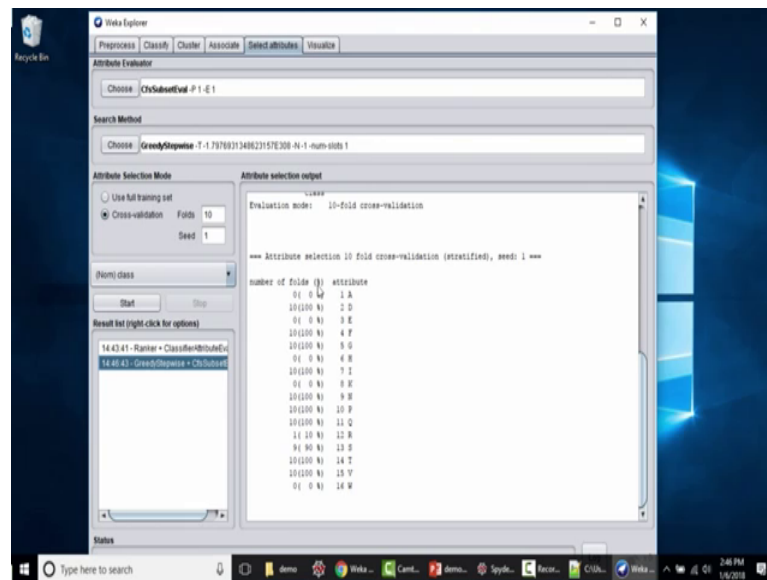


Now, the ranker has ranked the attributes, based on the importance for the performance based on this you can reduce some of the attributes. Let me remove some of the attributes for example, I am going to reduce 16 I am going to run the model again.

So, you can use feature selection to improve your model performance, let me choose ZeroR now, why ZeroR is a set as a default classifier WEKA is because, ZeroR works by predicting every data point as the class which is available frequently in our dataset for example, in our dataset alpha occurs 500 and 47 times which is more than beta.

So, ZeroR does it predicts everything as alpha as he could see the sensitivity is 1 and, accuracy is 0.8 0.7. So, you should remember that this is a baseline for your performance and so, to evaluate other methods only based on this performance, that is the reason ZeroR set has a default classifier in WEKA. So, anything which performs less than this may not be useful at all ok. So, let us go to our task since we have select attribute method.

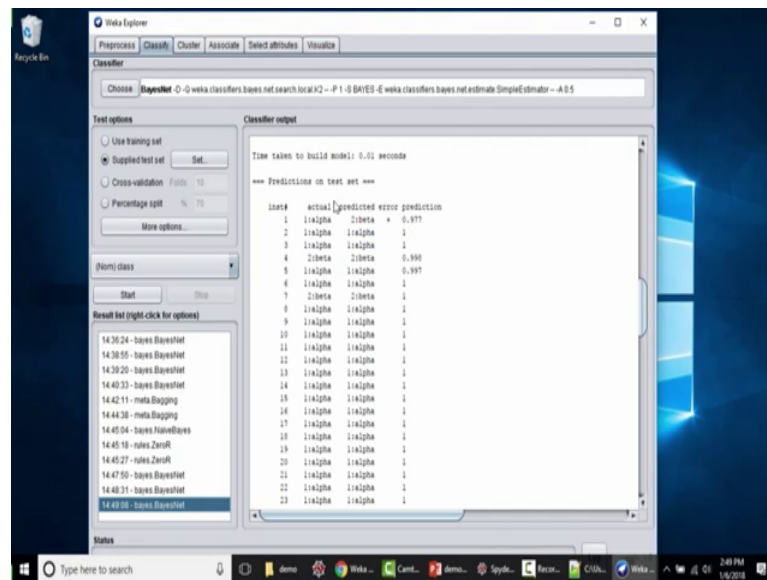
(Refer Slide Time: 25:15)



This time I am going to use cross validation and the here, here I could see many attributes do not contribute to the performance alanine, glutamic acid, histidine etcetera now I am going to remove this. I would suggest you to remove one by one because, though this results here show that attribute, attribute does not contact contribute to the performance, it may are an indirect role.

So, remove the attributes one by one and check the performance once, you are done with the attribute selection and proper validation let us test our model, selected model with the blind (Refer Time: 26:36). The model performance the on the test that is listed here sensitivity accuracy is 98 percentage. If you want to see the individual prediction go to more option, under output prediction choose plain text do ok.

(Refer Slide Time: 27:30)



The output predicted for each data point is listed here for example, first data point the actual class is alpha, the index for alpha is 1, but predicted as beta index 2. In case of error there will be a plus sign in this column and, the prediction probability is displayed here.

For each data point results are displayed here, the prediction probability for the sum of classifier such as Bayesnet and logistic regression is calculated in the algorithm whereas, for this it is calculated from the results. So, understand the difference between each classifier and use the WEKA wisely, you can save your model by clicking on the save model, which can be used later for new predictions.

Thank you.