**Data Transformation - Scale and Reduce**

**Scale and Reduce**

For machine learning, common data tasks include clipping, binning, and normalizing numerical values. Other modules support dimensionality reduction.

**Modeling numerical data**

Tasks such as normalizing, binning, or redistributing numerical variables are an important part of data preparation for machine learning. The modules in this group support the following data preparation tasks:

* Grouping data into bins of varying sizes or distributions.
* Removing outliers or changing their values.
* Normalizing a set of numeric values into a specific range.
* Creating a compact set of feature columns from a high-dimension dataset.

**Clip Values**

To identify and optionally replace data values that are above or below a specified threshold. This is useful when you want to remove outliers or replace them with a mean, a constant, or other substitute value.

You connect the module to a dataset that has the numbers you want to clip, choose the columns to work with, and then set a threshold or range of values, and a replacement method. The module can output either just the results or the changed values appended to the original dataset.

**How to configure Clip Values**

Before you begin, identify the columns you want to clip, and the method to use. We recommend that you test any clipping method on a small subset of data first. The module applies the same criteria and replacement method to all columns that you include in the selection. Therefore, be sure to exclude columns that you don't want to change.

If you need to apply clipping methods or different criteria to some columns, you must use a new instance of Clip Values for each set of similar columns.

1. Add the Clip Values module to your experiment and connect it to the dataset you want to modify. You can find this module under Data Transformation, in the Scale and Reduce category.
2. In the List of columns, use the Column Selector to choose the columns to which Clip Values will be applied.
3. For a Set of thresholds, choose one of the following options from the dropdown list. These options determine how you set the upper and lower boundaries for acceptable values vs. values that must be clipped.

* ***ClipPeaks:*** When you clip values by peaks, you specify only an upper boundary. Values greater than that boundary value are replaced or removed.
* ***ClipSubpeaks:*** When you clip values by sub-peaks, you specify only a lower boundary. Values that are less than that boundary value are replaced or removed.
* ***ClipPeaksAndSubpeaks:*** When you clip values by peaks and sub-peaks, you can specify both the upper and lower boundaries. Values that are outside that range are replaced or removed. Values that match the boundary values are not changed.

1. Depending on your selection in the preceding step, you can set the following threshold values:

* **Lower threshold:** Displayed only if you choose ClipSubPeaks
* **Upper threshold:** Displayed only if you choose ClipPeaks
* **Threshold**: Displayed only if you choose ClipPeaksAndSubPeaks

For each threshold type, choose either Constant or Percentile.

1. If you select Constant, type the maximum or minimum value in the text box. For example, assume that you know the value 999 was used as a placeholder value. You could choose Constant for the upper threshold, and type 999 in Constant value of upper threshold.
2. If you choose Percentile, you constrain the column values to a percentile range.

For example, assume you want to keep only the values in the 10-80 percentile range, and replace all others. You would choose Percentile, and then type 10 for Percentile value of lower threshold, and type 80 for Percentile value of upper threshold.

1. Define a substitute value.

Numbers that exactly match the boundaries you just specified are considered to be inside the allowed range of values, and thus are not replaced or removed. All numbers that fall outside the specified range are replaced with the substitute value.

* Substitute value for peaks: Defines the value to substitute for all column values that are greater than the specified threshold.
* Substitute value for subpeaks: Defines the value to use as a substitute for all column values that are less than the specified threshold.
* If you use the ClipPeaksAndSubpeaks option, you can specify separate replacement values for the upper and lower clipped values.

The following replacement values are supported:

* **Threshold:** Replaces clipped values with the specified threshold value.
* **Mean:** Replaces clipped values with the mean of the column values. The mean is computed before values are clipped.
* **Median:** Replaces clipped values with the median of the column values. The median is computed before values are clipped.
* **Missing:** Replaces clipped values with the missing (empty) value.

1. Add indicator columns: Select this option if you want to generate a new column that tells you whether or not the specified clipping operation applied to the data in that row. This option is particularly handy when you are testing a new set of clipping and substitution values.
2. Overwrite flag: Indicate how you want the new values to be generated. By default, Clip Values constructs a new column with the peak values clipped to the desired threshold. New values overwrite the original column.To keep the original column and add a new column with the clipped values, deselect this option.
3. Run the experiment.

Right-click the output of the Clip Values module and select Visualize to review the values and make sure the clipping operation met your expectations.

**Clipping using percentiles**

To understand how clipping by percentiles works, consider a dataset with 10 rows, which have one instance each of the values 1-10.

* If you are using percentile as the upper threshold, at the value for the 90th percentile, 90 percent of all values in the dataset must be less than that value.
* If you are using percentile as the lower threshold, at the value for the 10th percentile, 10 percent of all values in the dataset must be less than that value.

1. For Set of thresholds, choose ClipPeaksAndSubPeaks.
2. For Upper threshold, choose Percentile, and for Percentile number, type 90.
3. For Upper substitute value, choose Missing Value.
4. For Lower threshold, choose Percentile, and for Percentile number, type 10.
5. For Lower substitute value, choose Missing Value.
6. Deselect the option Overwrite flag, and select the option, Add indicator column.

Now try the same experiment using 60 as the upper percentile threshold and 30 as the lower percentile threshold, and use the threshold value as the replacement value. The following table compares these two results:

* Replace with missing; Upper threshold = 90; Lower threshold = 10
* Replace with threshold; Upper percentile = 60; Lower percentile = 30

**Group Data into Bins**

To group numbers or change the distribution of continuous data.

The Group Data into Bins module supports multiple options for binning data. You can customize how the bin edges are set and how values are apportioned into the bins. For example, you can:

* Manually type a series of values to serve as the bin boundaries.
* Calculate entropy scores to determine an information value for each range, to optimize the bins in the predictive model. + Assign values to bins by using quantiles, or percentile ranks.
* Control the number of values in each bin can also be controlled.
* Force an even distribution of values into the bins.

Binning and Grouping

Binning or grouping data (sometimes called quantization) is an important tool in preparing numerical data for machine learning, and is useful in scenarios like these:

* A column of continuous numbers has too many unique values to model effectively, so you automatically or manually assign the values to groups, to create a smaller set of discrete ranges. For example, you could use entropy scores generated by Group Data into Bins to identify the optimal groupings of data values and use those groups as features in your model.
* Replace a column of numbers with categorical values that represent specific ranges. For example, you might want to group values in an age column by specifying custom ranges, such as 1-15, 16-22, 23-30, and so forth for user demographics.
* A dataset has a few extreme values, all well outside the expected range, and these values have an outsized influence on the trained model. To mitigate the bias in the model, you might transform the data to a uniform distribution, using the quantiles (or equal-height) method. With this method, the Group Data into Bins module determines the ideal bin locations and bin widths to ensure that approximately the same number of samples fall into each bin. Then, depending on the normalization method you choose, the values in the bins are either transformed either to percentiles or mapped to a bin number.

**How to configure Group Data into Bins**

1. Add the Group Data Into Bins module to your experiment in Studio (classic). You can find this module in the category Data Transformation, under Scale and Reduce.
2. Connect the dataset that has numerical data to the bin. Quantization can be applied only to columns containing numeric data. If the dataset contains non-numeric columns, use the Select Columns in the Dataset module to select a subset of columns to work with.
3. Specify the binning mode. The binning mode determines other parameters so be sure to select the Binning mode option first! The following types of binning are supported:

***Entropy MDL:*** This method requires that you select the column you want to predict and the column or columns that you want to group into bins. It then makes a pass over the data and attempts to determine the number of bins that minimizes the entropy. In other words, it chooses a number of bins that allows the data column to best predict the target column. It then returns the bin number associated with each row of your data in a column named <colname>quantized.

If the Entropy MDL method cannot find a way to initially bin the data to make a good prediction, it assigns all data to a uniform bin. This does not mean that the column is not a good predictor. In this case, you can use other methods to find the number of bins that would minimize entropy, and make the data a better predictor.This method does not return the actual entropy scores.

***Quantiles:*** The quantile method assigns values to bins based on percentile ranks. Quantiles are also known as equal height binning.

***Equal Width***: With this option, you must specify the total number of bins. The values from the data column are placed in the bins such that each bin has the same interval between starting and end values. As a result, some bins might have more values if data is clumped around a certain point.

***Custom Edges***: You can specify the values that begin each bin. The edge value is always the lower boundary of the bin. For example, assume you want to group values into two bins, one with values greater than 0, and one with values less than or equal to 0. In this case, for bin edges, you would type 0 in a Comma-separated list of bin edges. The output of the module would be 1 and 2, indicating the bin index for each row value.

***Equal Width with Custom Start and Stop***: This method is like the Equal Width option, but you can specify both lower and upper bin boundaries.

1. Number of bins: If you are using the Entropy MDL, Quantiles, and Equal Width binning modes, use this option to specify how many bins, or quantiles, that you want to create.
2. For Columns to bin, use the Column Selector to choose the columns that have the values you want to bin. Columns must be a numeric data type.

The same binning rule is applied to all applicable columns that you choose. Therefore, if you need to bin some columns by using a different method, use a separate instance of Group Data into Bins for each set of columns.

1. For Output mode, indicate how you want to output the quantized values.

* **Append:** Creates a new column with the binned values and appends that to the input table.
* **In place:** Replace the original values with the new values in the dataset.
* **ResultOnly:** Returns just the result columns.

1. If you select the Quantiles binning mode, use the Quantile normalization option to determine how values are normalized prior to sorting into quantiles. Note that normalizing values transforms the values, but does not affect the final number of bins. For an example, see Effects of Different Normalization Methods.

The following normalization types are supported:

* ***Percent:*** Values are normalized within the range [0,100]
* ***PQuantile:*** Values are normalized within the range [0,1]
* ***QuantileIndex:*** Values are normalized within the range [1,number of bins]

1. If you choose the Custom Edges option, type a comma-separated list of numbers to use as bin edges in the + Comma-separated list of bin edges text box. The values mark the point that divides bins, Therefore, if you type one bin edge value, two bins will be generated; if you type two bin edge values, three bins will be generated, and so forth. The values must be sorted in order that the bins are created, from lowest to highest.
2. If you use the option, Equal Width With Custom Start And Stop, you must specify the boundaries of the bins.

* Define the lower boundary of the first bin by typing a value in the First edge position text box.
* Define the lower boundary of the last bin by typing a value in the Last edge position text box.

1. Tag columns as categorical: Select this option to automatically add a metadata flag to the column of binned values. The metadata flag indicates that the quantized columns should be handled as categorical variables.
2. Run the experiment, or select this module and click Run selected.

**Results**

The Group Data into Bins module returns a dataset in which each element has been binned according to the specified mode.

It also returns a Binning transformation, which is a function that can be passed to the Apply Transformation module to bin new samples of data using the same binning mode and parameters.

To see how well the binning method functions as a predictor, you can click the dataset output from Group Data to Bins, and compare the label column to the binned column. If the grouping to bins is predictive, the values in the cross-tab matrix should concentrate in a few cells.

**Normalize Data**

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information. Normalization is also required for some algorithms to model the data correctly.

For example, assume your input dataset contains one column with values ranging from 0 to 1, and another column with values ranging from 10,000 to 100,000. The great difference in the scale of the numbers could cause problems when you attempt to combine the values as features during modeling.

Normalization avoids these problems by creating new values that maintain the general distribution and ratios in the source data, while keeping values within a scale applied across all numeric columns used in the model.

This module offers several options for transforming numeric data:

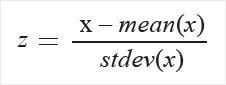
* You can change all values to a 0-1 scale, or transform the values by representing them as percentile ranks rather than absolute values.
* You can apply normalization to a single column, or to multiple columns in the same dataset.
* If you need to repeat the experiment, or apply the same normalization steps to other data, you can save the steps as a normalization transform, and apply it to other datasets that have the same schema.

**How to configure Normalize Data**

You can apply only one normalization method at a time using this module. Therefore, the same normalization method is applied to all columns that you select. To use different normalization methods, use a second instance of **Normalize Data**.

1. Add the **Normalize Data** module to your experiment. You can find the module in Machine Learning Studio (classic), under **Data Transformation**, in the **Scale and Reduce** category.
2. Connect a dataset that contains at least one column of all numbers.
3. Use the Column Selector to choose the numeric columns to normalize. If you don't choose individual columns, by default **all** numeric type columns in the input are included, and the same normalization process is applied to all selected columns.This can lead to strange results if you include numeric columns that shouldn't be normalized! Always check the columns carefully. If no numeric columns are detected, check the column metadata to verify that the data type of the column is a supported numeric type.
4. **Use 0 for constant columns when checked**: Select this option when any numeric column contains a single unchanging value. This ensures that such columns are not used in normalization operations.
5. From the **Transformation method** dropdown list, choose a single mathematical functions to apply to all selected columns.
   * **Zscore**: Converts all values to a z-score.

The values in the column are transformed using the following formula:

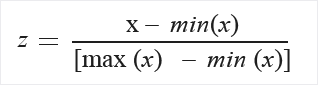


Mean and standard deviation are computed for each column separately. Population standard deviation is used.

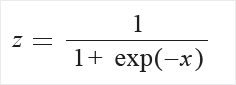
* + **MinMax**: The min-max normalizer linearly rescales every feature to the [0,1] interval.

Rescaling to the [0,1] interval is done by shifting the values of each feature so that the minimal value is 0, and then dividing by the new maximal value (which is the difference between the original maximal and minimal values).

The values in the column are transformed using the following formula:



* + **Logistic**: The values in the column are transformed using the following formula:



* + **LogNormal**: This option converts all values to a lognormal scale.

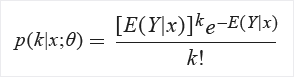
The values in the column are transformed using the following formula:

formula log-normal distribution

Here μ and σ are the parameters of the distribution, computed empirically from the data as maximum likelihood estimates, for each column separately.

* + **TanH**: All values are converted to a hyperbolic tangent.

The values in the column are transformed using the following formula:



1. Run the experiment, or double-click the **Normalize Data** module and select **Run Selected**.

**Results**

The Normalize Data module generates two outputs:

* To view the transformed values, right-click the module, select Transformed dataset, and click Visualize.

By default, values are transformed in place. If you want to compare the transformed values to the original values, use the Add Columns module to recombine the datasets and view the columns side-by-side.

* To save the transformation so that you can apply the same normalization method to another similar dataset, right-click the module, select Transformation function, and click Save as Transform.

You can then load the saved transformations from the Transforms group of the left navigation pane and apply it to a dataset with the same schema by using Apply Transformation.

**Algorithms that apply normalization**

Normalizing features so that they use a common scale is a general requirement for many machine learning algorithms.

* In linear classification algorithms, instances are viewed as vectors in multi-dimensional space. Since the range of values of raw data varies widely, some objective functions do not work properly without normalization. For example, if one of the features has a broad range of values, the distances between points is governed by this particular feature.

Therefore, numeric features should be normalized so that each feature contributes approximately proportionately to the final distance. This can provide significant speedup and accuracy benefits.

* When using the Logistic Regression and Averaged Perceptron algorithms, by default, features are normalized before training.

**Principal Component Analysis**

This describes how to use the **Principal Component Analysis** module in Machine Learning Studio (classic) to reduce the dimensionality of your training data. The module analyzes your data and creates a reduced feature set that captures all the information contained in the dataset, but in a smaller number of features.

The module also creates a transformation that you can apply to new data, to achieve a similar reduction in dimensionality and compression of features, without requiring additional training.

**Principal Component Analysis**

Principal Component Analysis (PCA) is a popular technique in machine learning. It relies on the fact that many types of vector-space data are compressible, and that compression can be most efficiently achieved by sampling.

Added benefits of PCA are improved data visualization, and optimization of resource use by the learning algorithm.

The **Principal Component Analysis** module in Machine Learning Studio (classic) takes a set of feature columns in the provided dataset, and creates a projection of the feature space that has lower dimensionality. The algorithm uses randomization techniques to identify a feature subspace that captures most of the information in the complete feature matrix. Hence, the transformed data matrices capture the variance in the original data while reducing the effect of noise and minimizing the risk of overfitting.

**How to configure Principal Component Analysis**

1. Add the **Principal Component Analysis** module to your experiment. You can find it in under **Data Transformation**, in the **Scale and Reduce** category.
2. Connect the dataset you want to transform, and choose the feature columns to analyze.

If it is not already clear which columns are features and which are labels, we recommend that you use the Edit Metadata module to mark the columns in advance.

1. **Number of dimensions to reduce to**: Type the desired number of columns in the final output. Each column represents a dimension capturing some part of the information in the input columns.

For example, if the source dataset has eight columns and you type 3, three new columns are returned that capture the information of the eight selected columns. The columns are named Col1, Col2, and Col3. These columns do not map directly to the source columns; instead, the columns contain an approximation of the feature space described by the original columns 1-8.

1. **Normalize dense dataset to zero mean**: Select this option if the dataset is dense, meaning it contains few missing values. If selected, the module normalizes the values in the columns to a mean of zero before any other processing.

For sparse datasets, this option should not be selected. If a sparse dataset is detected, the parameter is overridden.

1. Run the experiment.

**Results**

The module outputs a reduced set of columns that you can use in creating a model. You can save the output as a new dataset or use it in your experiment.