Machine Learning and Data Types & Formats

Machine Learning is an area of data science that involves analyzing data for discover or classification. Machine Learning methods can be **supervised**, such as Naïve Bayes, Support Vector Machines, Decision Trees, Random Forest, Regression, Neural Nets, or can be **unsupervised**, such as Clustering, Association Rule Mining, etc.

Supervised methods are generally used to build a model with known and labeled data. Then, that model is used to predict the class or category for new data with unknown labels. When building a supervised model, such as a Decision Tree, the model must first be **trained** with labeled data (data for which each row or vector is part of a known category or group), and then **tested** with unlabeled data. The goal is to correctly predict the labels of the unlabeled data using the model.

In general, supervised learning methods are used for prediction (also called classification).

Unsupervised methods use unlabeled data and are **discovery-based.** Unlike supervised methods, unsupervised methods do not train or test. Instead they evaluate and discover. For example, k-means clustering using a **distance metric** to measure the similarity between each row (vector) of data. The distance between data rows will determine which "cluster" it will be placed into. So clustering can be used to **discover** groups, categories, and/or similarities within a dataset.

As an example, ***Naïve Bayes is a supervised method that uses labeled data to train a probability-based prediction model***. This model can then be used to predict/classify data vectors for which the label is not known.

Before jumping into Data Analytics or Machine Learning, the above sentence needs to make sense.

What is a **data vector**?

What is the difference between **labeled and unlabeled data**?

Which data types are required for different models or methods?

Why must you have labeled data to perform supervised methods?

What does it mean to classify (predict) a label from a new data row (vector) using a trained model?

What is the difference between supervised and unsupervised learning?

Let's use a small, pretend, and simple clean dataset to answer these questions.

name	class	feathers	limbs	livebirth	canfly	height
chicken	BIRD	1	2	no	yes	27.6
salmon	FISH	0	0	no	no	40.1
whale	MAMMAL	0	4	yes	no	165.34
bat	MAMMAL	0	4	yes	yes	30.6
goose	BIRD	1	2	no	yes	62.2

pigeon	BIRD	1	2	no	yes	4.9
guppy	FISH	0	0	yes	no	4.5
cat	MAMMAL	0	4	yes	no	9.55

This is **record data**. It is made up of **rows** (vectors) where each row gives values for a specific species and **columns** that represent the **variables (also called attributes, fields, or features)** .

For example, the very top row of this dataset contains all variable (feature) names such as “name”, “class”, “feathers”, etc. All following rows contain the data.

A **data vector** is the same as a data row. For example, the first row or vector of data here is:

chicken	BIRD	1	2	no	yes	27.6

It is important to understand that data science and analytics (and related) **use many different words to say the same thing**. Why? The reason is because data science was born from many different areas and is used in many different areas.

So, words like **row, vector, observation, instance, person, etc. can all mean the same thing**.

Similarly, words like **column, name, variable, attribute, feature, dimension, etc. can all mean the same thing**.

To avoid excessive writing, I will use **vector** to describe rows and **variable** to describe columns.

Now we know that this dataset contains 8 vectors (rows) of data and 7 variables (columns).

Our next important task is to make sure that we understand the **type** of each variable. A variable in a dataset can be qualitative or quantitative. I can be discrete or continuous. I can be nominal, ordinal, or categorical. It can be a **label**.

Why does this matter? It matters enormously because not all methods and not all programming libraries for methods will use the same data types. For example, the machine learning modeling method called Support Vector Machines (SVMs) only works on numeric (quantitative) data.

It is critical to understand the nature of your data, if and how you will need to reformat it to use various models and methods, which column is the label (if you have labeled data), and which R or Python libraries expect which data types.

This process is often the difference between being able to copy someone else’s code on a canned (pretend) dataset versus being able to perform true ML on real-world data. But more on that later.

Let’s look at our dataset here. The following is true - make sure you agree!

1) This is mixed data. That means that some variables are numeric (quantitative) and some variables are qualitative.
2) This **is** labeled data and the label is called, "class". The label is categorical data (called factor in R) and there are three categories in this case: BIRD, FISH, MAMMAL.
3) The **label** is not actually part of the data. **This is important.** The label is the category that each data vector fits into.

The following data vector:

whale	MAMMAL	0	4	yes	no	165.34

Should actually be viewed as

whale	0	4	yes	no	165.34

With label: MAMMAL

You will find that ***sklearn*** in Python requires you to remove (separate) the label from the data before performing ML modeling (training and testing).

It is critical to understand the difference between the label (which is the category or class or group that the row is a member of) and the data (which describes the row).

Next, let's start with the first variable in the dataset which is "name".

name
chicken
salmon
whale
bat
goose
pigeon
guppy
cat

While the values for **name**, such as chicken, salmon, etc. are useful to humans, they are useless to computers. They are also BAD for modeling! Machine learning modeling - whether it be Naïve Bayes, Decision Trees, Support Vector Machines, etc. looks for **patterns** in the data that can be modeled. If there are no patterns of any kind, one cannot use the data to build a predictive model.

Think about that for a second. Why can **we** (or computers) make predictions? What is a prediction? A prediction is an outcome that we expect to be true based on known knowledge or information. For example, I can predict within +/- 2 minutes how long a new type of cookie will take to bake based on the attributes of the cookie and my experience (my personal model) for baking cookies.

Similarly, one can attempt to look at attributes (variables) of the stock market to try to predict is a stock will exceed a price point on the following day.

Humans and computers both require information with **patterns** to make predictions.

If our data were 100% chaotic, there would be no way to use it to predict anything!

Therefore, variables with patterns help us and variables that have no real patters hinder us. Here, we have 8 different species names - all different. Therefore, none of them are more or less related to anything. If you are still skeptical, think about this example :

name
Bob
Sally
Fred
Bo
Byron
Peg
Fran
Harry

If you are training a model (or person) to be able to tell which of these people is predicted to major in Engineering, will the names help?

How about this data instead?

Major
History
Math
Statistics
History
Statistics
Statistics
History
Math

While there are always exceptions and this is only one dimensional data, we can likely see a **pattern** that STEM majors are more likely to enter into Engineering.

Some variables are very helpful with modeling and prediction. These are often called **important features** and R and Python have method for extracting such important features. Other variables are not only useless, but they can confuse a model if used as part of the training.

Therefore, to use this data for ML modeling, we now must take two steps:

1) Separate the label from the data. Keep both.
2) Remove the column called “name”. (You can keep in in a separate dataframe for reference)

This leaves us with:

THE LABELS

class
BIRD
FISH
MAMMAL
MAMMAL
BIRD
BIRD
FISH
MAMMAL

THE DATA

feathers	limbs	livebirth	canfly	height
1	2	no	yes	27.6
0	0	no	no	40.1
0	4	yes	no	165.34
0	4	yes	yes	30.6
1	2	no	yes	62.2
1	2	no	yes	4.9
0	0	yes	no	4.5
0	4	yes	no	9.55

Next, let's look at all the data types.

The first variable is "feathers". This is categorical. Categorical data is qualitative data (non-numeric) such as each value is part of a category or group. For feathers, there are two groups: 0 (meaning none) or 1 (meaning yes the animal has feathers).

Do not let the 0 and 1 fool you! Just because we are using numbers as names, does not mean they are real numbers. Numbers are often repurposed as names - such as #15 Tim Tebow - Florida Gators! However, the number 15 is a naming method - it is not numeric. We cannot add two footballs player numbers together and get a different player 😊 There are many such examples, like Driver License numbers, phone numbers. SS#s, rankings, etc. **Never** confuse number-names with numeric data.

The next variable is "limbs". This one is quantitative. It gives that actual count of the number of limbs each animal has. Snakes have no limbs, birds have two (as far as I understand - I am not a Biologist 😊), humans have 4, spiders have 8, etc. Counts, speed, temperature, height, weight, etc. are all quantitative and numeric.

Why does this matter so much??

Because many methods, such as k-means clustering and SVMs require that the data is truly numeric. The reasons for this requirement is that both use mathematical calculations (like Euclidean distance or vector math) as part of the modeling process. Such math calculations cannot be done on qualitative data - you cannot find the Euclidean distance between Football Jersey 15 and Football Jersey 26.

The two variables, livebirth and canfly are all categorical. The variable, height, is quantitative.

This tells us a few things:

1) To use this dataset with SVM or clustering, or any other numeric method, we must remove all qualitative variables.

For Example, we can use this for SVM or k-means, where the label is highlighted and NOT part of the dataset. Let's look at this smaller dataset for a minute. This is numeric, labeled data. This is two dimensional data (it has two columns or variables). This data would likely NOT make a very accurate model for prediction between with only "limbs" and "height" there may not be enough patterned information to predict something. I might do OK with accuracy - perhaps 60%.

Try it! Can you predict the label (the "class") for this new data vector [2 85] ?

This data vector tells us that we have an animal with 2 limbs and that weighs 85 lbs.

Hmmm - we want to say MAMMAL because of the weight - but - the limbs suggest BIRD….

Interesting right! This is what ML methods do. They use all the data to try to find patterns. They then create a model using these patterns. Different ML methods use different methods and measures. Different programming languages have differing requirements and expected data formats.

class	limbs	height
BIRD	2	27.6
FISH	0	40.1
MAMMAL	4	165.34
MAMMAL	4	30.6
BIRD	2	62.2
BIRD	2	4.9
FISH	0	4.5
MAMMAL	4	9.55

Text Data

Let's look at one more idea before we close this tutorial. Not all data is record data! In fact, there are SO MANY formats for data such as sequential (DNA or AA data), image data, music data, text data, temporal data, spatial data, etc.

For our purposes, we will stick to text and record data. However, if you work in bio-informatics, as a biostatistician, or with a medical or public health group, you will certainly use sequential data - like RNA, DNA, protein data, etc. If you work in image recognition, image databases, etc, you will use image data.

Text data is data that starts out as words. We will only talk about English, but this applies to all languages and can get very complex if using languages with symbols like Hebrew, Mandarin, etc.

Text data can be novels, articles, news, Tweets, chats, emails, posts, reviews, text files, word docs, feedback, comments, webpages, and so on. Anything made up of **words** is text data.

We can perform all ML methods on text data! We just need to make sure it is in the **format** that we need it to be in. Formatting data is an important part of the cleaning and preparation process (as is normalization, feature engineering, etc - but more on that later).

Here are 3 pretend and very silly documents. Each is very short.

Document 1

People who own a dog tend to live longer. A Dog can bring happiness into your life.

A Dog will have their own bacterial biome that can help a person to improve their biome. A Dog is also loving, loyal, and great hiking or walking companions.

Document 2

Rescuing dog is wonderful, fulfilling, and healthy for you and your family. Dog can add love and companionship.

Never leave your dog in the car, ever. Dog need attention. Do not leave your dog home alone or locked in a box.

Think about how you would like to be treated and treat your dog the same way. Loving.

Document 3

Getting a pet, such as a dog, is a big decision. Loving. Dogs are like children. They need love, time, attention, and patience.

Dogs like to play and run. They are great exercise companions. Dogs need a lot of attention. Paying attention to your dog will offer you and your dog joy.

Right now, we cannot perform any ML or analytics on this data because it is not yet formatted.

The way we choose to format the data depends on the method we plan to use. For example, if I am planning to perform Association Rule Mining on this data, I would reformat this as **transaction format**. However, if I plan to use SVM or Naïve Bayes, I will **vectorize** this data.

I will not get into these details here as this will be in a different tutorial, but here is what **vectorized text data** looks like. This is a **normalized** example. Notice that the **words** are the variable names. Each row is a document (or tweet or novel or whatever). Usually, there are thousands of columns. Why? Because there are 1000s of possible words!

active	add	adults	allows	animal	anxiety
0	0	0	0	0	0
0	0.200094	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0.194806	0	0.194806	0	0.194806	0.194806
0	0	0	0.180036	0	0
0	0	0	0	0	0

This is a TINY portion of what we would get if we tokenized (broke into words) and the vectorized (made a data frame out of) all of the documents. Each **word** is a **column,** and each **document** is a **row**. Remember that a "document" can be anything from a Tweet to a review to a novel to a webpage.

The great news here is that we have converted text data into record data. We now have rows and columns. As such, we can apply all the same ML methods as we would with any record data.