

Chapter 9

How to Better Understand Your Machine Learning Data

It is important to take your time to learn about your data when starting on a new machine learning problem. There are key things that you can look at to very quickly learn more about your dataset, such as descriptive statistics and data visualizations. In this lesson you will discover how you can learn more about your data in the Weka machine learning workbench by reviewing descriptive statistics and visualizations of your data. After reading this lesson you will know about:

- The distribution of attributes from reviewing statistical summaries.
- The distribution of attributes from reviewing univariate plots.
- The relationship between attributes from reviewing multivariate plots.

Let's get started

9.1 Descriptive Statistics

The *Weka Explorer* will automatically calculate descriptive statistics for numerical attributes.

1. Open the *Weka GUI Chooser*.
2. Click *Explorer* to open the *Weka Explorer*.
3. Load the Pima Indians datasets from `data/diabetes.arff`.

The Pima Indians dataset contains numeric input variables that we can use to demonstrate the calculation of descriptive statistics. You can learn more about this dataset in [Section 8.2.1](#). Firstly, note that the dataset summary in the *Current Relation* section. This pane summarizes the following details about the loaded datasets:

- Dataset name (relation).
- The number of rows (instances).

- The number of columns (attributes).

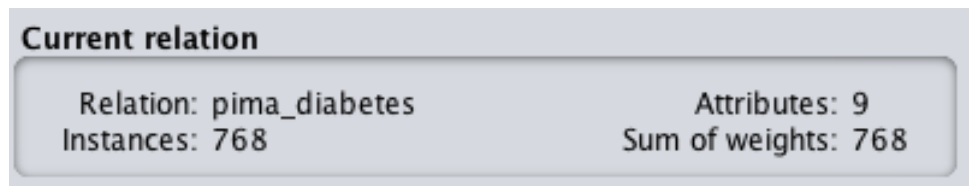


Figure 9.1: Weka Summary of Dataset.

- Click on the first attribute in the dataset in the *Attributes* pane.

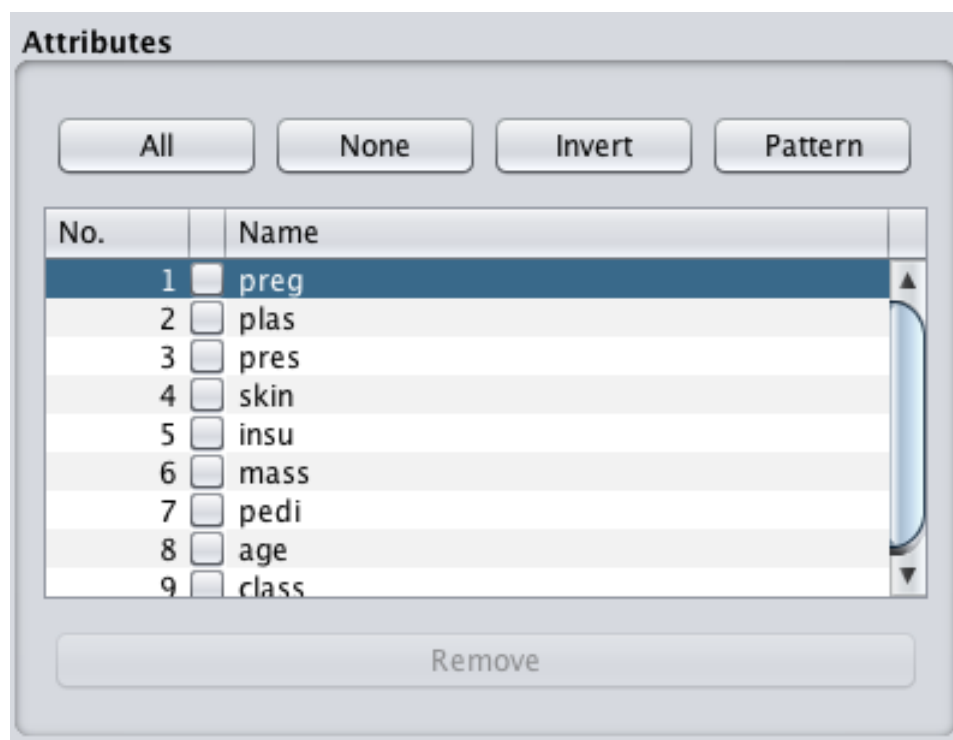
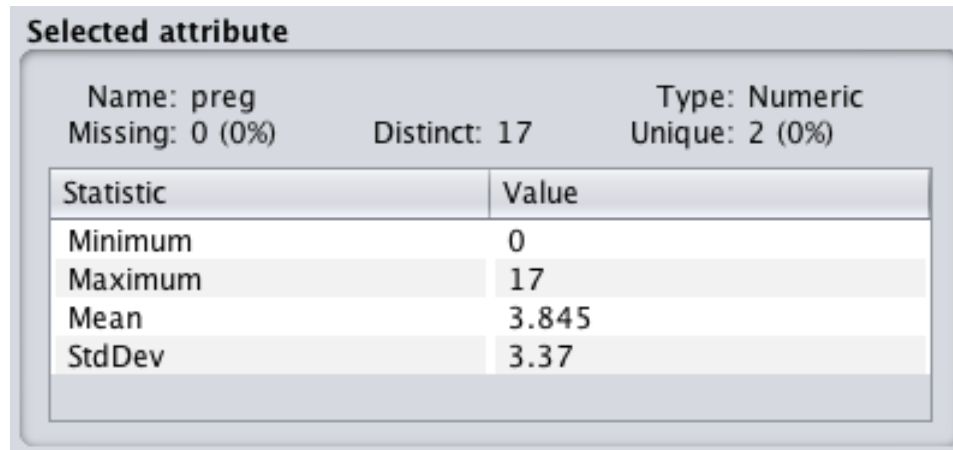


Figure 9.2: Weka List of Attributes.

Take note of the details in the *Selected attribute* pane. It lists a lot of information about the selected attribute, such as:

- The name of the attribute.
- The number of missing values and the ratio of missing values across the whole dataset.
- The number of distinct values.
- The data type.



The image shows a screenshot of the 'Selected attribute' window in Weka. It displays summary statistics for the 'preg' attribute, which is of type 'Numeric'. The summary includes the number of missing values (0), distinct values (17), and unique values (2). Below this, a table lists four statistics: Minimum (0), Maximum (17), Mean (3.845), and StdDev (3.37).

Selected attribute		
Name: preg	Distinct: 17	Type: Numeric
Missing: 0 (0%)		Unique: 2 (0%)
Statistic	Value	
Minimum	0	
Maximum	17	
Mean	3.845	
StdDev	3.37	

Figure 9.3: Weka Summary of Attribute.

The table below lists a number of descriptive statistics and their values. A useful four number summary is provided for numeric attributes including:

- Minimum value.
- Maximum value.
- Mean value.
- Standard deviation.

You can learn a lot from this information. For example:

- The presence and ratio of missing data can give you an indication of whether or not you need to remove or impute values.
- The mean and standard deviation give you a quantified idea of the spread of data for each attribute.
- The number of distinct values can give you an idea of the granularity of the attribute distribution.
- Click the class attribute. This attribute has a nominal type. Review the *Selected attribute* pane.

Selected attribute			
Name: class		Type: Nominal	
Missing: 0 (0%)		Distinct: 2	
		Unique: 0 (0%)	
No.	Label	Count	Weight
1	tested_negat...	500	500.0
2	tested_positive	268	268.0

Figure 9.4: Weka Summary of Class Attribute.

We can now see that for nominal attributes that we are provided with a list of each category and the count of instances that belong to each category. There is also mention of weightings, which we can ignore for now. This is used if we want to assign more or less weight to specific attribute values or instances in the dataset.

9.2 Univariate Attribute Distributions

The distribution of each attribute can be plotted to give a visual qualitative understanding of the distribution. Weka provides these plots automatically when you select an attribute in the *Preprocess* tab. We can follow on from the previous section where we already have the Pima Indians dataset loaded.

- Click on the **preg** attribute in the *Attributes* pane and note the plot below the *Selected attribute* pane.

You will see the distribution of **preg** values between 0 and 17 along the x-axis. The y-axis shows the count or frequency of values with each **preg** value.

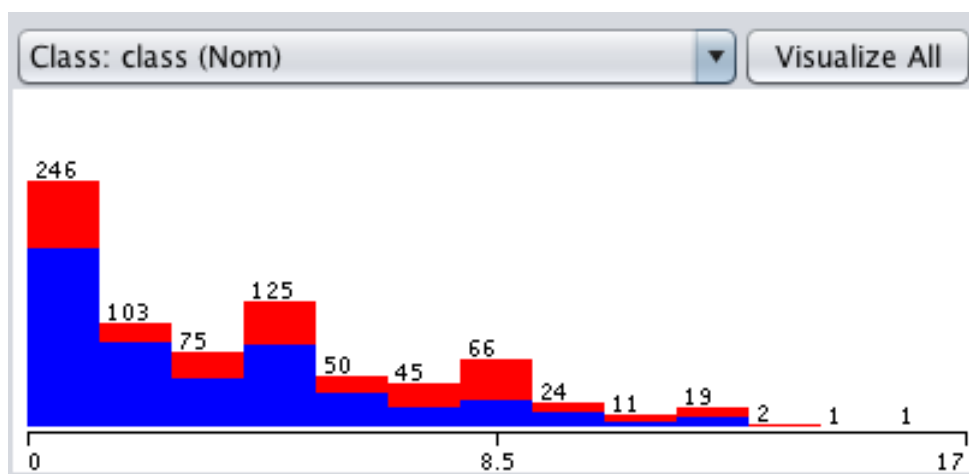


Figure 9.5: Weka Univariate Attribute Distribution.

Note the red and blue colors referring to the positive and negative classes respectively. The colors are assigned automatically to each categorical value. If there were three categories for the `class` value, we would see the breakdown of the `preg` distribution by three colors rather than two. This is useful to get a quick idea of whether the problem is easily separable for a given attribute, e.g. all the red and blue are cleanly separated for a single attribute. Clicking through each attribute in the list of *Attributes* and reviewing the plots, we can see that there is no such easy separation of the classes. We can quickly get an overview of the distribution of all attributes in the dataset and the breakdown of distributions by class by clicking the *Visualize All* button above the univariate plot.

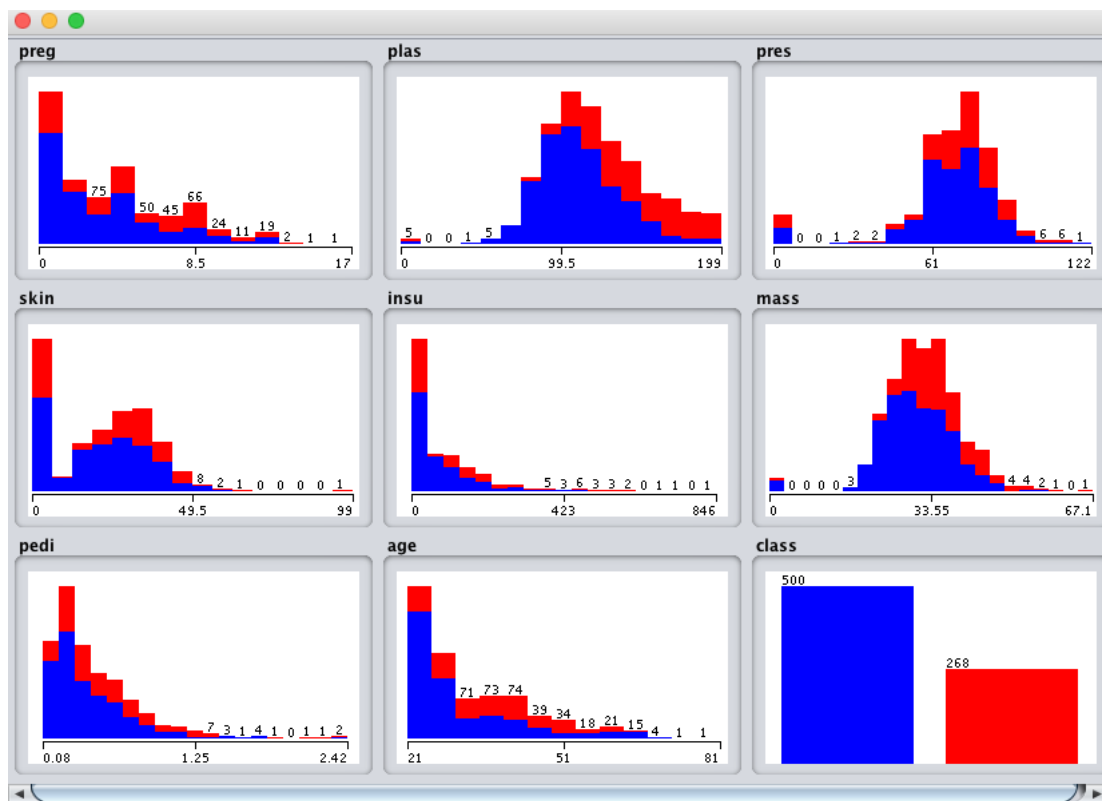


Figure 9.6: Weka All Univariate Attribute Distributions.

Looking at these plots we can see a few interesting things about this dataset.

- It looks like the `plas`, `pres` and `mass` attributes have a nearly Gaussian distribution.
- It looks like `pres`, `skin`, `insu` and `mass` have values at 0 that look out of place.

Looking at plots like this and jotting down things that come to mind can give you an idea of further data preparation operations that could be applied (like marking zero values as corrupt) and even techniques that might be useful (like linear discriminant analysis and logistic regression that assume a Gaussian distribution in input variables).

9.3 Visualize Attribute Interactions

So far we have only been looking at the properties of individual features, next we will look at patterns in combinations of attributes. When attributes are numeric we can create a scatter plot of one attribute against another. This is useful as it can highlight any patterns in the relationship between the attributes, such as positive or negative correlations. We can create scatter plots for all pairs of input attributes. This is called a scatter plot matrix and reviewing it before modeling your data can shed more light on further pre-processing techniques that you could investigate. Weka provides a scatter plot matrix for review by default in the *Visualise* tab.

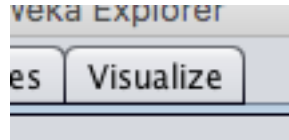


Figure 9.7: Weka Visualize Tab.

Continuing on from the previous section with the Pima Indians dataset loaded, click the *Visualize* tab, and make the window large enough to review all of the individual scatter plots.

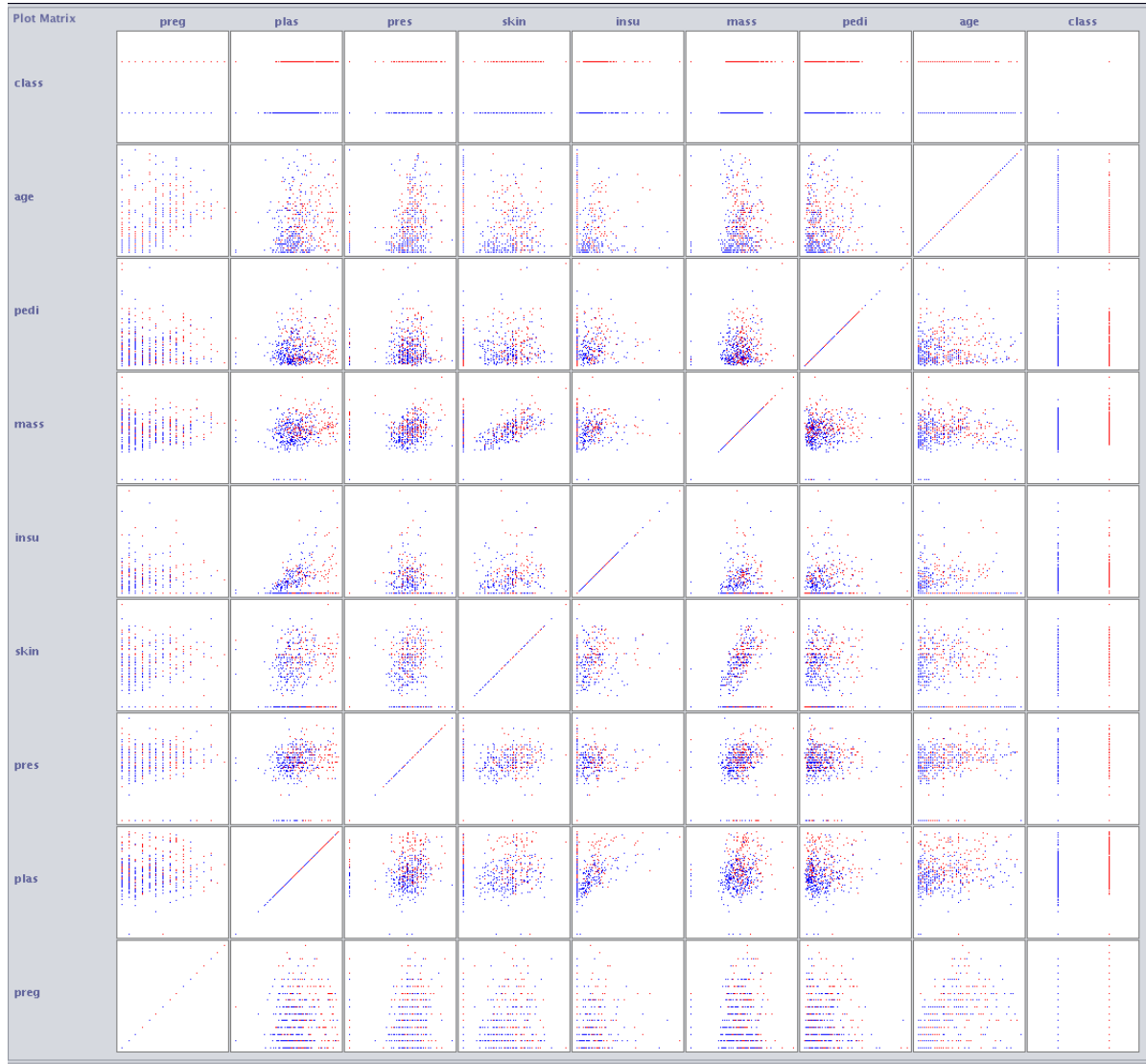


Figure 9.8: Weka Scatter Plot Matrix.

You can see that all combinations of attributes are plotted in a systematic way. You can also see that each plot appears twice, first in the top left triangle and again in the bottom right triangle with the axes flipped. You can also see a series of plots starting in the bottom left and continuing to the top right where each attribute is plotted against itself. These can be ignored. Finally, notice that the dots in the scatter plots are colored by their class value. It is good to look for trends or patterns in the dots, such as clear separation of the colors.

- Clicking on a plot will give you a new window with the plot that you can further play with.

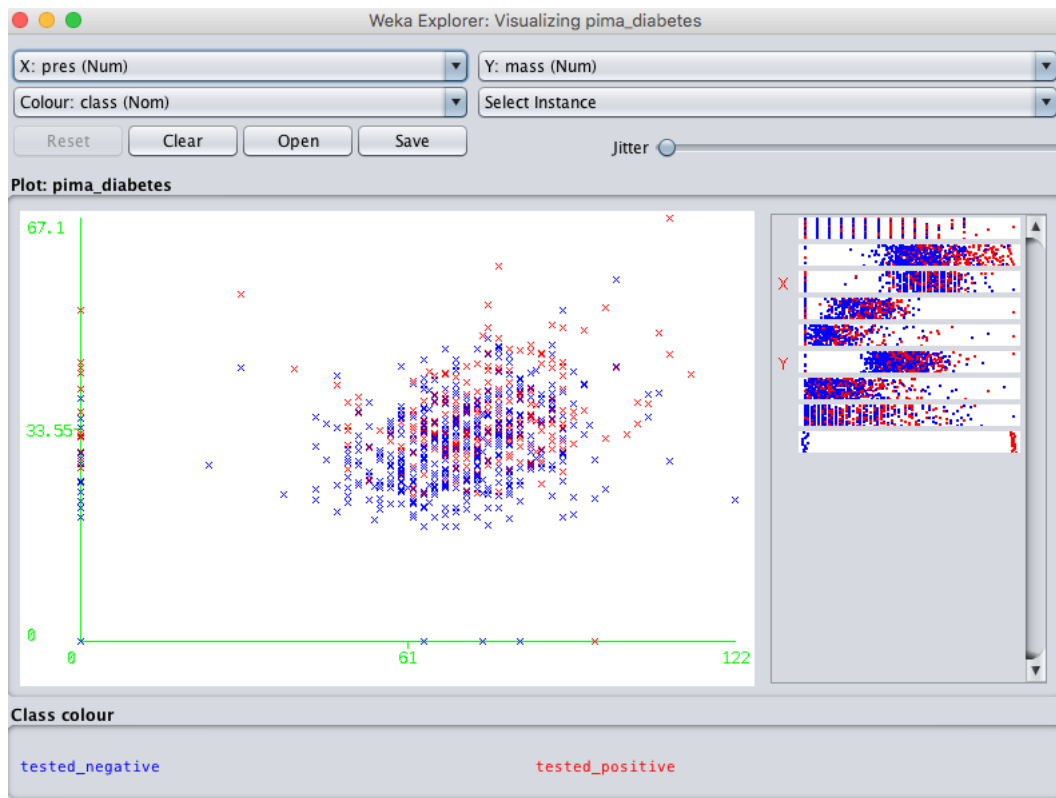


Figure 9.9: Weka Individual Scatter Plot.

Note the controls at the bottom of the screen. They let you increase the size of the plots, increase the size of the dots and add jitter. This last point about jitter is useful when you have a lot of dots overlaying each other and it is hard to see what is going on. Jitter will add some random noise to the data in the plots, spread out the points a bit and help you see what is going on. When you make a change to these controls, click the *Update* button to apply the changes.

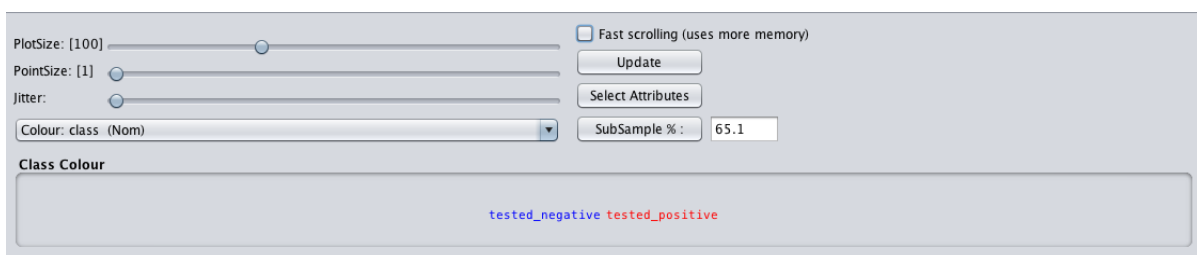


Figure 9.10: Weka Controls for Scatter Plot Matrix.

For example, below are the same plots with a larger dot size that makes it easier to see any trends in the data.

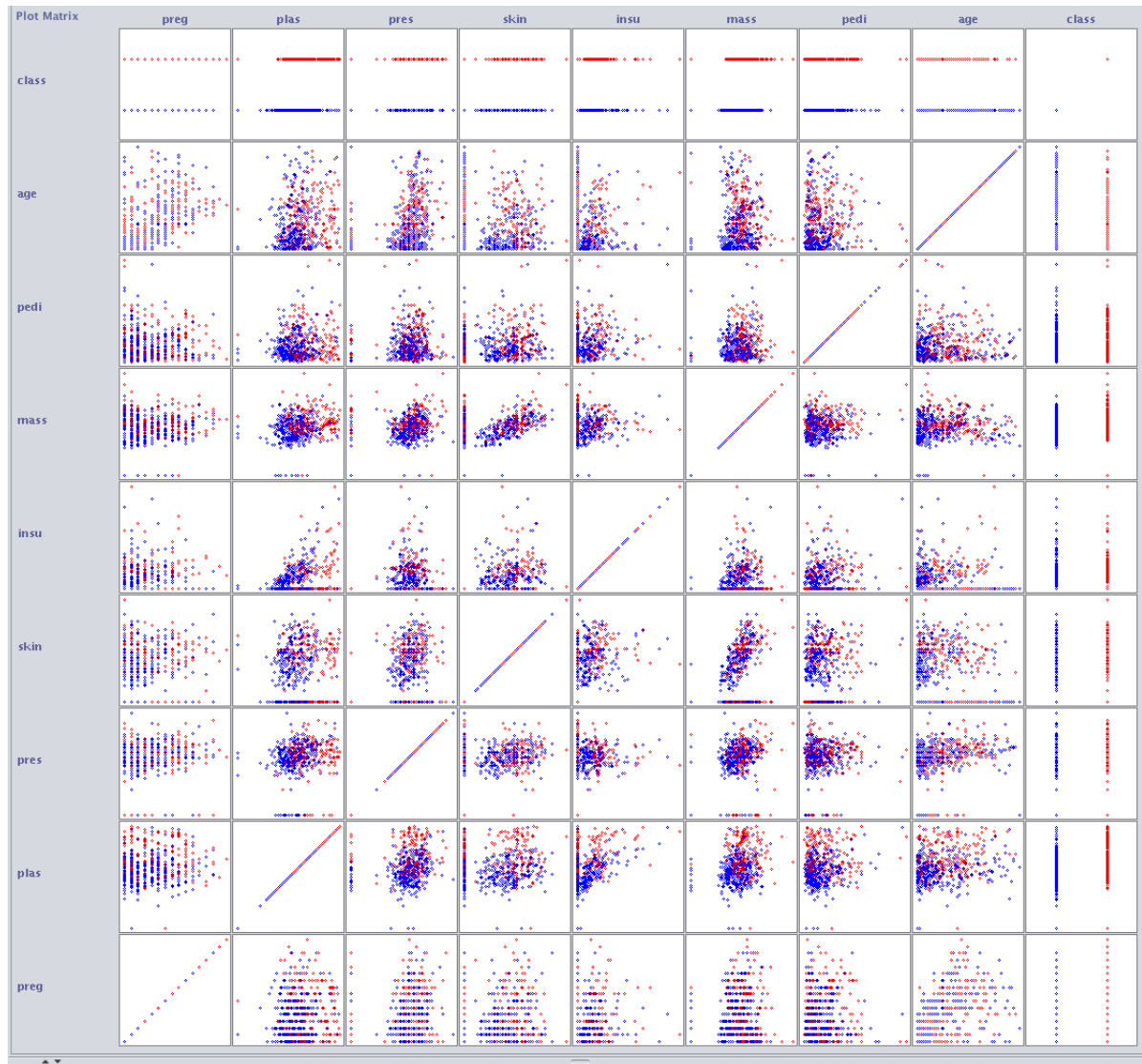


Figure 9.11: Weka Improved Scatter Plot Matrix.

9.4 Summary

In this lesson you discovered how you can learn more about your machine learning data by reviewing descriptive statistics and data visualizations. Specifically, you learned:

- That Weka automatically calculates descriptive statistics for each attribute.
- That Weka allows you to review the distribution of each attribute easily.
- That Weka provides a scatter plot visualization to review the pairwise relationships between attributes.

9.4.1 Next

We now know how to load and analyze data in Weka. In the next lesson we will learn about data filters and how they can be used to rescale numerical data.

Chapter 10

How to Normalize and Standardize Your Machine Learning Data

Machine learning algorithms make assumptions about the dataset you are modeling. Often, raw data is comprised of attributes with varying scales. For example, one attribute may be in kilograms and another may be a count. Although not required, you can often get a boost in performance by carefully choosing methods to rescale your data. In this lesson you will discover how you can rescale your data so that all of the data has the same scale. After reading this lesson you will know:

- How to normalize your numeric attributes between the range of 0 and 1.
- How to standardize your numeric attributes to have a zero mean and unit variance.
- When to choose normalization or standardization.

Let's get started.

10.1 About Data Filters in Weka

Weka provides filters for transforming your dataset. The best way to see what filters are supported and to play with them on your dataset is to use the *Weka Explorer*. The *Filter* pane allows you to choose a filter.

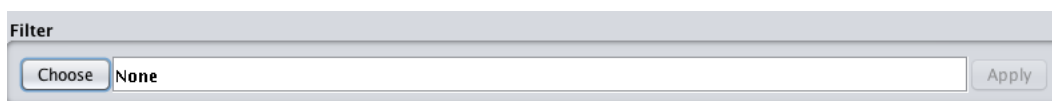


Figure 10.1: Weka Filter Pane for Choosing Data Filters.

Filters are divided into two types:

- **Supervised Filters:** That can be applied but require user control in some way. Such as rebalancing instances for a class.
- **Unsupervised Filters:** That can be applied in an undirected manner. For example, rescale all values to the range 0-to-1.

Personally, I think the distinction between these two types of filters is a little arbitrary and confusing. Nevertheless, that is how they are laid out. Within these two groups, filters are further divided into filters for Attributes and Instances:

- **Attribute Filters:** Apply an operation on attributes or one attribute at a time.
- **Instance Filters:** Apply an operation on instance or one instance at a time.

This distinction makes a lot more sense. After you have selected a filter, its name will appear in the box next to the *Choose* button. You can configure a filter by clicking its name which will open the configuration window. You can change the parameters of the filter and even save or load the configuration of the filter itself. This is great for reproducibility.

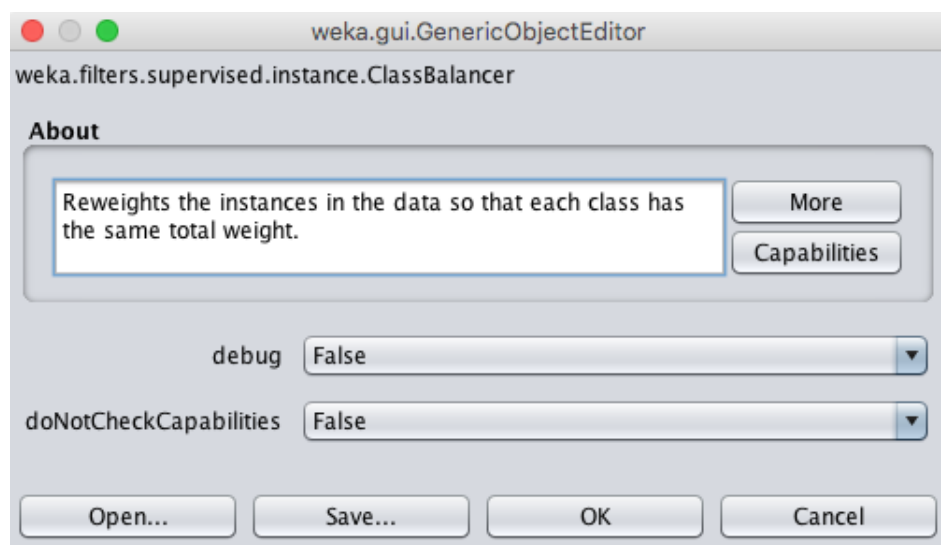


Figure 10.2: Weka Data Filter Configuration.

You can learn more about each configuration option by hovering over it and reading the tooltip. You can also read all of the details about the filter including the configuration, papers and books for further reading and more information about the filter works by clicking the *More* button.

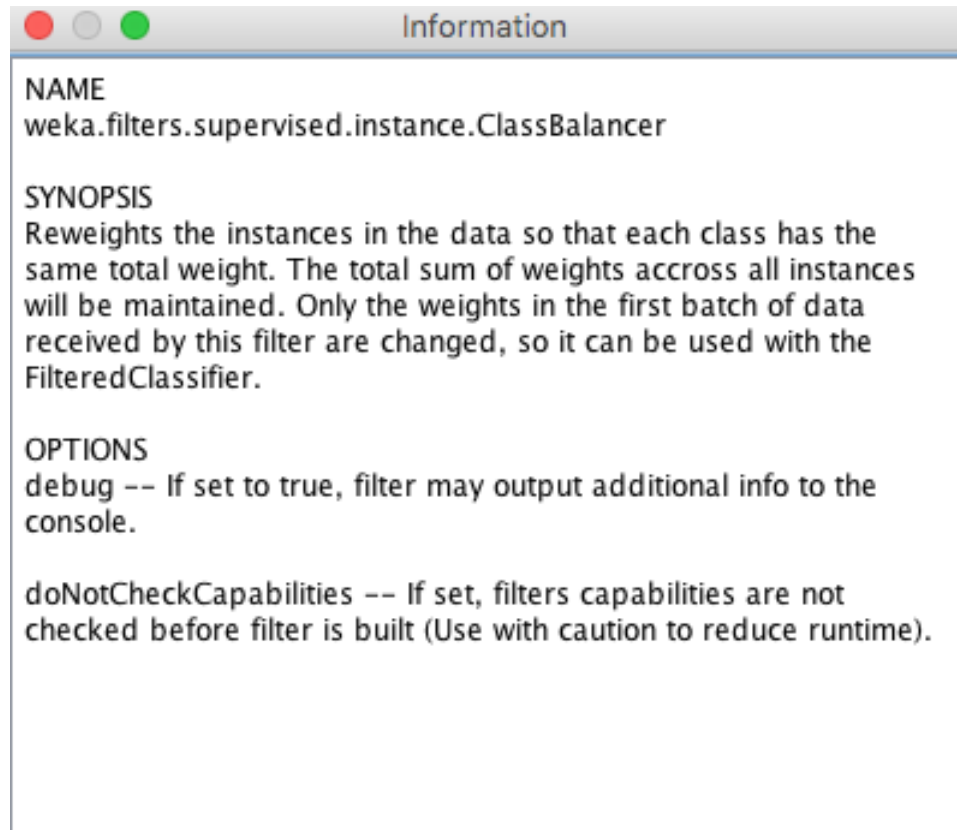


Figure 10.3: Weka Data Filter More Information.

You can close the help and apply the configuration by clicking the *OK* button. You can apply a filter to your loaded dataset by clicking the *Apply* button next to the filter name.

10.2 Normalize Your Numeric Attributes

Data normalization is the process of rescaling one or more attributes to the range of 0 to 1. This means that the largest value for each attribute is 1 and the smallest value is 0. Normalization is a good technique to use when you do not know the distribution of your data or when you know the distribution is not Gaussian (a bell curve).

The dataset used for this example is the Pima Indians onset of diabetes dataset. You can learn more about this dataset in Section 8.2.1. You can normalize all of the attributes in your dataset with Weka by choosing the *Normalize* filter and applying it to your dataset. You can use the following recipe to normalize your dataset:

- 1. Open the *Weka Explorer*.
- 2. Load the `data/diabetes.arff` dataset.

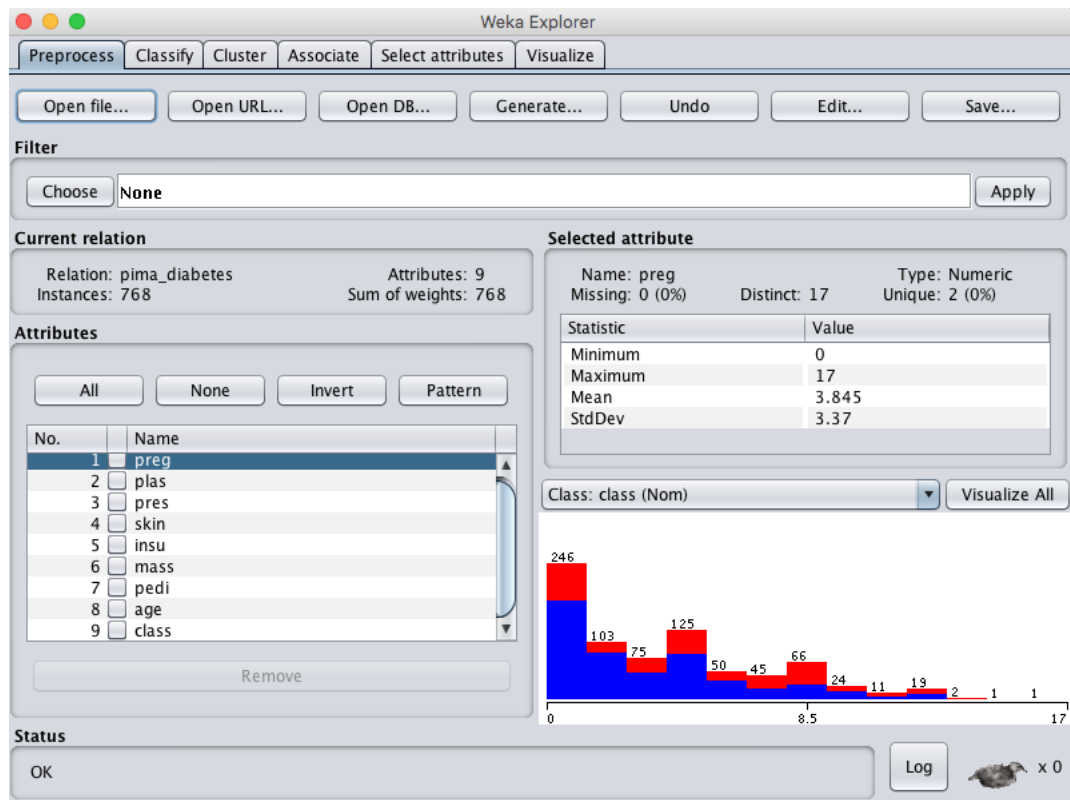


Figure 10.4: Weka Explorer Loaded Diabetes Dataset.

- 3. Click the *Choose* button and select the *unsupervised.attribute.Normalize* filter.

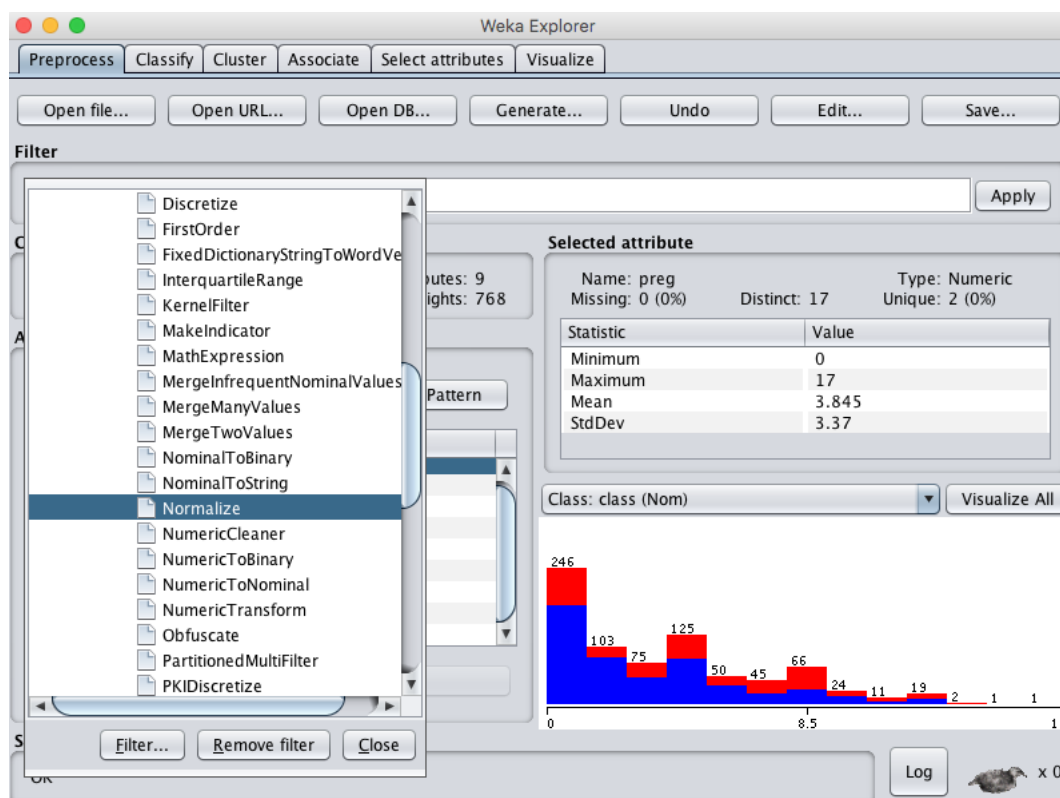


Figure 10.5: Weka Select Normalize Data Filter.

- 4. Click the *Apply* button to normalize your dataset.
- 5. Click the *Save* button and type a filename to save the normalized copy of your dataset.

Reviewing the details of each attribute in the *Selected attribute* window will give you confidence that the filter was successful and that each attribute was rescaled to the range of 0 to 1.

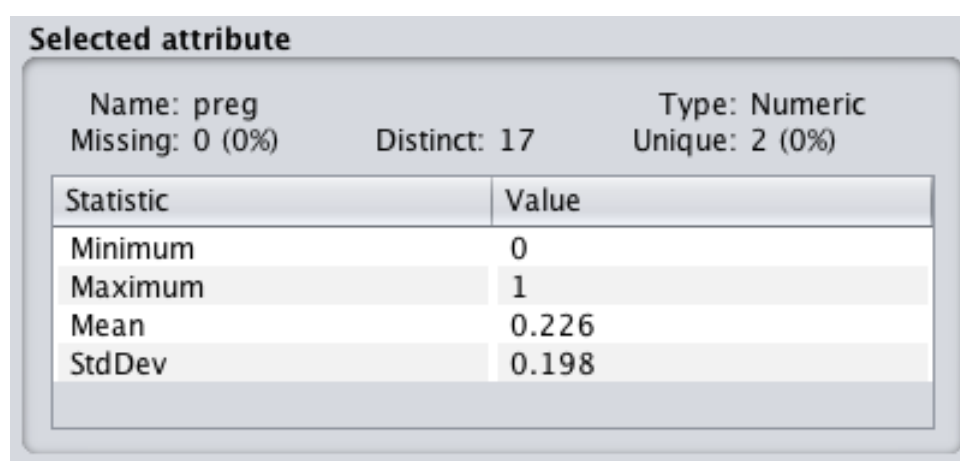


Figure 10.6: Weka Normalized Data Distribution.

You can use other scales such as -1 to 1, which is useful when using Support Vector Machines and AdaBoost. Normalization is useful when your data has varying scales and the algorithm

you are using does not make assumptions about the distribution of your data, such as k -Nearest Neighbors and Artificial Neural Networks.

10.3 Standardize Your Numeric Attributes

Data standardization is the process of rescaling one or more attributes so that they have a mean value of 0 and a standard deviation of 1. Standardization assumes that your data has a Gaussian (bell curve) distribution. This does not strictly have to be true, but the technique is more effective if your attribute distribution is Gaussian. You can standardize all of the attributes in your dataset with Weka by choosing the *Standardize* filter and applying it your dataset. You can use the following recipe to standardize your dataset:

- 1. Open the *Weka Explorer*.
- 2. Load the `data/diabetes.arff` dataset.
- 3. Click the *Choose* button to and select the *unsupervised.attribute.Standardize* filter.

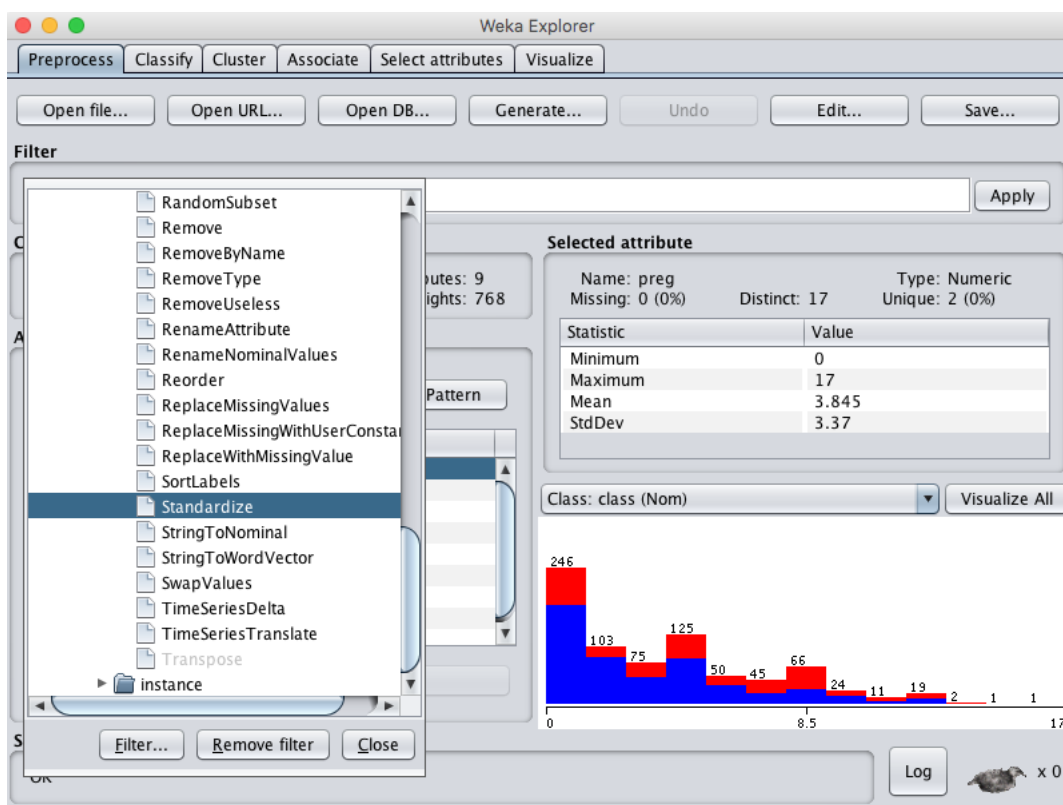
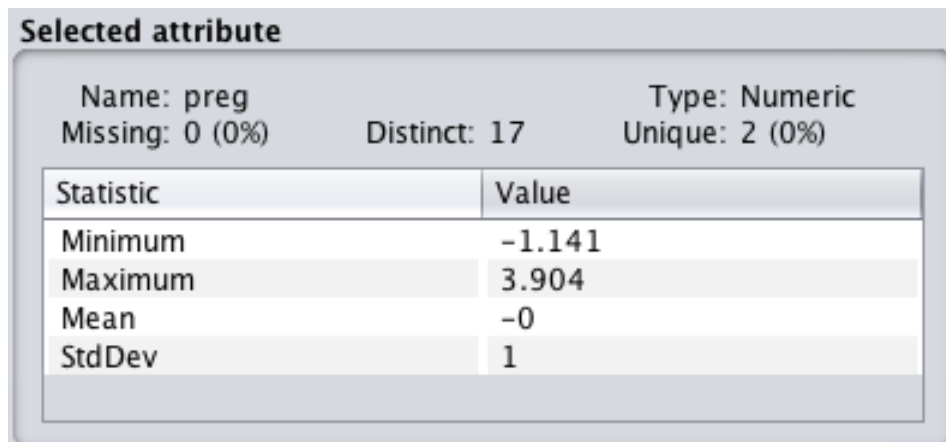


Figure 10.7: Weka Select Standardize Data Filter.

- 4. Click the *Apply* button to normalize your dataset.
- 5. Click the *Save* button and type a filename to save the standardized copy of your dataset.

Reviewing the details of each attribute in the *Selected attribute* window will give you confidence that the filter was successful and that each attribute has a mean of 0 and a standard deviation of 1.



Selected attribute		
Name: preg	Distinct: 17	Type: Numeric
Missing: 0 (0%)		Unique: 2 (0%)
Statistic	Value	
Minimum	-1.141	
Maximum	3.904	
Mean	-0	
StdDev	1	

Figure 10.8: Weka Standardized Data Distribution.

Standardization is useful when your data has varying scales and the algorithm you are using does make assumptions about your data having a Gaussian distribution, such as linear regression, logistic regression and linear discriminant analysis.

10.4 Summary

In this lesson you discovered how to rescale your dataset in Weka. Specifically, you learned:

- How to normalize your dataset to the range 0 to 1.
- How to standardize your data to have a mean of 0 and a standard deviation of 1.
- When to use normalization and standardization.

10.4.1 Next

Weka provides a large assortment of data filters. In the next lesson you will learn how you can transform attributes using more advanced data filters.

Chapter 11

How to Transform Your Machine Learning Data

Often your raw data for machine learning is not in an ideal form for modeling. You need to prepare or reshape it to meet the expectations of different machine learning algorithms. In this lesson you will discover two techniques that you can use to transform your machine learning data ready for modeling. After reading this lesson you will know:

- How to convert a real valued attribute into a discrete distribution called discretization.
- How to convert a discrete attribute into multiple real values called dummy variables.
- When to discretize or create dummy variables from your data.

Let's get started.

11.1 Discretize Numerical Attributes

Some machine learning algorithms prefer or find it easier to work with discrete attributes. For example, decision tree algorithms can choose split points in real valued attributes, but are much cleaner when split points are chosen between bins or predefined groups in the real-valued attributes. Discrete attributes are those that describe a category, called nominal attributes. Those attributes that describe a category that where there is a meaning in the order for the categories are called ordinal attributes. The process of converting a real-valued attribute into an ordinal attribute or bins is called discretization.

You can discretize your real valued attributes in Weka using the *Discretize* filter. The tutorial below demonstrates how to use the *Discretize* filter. The Pima Indians onset of diabetes dataset is used to demonstrate this filter because of the input values are real-valued and grouping them into bins may make sense. You can learn more about this dataset in [Section 8.2.1](#).

- 1. Open the *Weka Explorer*.
- 2. Load the `data/diabetes.arff` dataset.

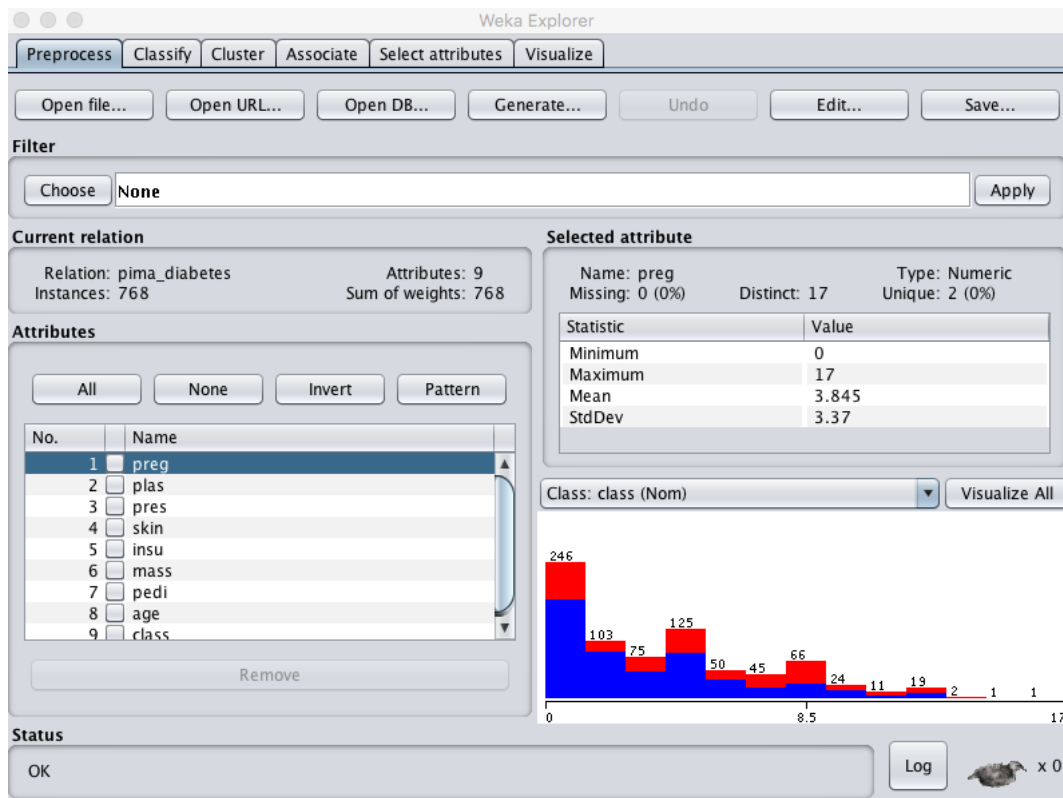


Figure 11.1: Weka Explorer Loaded Diabetes Dataset.

- 3. Click the *Choose* button for the Filter and select the *unsupervised.attribute.Discretize* filter.

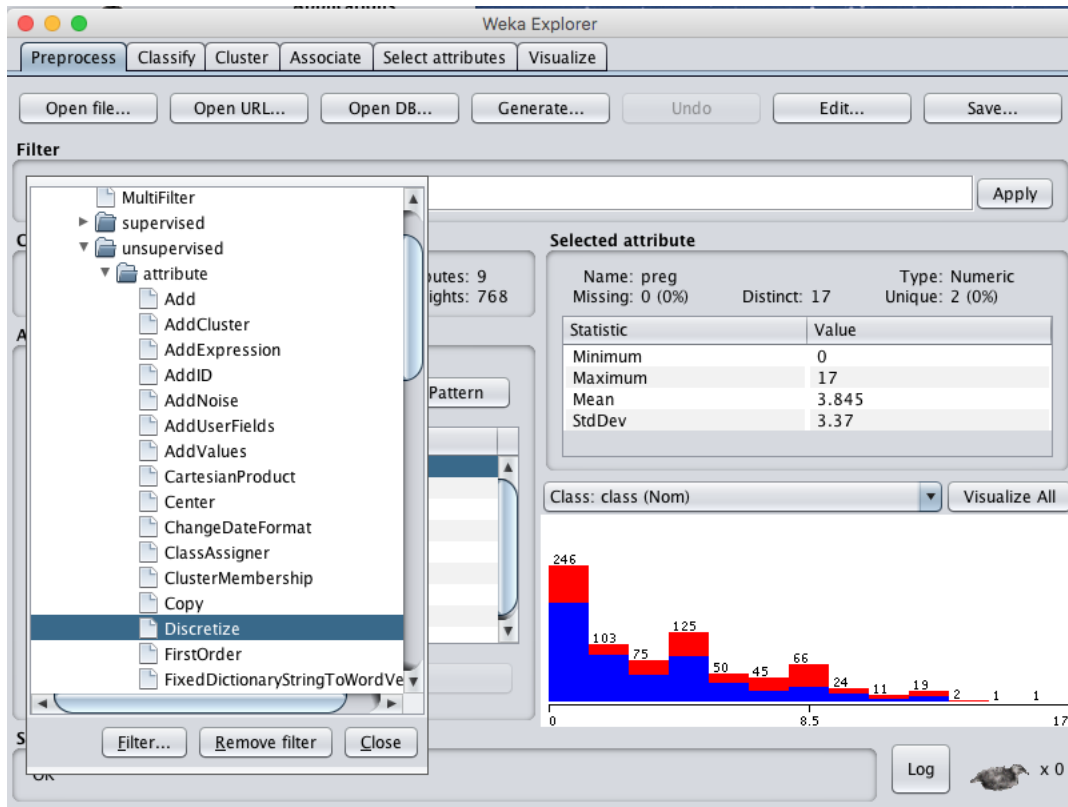


Figure 11.2: Weka Select Discretize Data Filter.

- 4. Click on the filter to configure it. You can select the indices of the attributes to discretize, the default is to discretize all attributes, which is what we will do in this case. Click the *OK* button.
- 5. Click the *Apply* button to apply the filter.

You can click on each attribute and review the details in the *Selected attribute* pane to confirm that the filter was applied successfully.

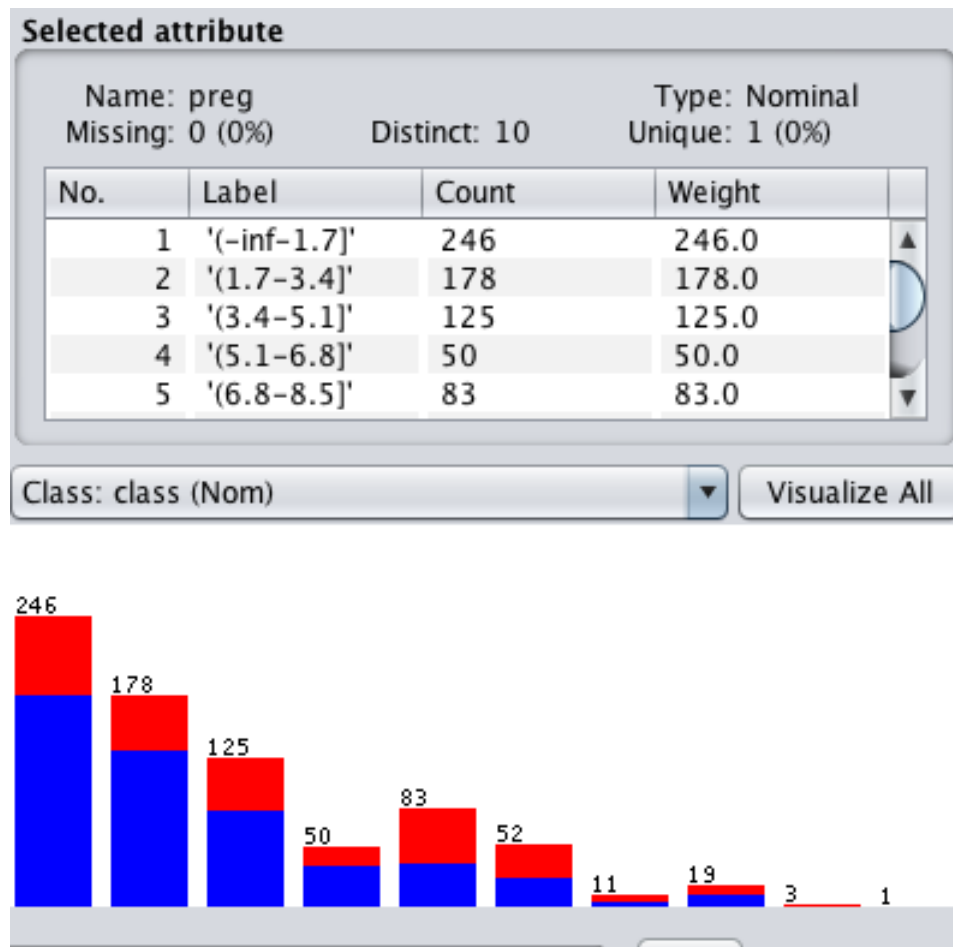


Figure 11.3: Weka Discretized Attribute.

Discretizing your real valued attributes is most useful when working with decision tree type algorithms. It is perhaps more useful when you believe that there are natural groupings within the values of given attributes.

11.2 Convert Nominal Attributes to Dummy Variables

Some machine learning algorithms prefer to use real valued inputs and do not support nominal or ordinal attributes. Nominal attributes can be converted to real values. This is done by creating one new binary attribute for each category. For a given instance that has a category for that value, the binary attribute is set to 1 and the binary attributes for the other categories is set to 0. This process is called creating dummy variables.

You can create dummy binary variables from nominal attributes in Weka using the *NominalToBinary* filter. The recipe below demonstrates how to use the *NominalToBinary* filter. The Contact Lenses dataset is used to demonstrate this filter because the attributes are all nominal and provide plenty of opportunity for creating dummy variables. You can download the Contact Lenses dataset from the UCI Machine learning repository. You can also access the dataset directory in your installation of Weka under the `data/` directory by loading the file `contact-lenses.arff`.

- 1. Open the *Weka Explorer*.
- 2. Load the `data/contact-lenses.arff` dataset.

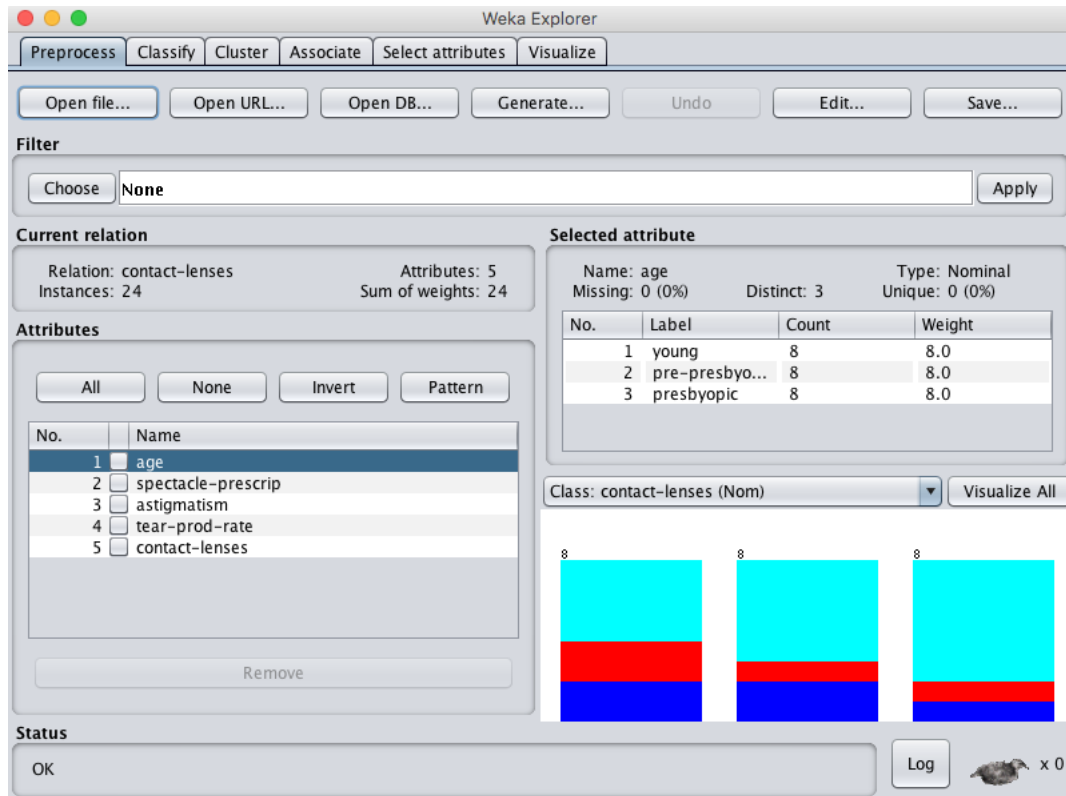


Figure 11.4: Weka Explorer Loaded Contact Lenses Dataset.

- 3. Click the *Choose* button the *unsupervised.attribute.NominalToBinary* filter.

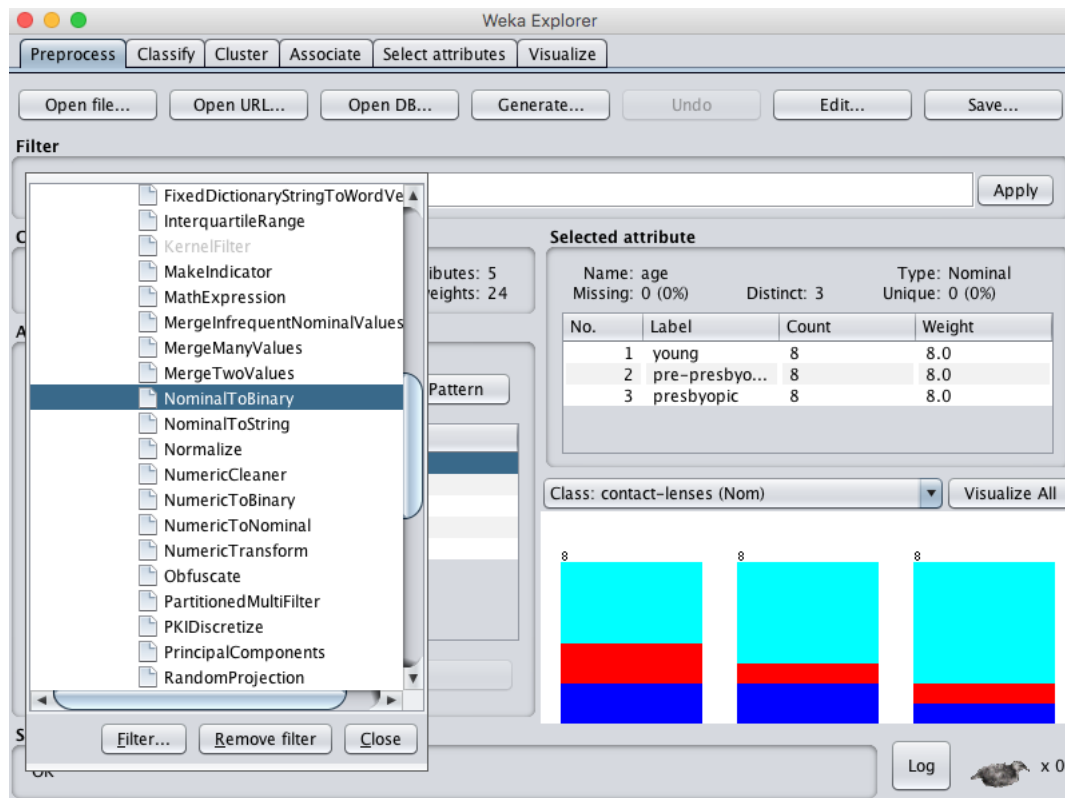


Figure 11.5: Weka Select NominalToBinary Data Filter.

- 4. Click on the filter to configure it. You can select the indices of the attributes to convert to binary values, the default is to convert all attributes. Change it to only the first attribute. Click the *OK* button.

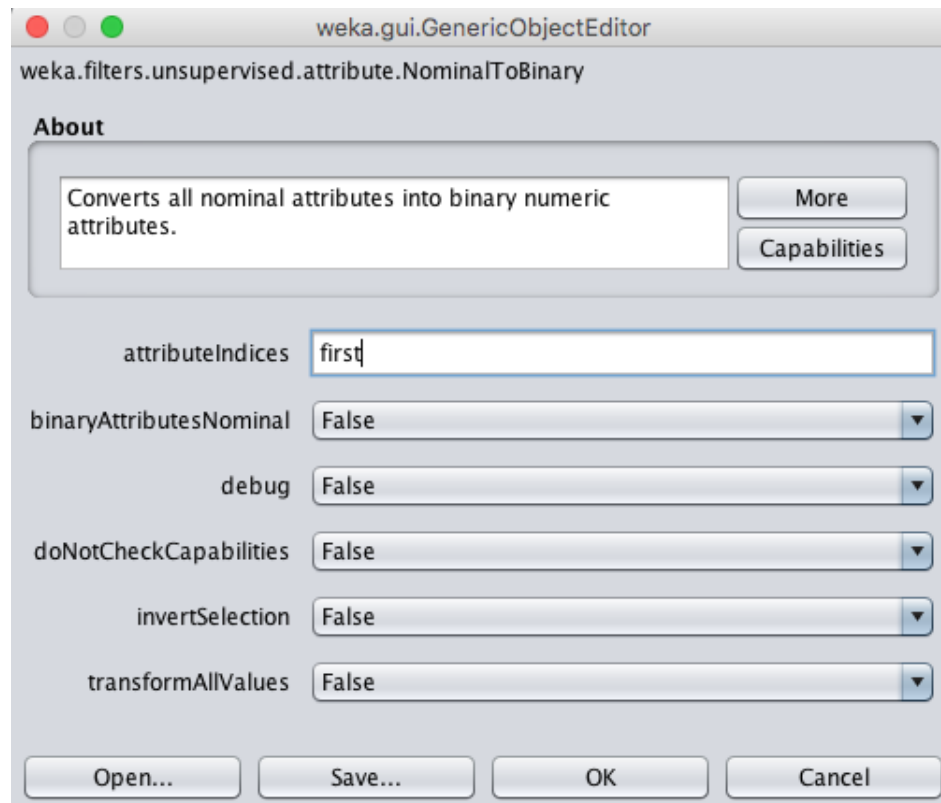


Figure 11.6: Weka NominalToBinary Data Filter Configuration.

- 5. Click the *Apply* button to apply the filter.

Reviewing the list of attributes will show that the age attribute has been removed and replaced with three new binary attributes: `age=young`, `age=pre-presbyopic` and `age=presbyopic`.

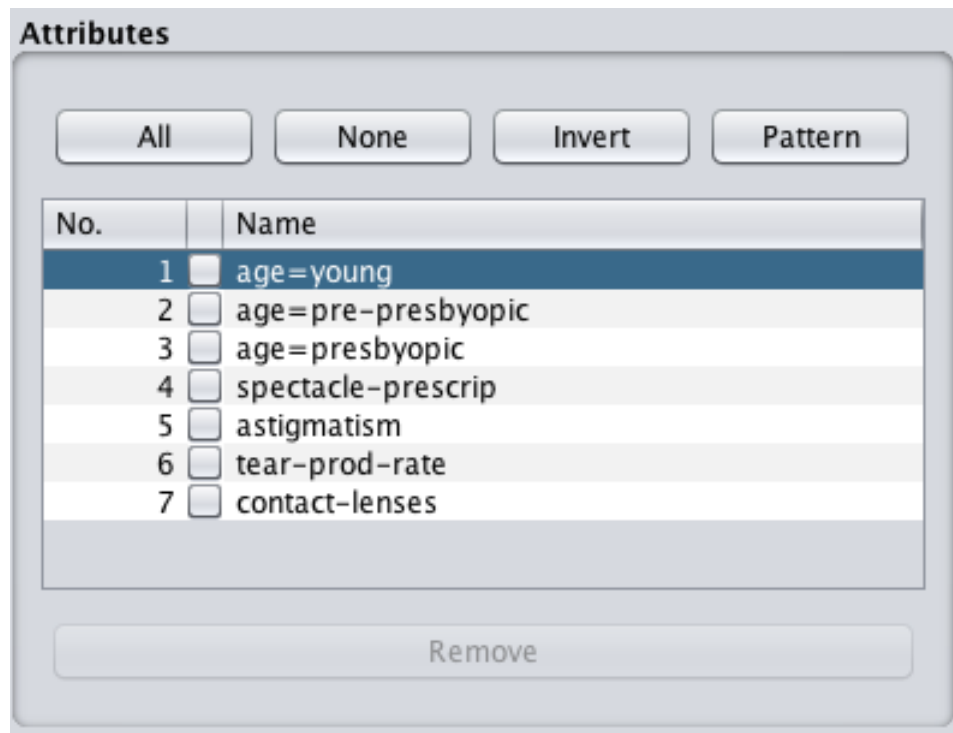


Figure 11.7: Weka Nominal Attribute Converted to Dummy Variables.

Creating dummy variables is useful for techniques that do not support nominal input variables like linear regression and logistic regression. It can also prove useful in techniques like k -nearest neighbors and artificial neural networks.

11.3 Summary

In this lesson you discovered how to transform your machine learning data to meet the expectations of different machine learning algorithms. Specifically, you learned:

- How to convert real valued input attributes to nominal attributes called discretization.
- How to convert a categorical input variable to multiple binary input attributes called dummy variables.
- When to use discretization and dummy variables when modeling data.

11.3.1 Next

Raw data is often not suitable for modeling directly. It may need to be cleaned first. In the next lesson you will learn how to identify, mark, remove and impute missing values in your dataset.

Chapter 12

How To Handle Missing Values In Machine Learning Data

Data is rarely clean and often you can have corrupt or missing values. It is important to identify, mark and handle missing data when developing machine learning models in order to get the very best performance. In this lesson you will discover how to handle missing values in your machine learning data using Weka. After reading this lesson you will know:

- How to mark missing values in your dataset.
- How to remove data with missing values from your dataset.
- How to impute missing values.

Let's get started.

12.1 Mark Missing Values

The problem used for this example is the Pima Indians onset of diabetes dataset. You can learn more about this dataset in Section 8.2.1. The Pima Indians dataset is a good basis for exploring missing data. Some attributes such as blood pressure (**pres**) and Body Mass Index (**mass**) have values of zero, which are impossible. These are examples of corrupt or missing data that must be marked manually. You can mark missing values in Weka using the *NumericCleaner* filter. The recipe below shows you how to use this filter to mark the 11 missing values on the Body Mass Index (**mass**) attribute.

- 1. Open the *Weka Explorer*.
- 2. Load the `data/diabetes.arff` dataset.
- 3. Click the *Choose* button and select the *unsupervised.attribute.NumericCleaner* filter.

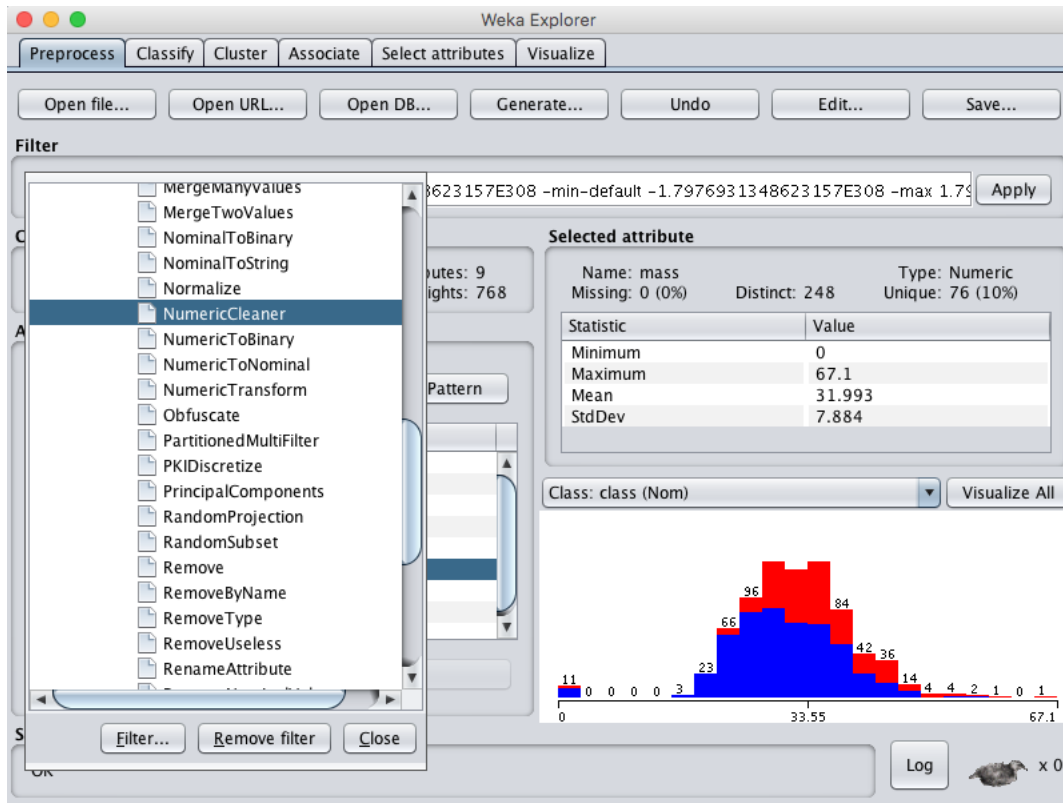


Figure 12.1: Weka Select NumericCleaner Data Filter.

- 4. Click on the filter to configure it.
- 5. Set the *attributeIndices* to 6, the index of the mass attribute.
- 6. Set *minThreshold* to 0.1E-8 (close to zero), which is the minimum value allowed for the attribute.
- 7. Set *minDefault* to NaN, which is unknown and will replace values below the threshold.
- 8. Click the *OK* button on the filter configuration.
- 9. Click the *Apply* button to apply the filter.
- 10. Click **mass** in the *Attributes* pane and review the details of the *Selected attribute*.

Notice that the 11 attribute values that were formally set to 0 are now marked as *Missing*.

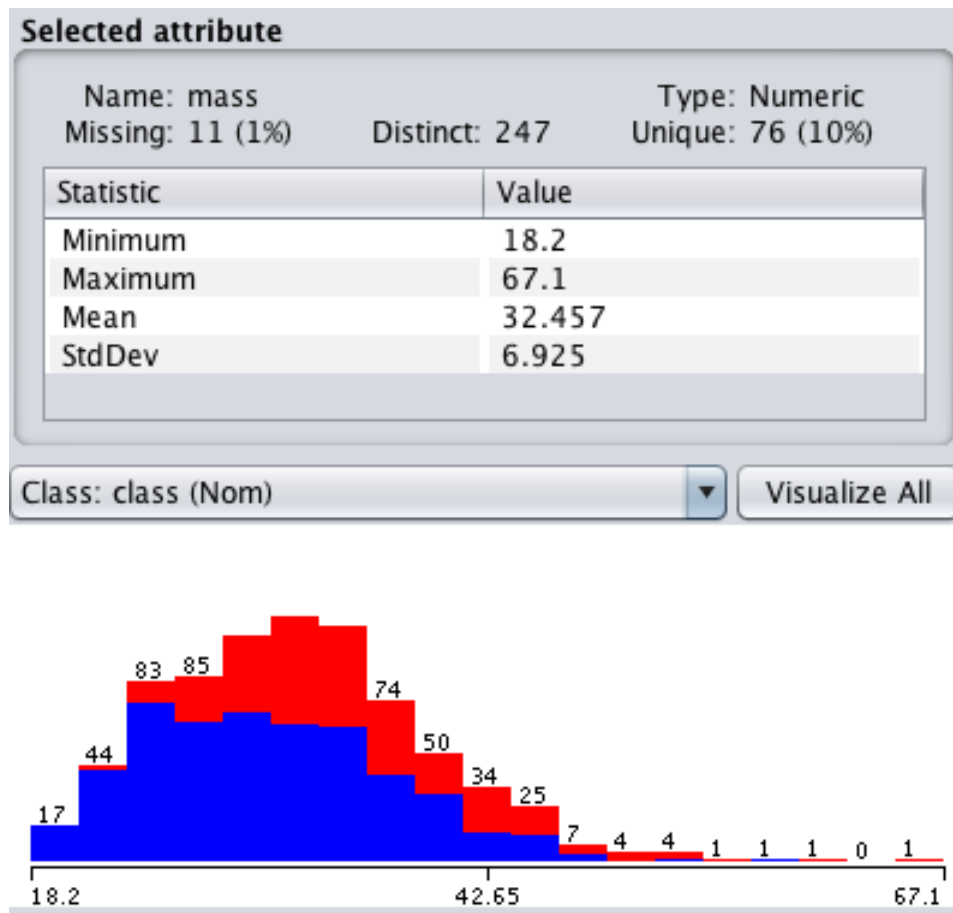


Figure 12.2: Weka Missing Data Marked.

In this example we marked values below a threshold as missing. You could just as easily mark them with a specific numerical value. You could also mark values missing between an upper and lower range of values. Next, let's look at how we can remove instances with missing values from our dataset.

12.2 Remove Missing Data

Now that you know how to mark missing values in your data, you need to learn how to handle them. A simple way to handle missing data is to remove those instances that have one or more missing values. You can do this in Weka using the *RemoveWithValues* filter. Continuing on from the above recipe to mark missing values, you can remove missing values as follows:

- 1. Click the *Choose* button and select the *unsupervised.instance.RemoveWithValues* filter.

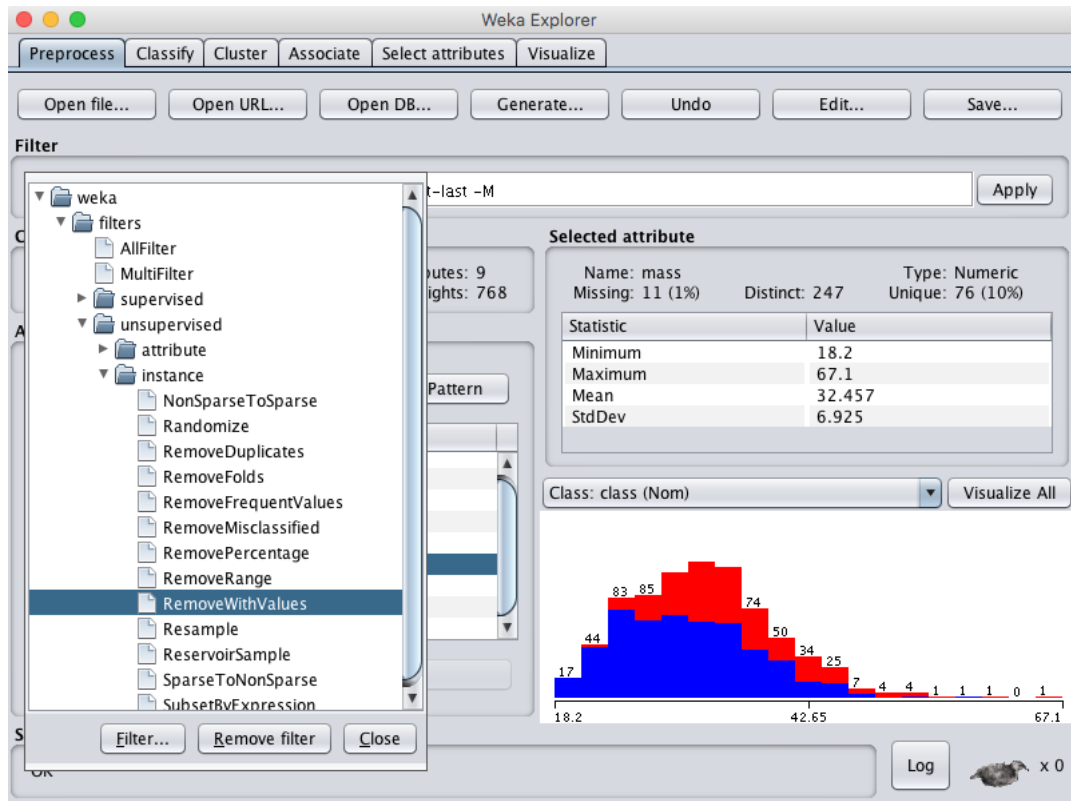


Figure 12.3: Weka Select RemoveWithValues Data Filter.

- 2. Click on the filter to configure it.
- 3. Set the *attributeIndices* to 6, the index of the mass attribute.
- 4. Set *matchMissingValues* to *True*.
- 5. Click the *OK* button to use the configuration for the filter.
- 6. Click the *Apply* button to apply the filter.
- 7. Click **mass** in the *Attributes* section and review the details of the *Selected attribute*.

Notice that the 11 attribute values that were marked Missing have been removed from the dataset.

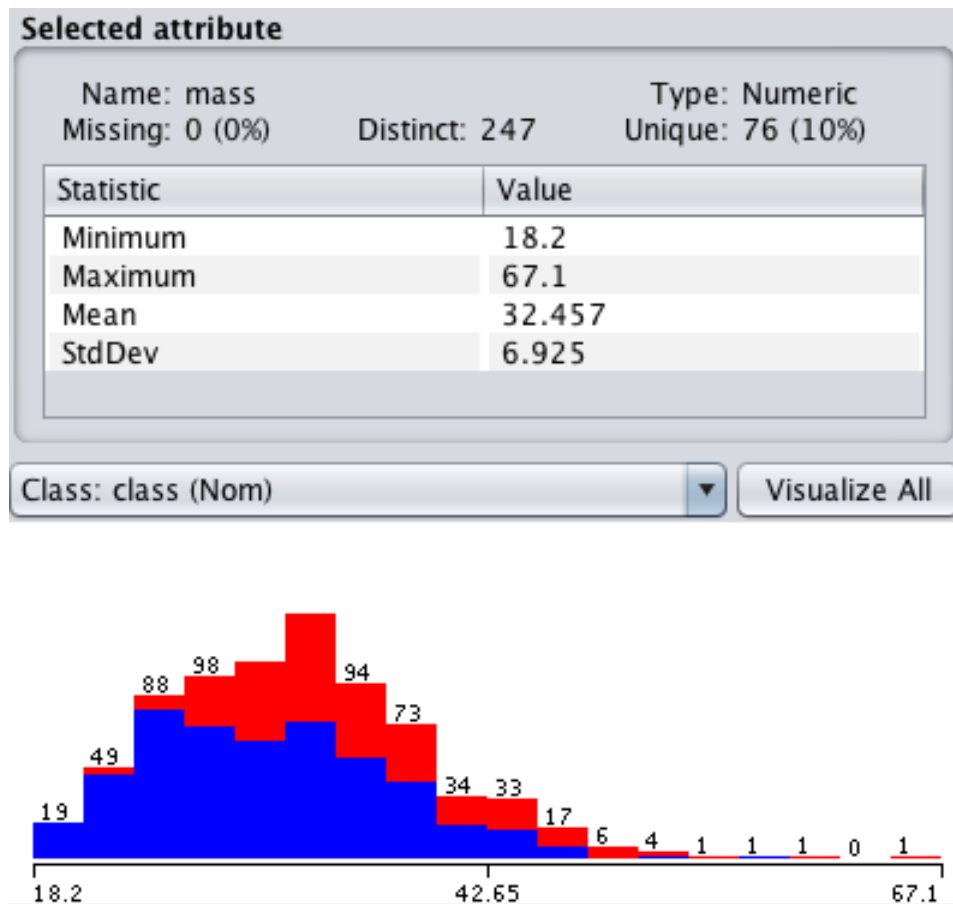


Figure 12.4: Weka Missing Values Removed.

Note, you can undo this operation by clicking the *Undo* button.

12.3 Impute Missing Values

Instances with missing values do not have to be removed, you can replace the missing values with some other value. This is called imputing missing values. It is common to impute missing values with the mean of the numerical distribution. You can do this easily in Weka using the *ReplaceMissingValues* filter. Continuing on from the first recipe above to mark missing values, you can impute the missing values as follows:

- 1. Click the *Choose* button and select the *unsupervised.attribute.ReplaceMissingValues* filter.

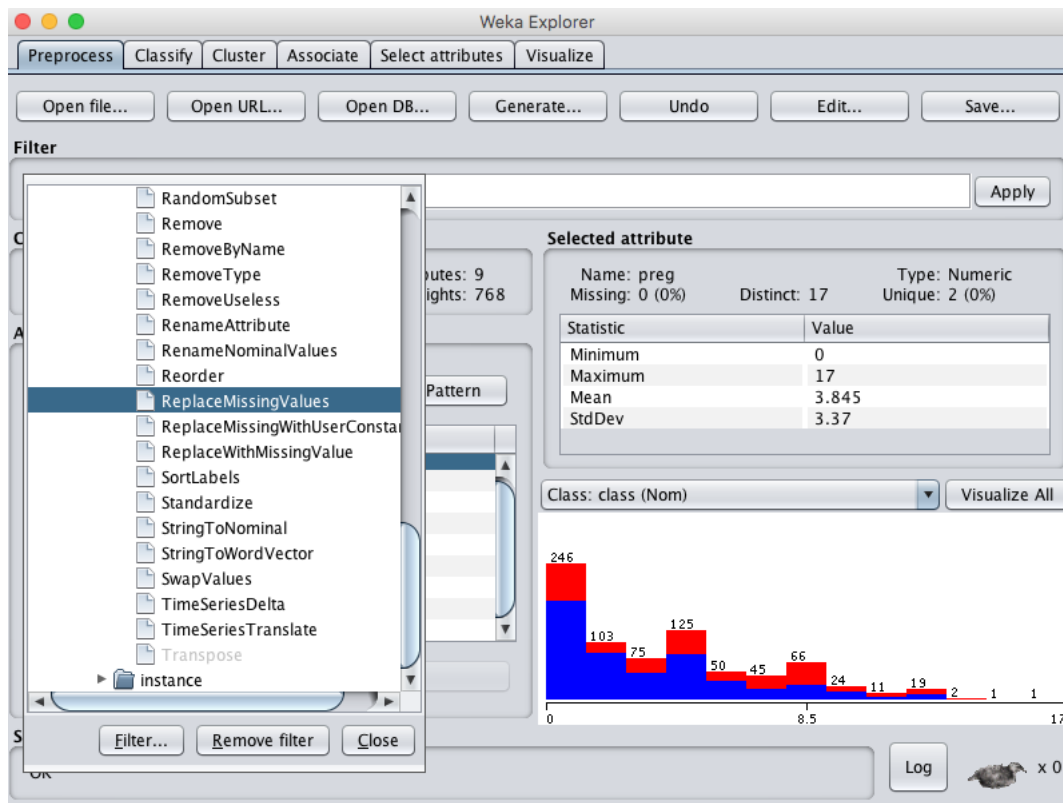


Figure 12.5: Weka ReplaceMissingValues Data Filter.

- 2. Click the *Apply* button to apply the filter to your dataset.
- 3. Click *mass* in the *Attributes* section and review the details of the *Selected attribute*.

Notice that the 11 attribute values that were marked *Missing* have been set to the mean value of the distribution.

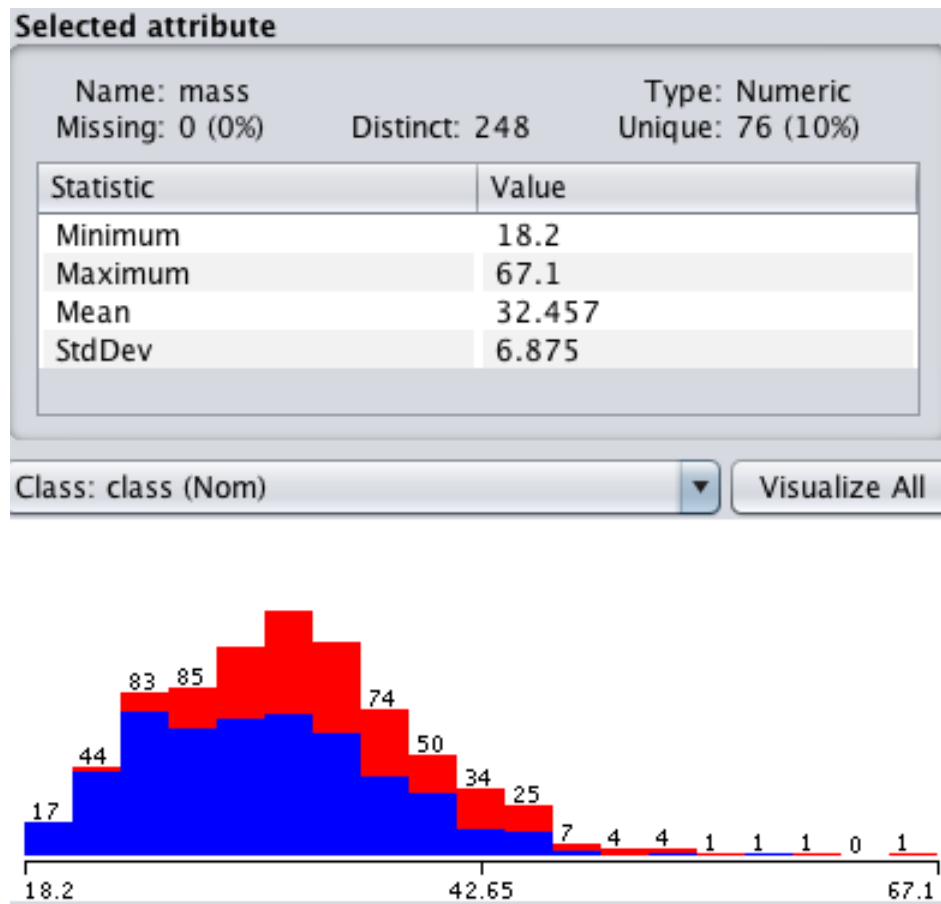


Figure 12.6: Weka Imputed Values.

12.4 Summary

In this lesson you discovered how you can handle missing data in your machine learning dataset using Weka. Specifically, you learned:

- How to mark corrupt values as missing in your dataset.
- How to remove instances with missing values from your dataset.
- How to impute mean values for missing values in your dataset.

12.4.1 Next

Raw data may have many attributes, but not all may be relevant to a model in order to make predictions. In the next lesson you will learn how you can use feature selection to identify only those most relevant features to the output attribute.

Chapter 22

How to Save Your Machine Learning Model and Make Predictions

After you have found a well performing machine learning model and tuned it, you must finalize your model so that you can use it to make predictions on new data. In this lesson you will discover how to finalize your machine learning model, save it to file and load it later in order to make predictions on new data. After reading this lesson you will know:

- How to train a final version of your machine learning model in Weka.
- How to save your finalized model to file.
- How to load your finalized model later and use it to make predictions on new data.

Let's get started.

22.1 Tutorial Overview

This tutorial is broken down into 4 parts:

1. Finalize Model where you will discover how to train a finalized version of your model.
2. Save Model where you will discover how to save a model to file.
3. Load Model where you will discover how to load a model from file.
4. Make Predictions where you will discover how to make predictions for new data.

The tutorial provides a template that you can use to finalize your own machine learning algorithms on your data problems. We are going to use the Pima Indians Onset of Diabetes dataset. You can learn more about this dataset in [Section 8.2.1](#). We are going to finalize a logistic regression model on this dataset, both because it is a simple algorithm that is well understood and because it does very well on this problem.

22.2 Finalize a Machine Learning Model

Perhaps the most neglected task in a machine learning project is how to finalize your model. Once you have gone through all of the effort to prepare your data, compare algorithms and tune them on your problem, you actually need to create the final model that you intend to use to make new predictions. Finalizing a model involves training the model on the entire training dataset that you have available.

- 1. Open the *Weka GUI Chooser*.
- 2. Click the *Explorer* button to open the *Weka Explorer* interface.
- 3. Load the Pima Indians onset of diabetes dataset from the `data/diabetes.arff` file.

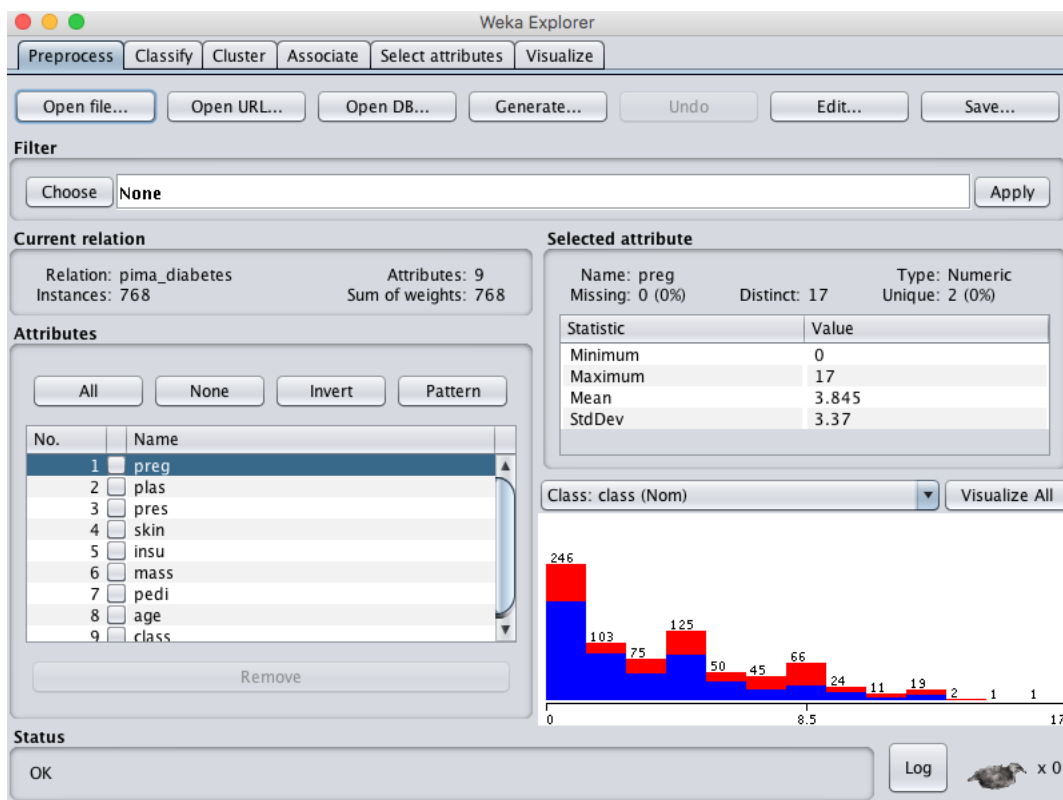


Figure 22.1: Weka Load Pima Indians Onset of Diabetes Dataset.

- 4. Click the *Classify* tab to open up the classifiers.
- 5. Click the *Choose* button and choose *Logistic* under the *functions* group.
- 6. Select *Use training set* under *Test options*.
- 7. Click the *Start* button.

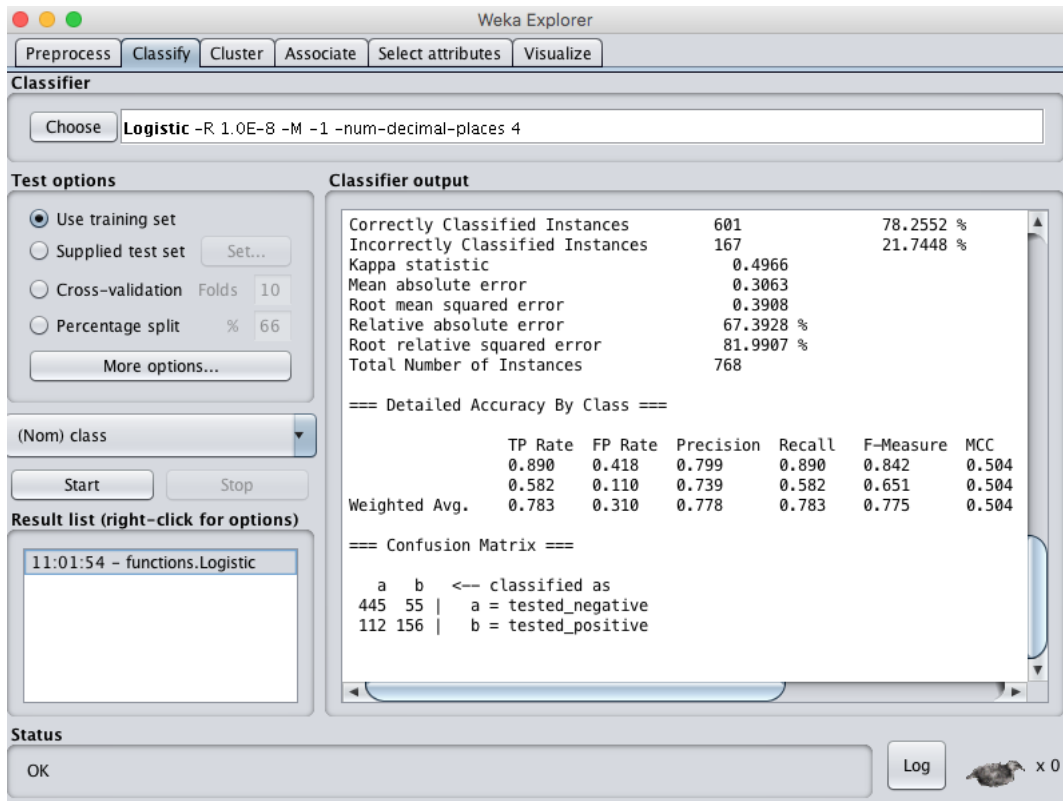


Figure 22.2: Weka Train Logistic Regression Model.

This will train the chosen Logistic regression algorithm on the entire loaded dataset. It will also evaluate the model on the entire dataset, but we are not interested in this evaluation. It is assumed that you have already estimated the performance of the model on unseen data using cross validation as a part of selecting the algorithm you wish to finalize. It is this estimate you prepared previously that you can report when you need to inform others about the skill of your model. Now that we have finalized the model, we need to save it to file.

22.3 Save Finalized Model To File

Continuing on from the previous section, we need to save the finalized model to a file on your disk. This is so that we can load it up at a later time, or even on a different computer in the future and use it to make predictions. We won't need the training data in the future, just the model of that data. You can easily save a trained model to file in the *Weka Explorer* interface.

- 1. Right click on the result item for your model in the *Result list* on the *Classify* tab.
- 2. Click *Save model* from the right click menu.

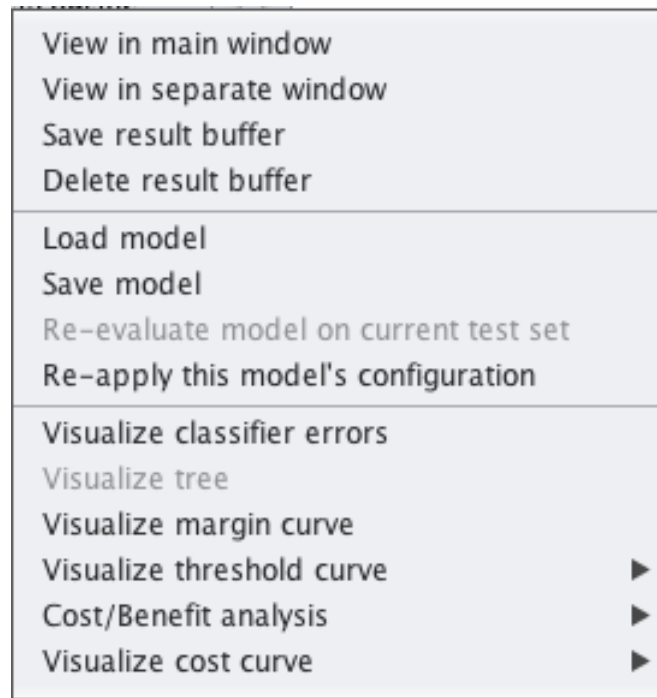


Figure 22.3: Weka Save Model to File.

- 3. Select a location and enter a filename such as *logistic*, click the *Save button*.

Your model is now saved to the file `logistic.model`. It is in a binary format (not text) that can be read again by the Weka platform. As such, it is a good idea to note down the version of Weka you used to create the model file, just in case you need the same version of Weka in the future to load the model and make predictions. Generally, this will not be a problem, but it is a good safety precaution. You can close the *Weka Explorer* now. The next step is to discover how to load up the saved model.

22.4 Load a Finalized Model

You can load saved Weka models from file. The *Weka Explorer* interface makes this easy.

- 1. Open the *Weka GUI Chooser*.
- 2. Click the *Explorer* button to open the *Weka Explorer* interface.
- 3. Load any old dataset, it does not matter. We will not be using it, we just need to load a dataset to get access to the *Classify* tab. If you are unsure, load the `data/diabetes.arff` file again.
- 4. Click the *Classify* tab to open up the classifiers.
- 5. Right click on the *Result list* and click *Load model*, select the model saved in the previous section *logistic.model*.

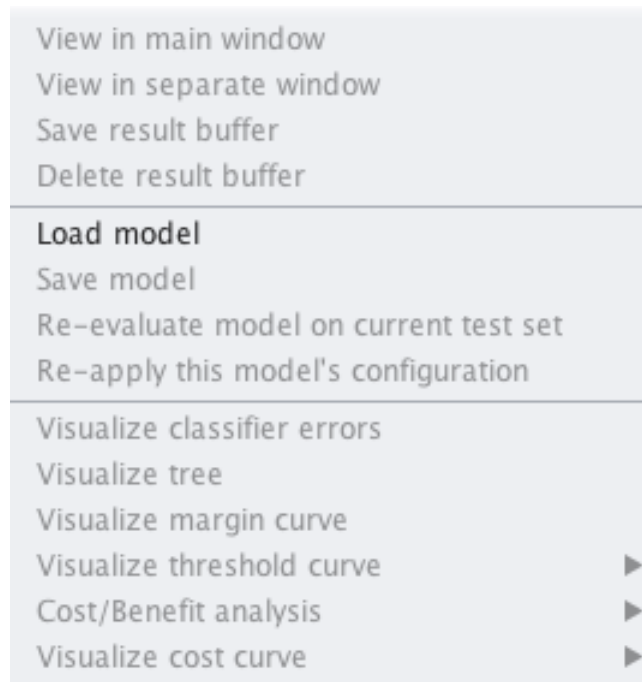


Figure 22.4: Weka Load Model From File.

The model will now be loaded into the *Weka Explorer*. We can now use the loaded model to make predictions for new data.

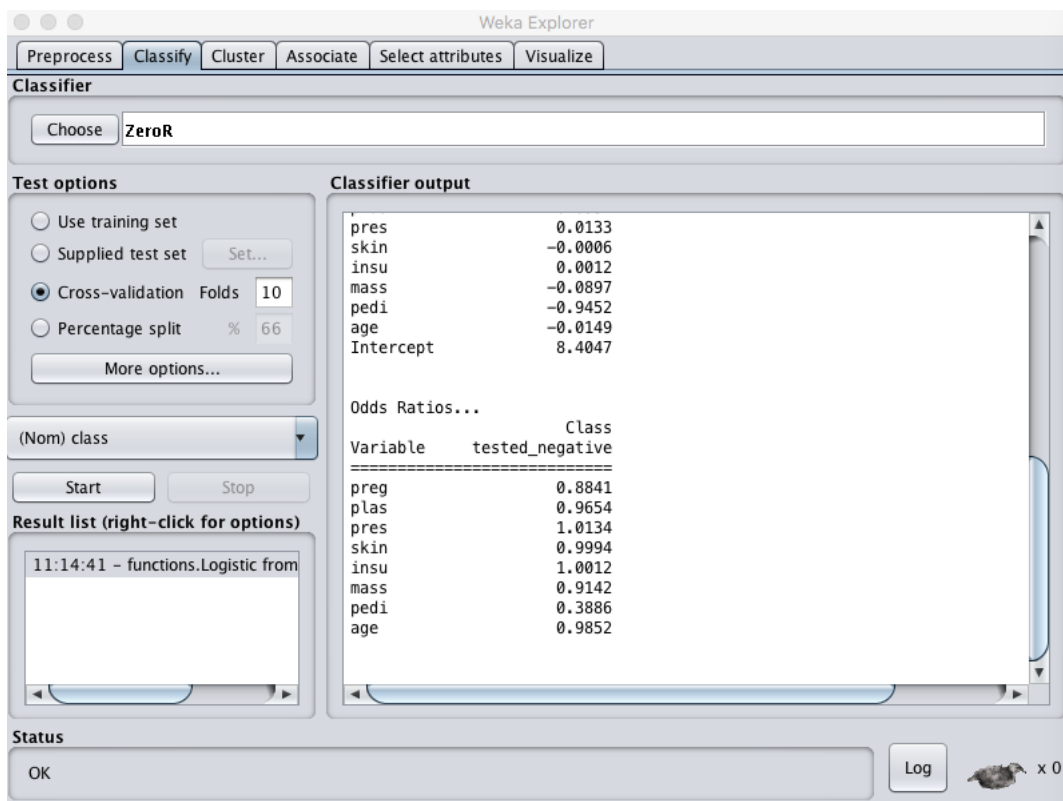


Figure 22.5: Weka Model Loaded From File Ready For Use.

22.5 Make Predictions on New Data

We can now make predictions on new data. First, let's create some pretend new data. Make a copy of the file `data/diabetes.arff` and save it as `data/diabetes-new-data.arff`.

- Open the file in a text editor.
- Find the start of the actual data in the file with the `@data` on line 95.
- We only want to keep 5 records. Move down 5 lines, then delete all the remaining lines of the file.

The class value (output variable) that we want to predict is on the end of each line. Delete each of the 5 output variables and replace them with question mark symbols (?).

```

85  @relation pima_diabetes
86  @attribute 'preg' numeric
87  @attribute 'plas' numeric
88  @attribute 'pres' numeric
89  @attribute 'skin' numeric
90  @attribute 'insu' numeric
91  @attribute 'mass' numeric
92  @attribute 'pedi' numeric
93  @attribute 'age' numeric
94  @attribute 'class' { tested_negative, tested_positive}
95  @data
96  6,148,72,35,0,33.6,0.627,50,?
97  1,85,66,29,0,26.6,0.351,31,?
98  8,183,64,0,0,23.3,0.672,32,?
99  1,89,66,23,94,28.1,0.167,21,?
100 0,137,40,35,168,43.1,2.288,33,?
101

```

Figure 22.6: Weka Dataset For Making New Predictions.

We now have *unseen* data with no known output for which we would like to make predictions. Continue on from the previous part of the tutorial where we already have the model loaded.

- 1. On the *Classify* tab, select the *Supplied test set* option in the *Test options* pane.

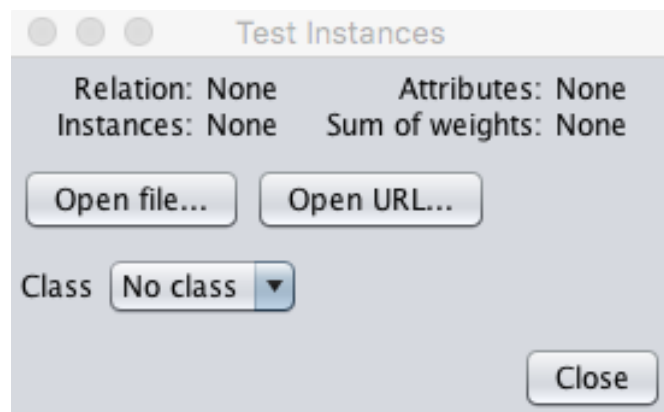


Figure 22.7: Weka Select New Dataset On Which To Make New Predictions.

- 2. Click the *Set* button, click the *Open file* button on the options window and select the mock new dataset we just created with the name *diabetes-new-data.arff*. Click *Close* on the window.
- 3. Click the *More options...* button to bring up options for evaluating the classifier.
- 4. Uncheck the information we are not interested in, specifically:
 - *Output model*
 - *Output per-class stats*
 - *Output confusion matrix*
 - *Store predictions for visualization*

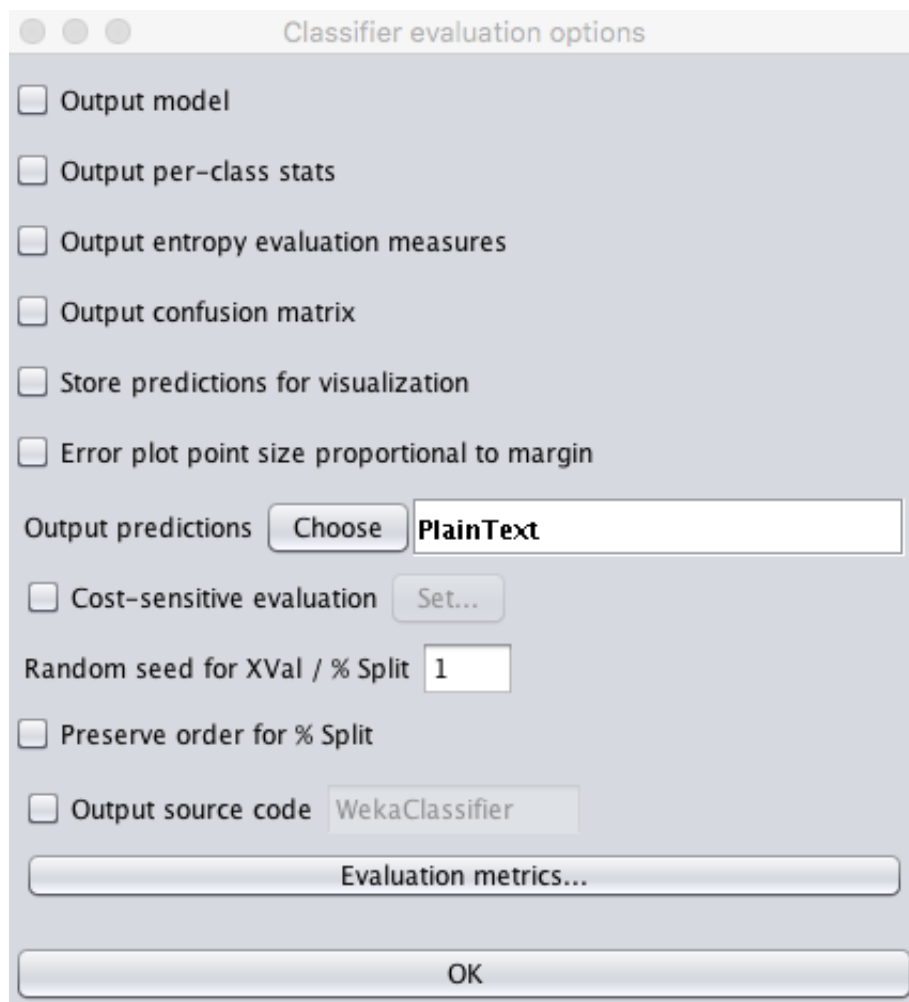


Figure 22.8: Weka Customized Test Options For Making Predictions.

- 5. For the *Output predictions* option click the *Choose* button and select *PlainText*.

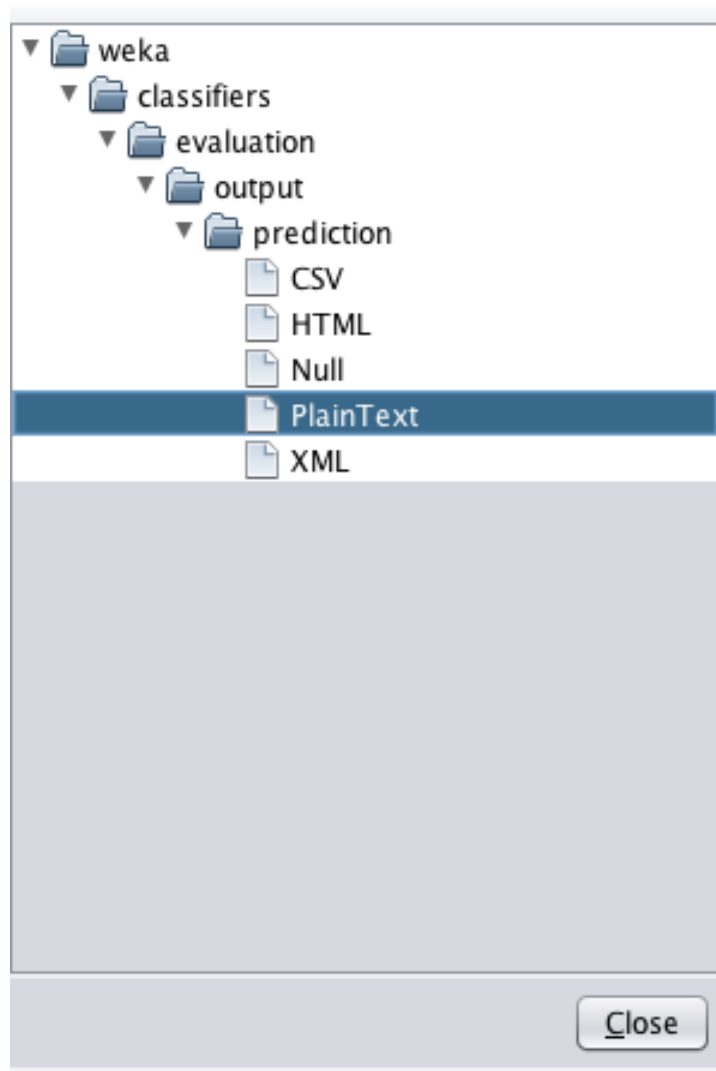


Figure 22.9: Weka Output Predictions in Plain Text Format.

- 6. Click the *OK* button to confirm the Classifier evaluation options.
- 7. Right click on the list item for your loaded model in the *Results list* pane.
- 8. Select *Re-evaluate model on current test set*.

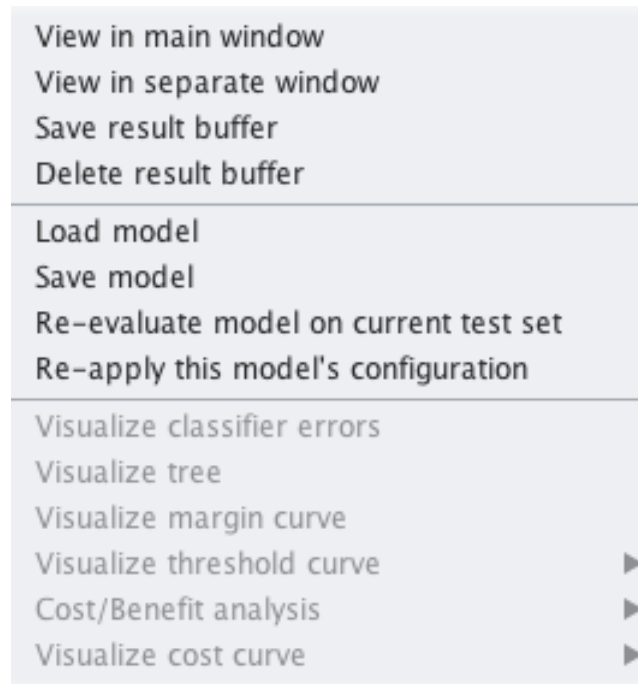


Figure 22.10: Weka Reevaluate Loaded Model On Test Data And Make Predictions.

The predictions for each test instance are then listed in the *Classifier Output* pane. Specifically the middle column of the results with predictions like *tested_positive* and *tested_negative*. You could choose another output format for the predictions, such as CSV, that you could later load into a spreadsheet like Excel. For example, below is an example of the same predictions in CSV format.

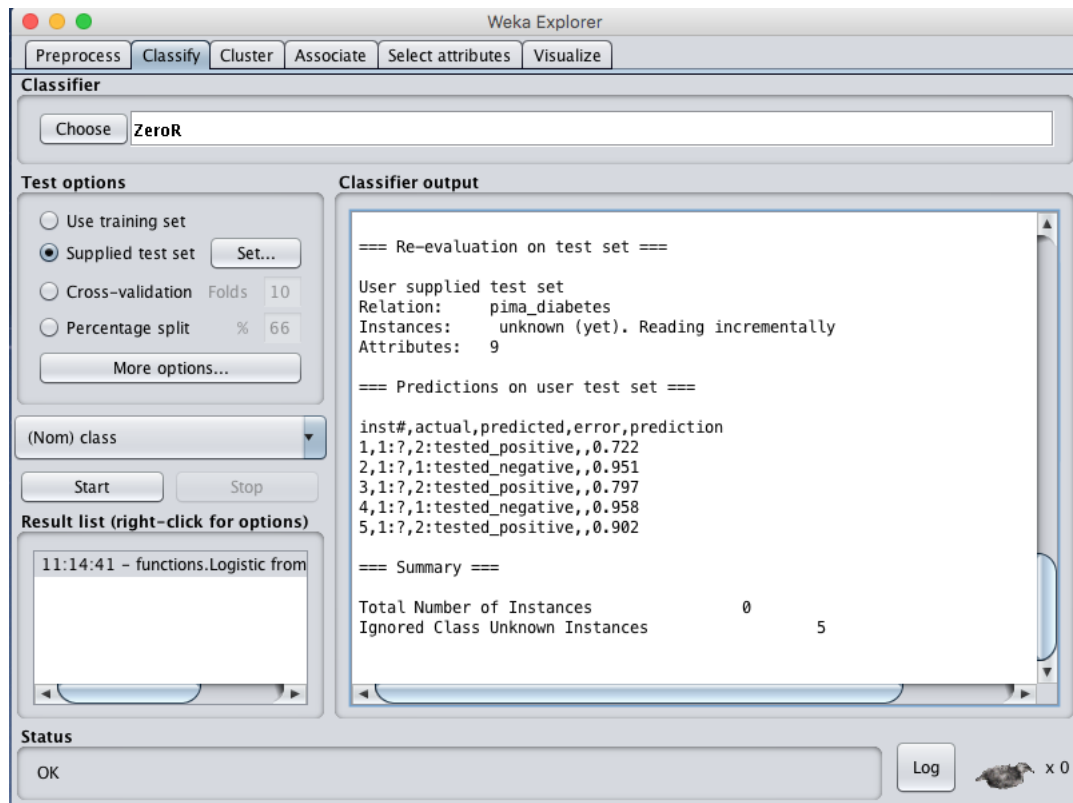


Figure 22.11: Weka Predictions Made on New Data By a Loaded Model.

22.6 Summary

In this lesson you discovered how to finalize your model and make predictions for new unseen data. You can see how you can use this process to make predictions on new data yourself. Specifically, you learned:

- How to train a final instance of your machine learning model.
- How to save a finalized model to file for later use.
- How to load a model from file and use it to make predictions on new data.

22.6.1 Next

This was the last lesson and concludes Part II of this book. Next in Part III you will take on your first end-to-end machine learning project on a multiclass classification problem.