[1]

In this lecture we address the important tasks required to get our data ready for machine learning training and testing. With the data exploration phase completed, we usually know which learning algorithms will suit our application, that is, which approach will produce the best model and be most compatible with our dataset. Getting ready for training usually requires data cleaning and data transformation activities be carried out before we can move on. Let’s look at these in more detail.

[2]

As previously noted, even the instances from a single dataset can emanate from multiple heterogenous sources such as sensors, observations, logs, surveys, archives and so on. This variety inevitably means that there will be problems with the data. Those problems must be addressed before attempting to train a machine learning model as the quality and viability of the model is directly affected by the quality of the dataset. The reasons for quality issues with data are manifold. Commonly they are caused by human error, either as the result data entry errors when collating data or because of programmer error and bugs leading to mistakes in a prior data curation or processing step. For different reasons, the data may incomplete either because there are missing values for some descriptive features or because of sampling error leading to a biased or unrepresentative distribution compared to the real-world problem domain being modelled.

[3]

The problem of missing values is probably the most common problem that needs addressing in datasets. A value is considered missing if the feature has no value present at all, has an invalid value present, has a zero value when something else would be expected or has some placeholder value indicating that the value is not applicable or could not be obtained. In any case, this represents an instance of a descriptive feature which cannot play a part in model training. The reasons why values may be missing are manifold but are most commonly due to data collection or data entry errors or mistakes make in preparatory pre-processing. The safest approach when dealing with missing values is to simply remove the affected instances from the dataset. The risk is that this could significantly reduce the number of the available instances to train and test with and impact the accuracy of the resultant model. On the other hand, not removing instances with missing features will almost certain impact the quality of the model.

[4]

The cardinality of a feature is the number of distinct values of that feature that are present in the dataset. For continuous features, the cardinality would be expected to be close to the total number of features as relatively few duplicate values are commonly found in real-world data. For categorical features, we could expect only a handful of distinct feature values in our dataset so this number should be relatively small. Inspecting the cardinality of a feature is a good sanity check of it value range. If a feature has a cardinality of one, then this might indicate that an error has occurred in data collection. At the very least, we can discard this feature altogether as it will not add any information to our model. For continuous features, we might need to investigate further if we measure a small cardinality as this is unusual, though not necessarily a problem. For categorical features we would be concerned if we see a relatively large cardinality as this would suggest a data collection or entry error or that that the feature has been misclassified. As, we noted in a previous lecture, we treat continuous and categorical features differently depending on the training algorithm we are using so establishing the correctness of these feature value domains is important to get right at this phase.

[5]

Another common issue to deal with is the problem of invalid data types and invalid data values for a given descriptive feature. For example, suppose we are expecting a feature to be numeric but we find a string or Boolean value present. Similarly, if we are expecting a string but see a numeric value present, we have a problem. Invalid data can sometimes be considered as missing data and can be discarded altogether. However, it may be possible to transform the data into the correct type or value based on what we have. For example, a string representation of a numeric value can be type-converted. Or a value of TRUE can be converted to the digit one if that is what is expected by the model. In some cases, the wrong measurement units may be the problem. For example, the model is expecting a length feature to be measure in millimetres, but meters are used in some instances. That kind of fix is trivial and is preferable to just discarding the affected instances. It is worth spending time to carefully examine data types and values in our datasets as, not to do so, adds risk that our resultant models will not perform well.

[6]

In the context of continuous features, outliers are values that lie far from the central tendency of the feature value distribution. The easiest way to spot outliers is to look at the maximum and minimum values for the feature to see if they make sense. Any values outside the expected range of a feature are probably outliers. In some cases very large or very small values are sentinel values representing the extremes of the range but which, by themselves, skew the probability distribution in an unhelpful way.

[7]

There are several approaches to handling missing values and outliers in a dataset. The simplest approach is to just discard affected instances from the training and test subsets. The problem with this approach is that it might adversely affect the number of available instances for learning and render the remaining features unrepresentative of the problem domain. Where possible, we would like to incorporate as much information as possible from our data, even from missing values. In some cases that fact that a value is missing may be, in itself, significant and useful for our model training. In such cases we could elect to transform the affected feature into a categorical variable (or just add an associated categorical feature) to classify the missing and present data in some way. It may transpire that this new feature type is correlated to the target variable in a helpful way. Alternatively, we can impute a plausible estimate for the missing value. For categorical features, we could just replace the missing value or outlier with the distribution mode value. For continuous features, we could use the mean or median value as a replacement. However imputation is generally only suitable for a relatively small number of missing values or outliers. As a rule of thumb, it can be used if the impact is less that 30% of the feature values in question.

[8]

Data cleaning (or fixing) is an essential part for data preparation but there are other kinds of transformations which may be necessary before submitting our dataset for model training. We’ve already discussed how the data exploration phase can help to uncover redundant or undesirable features from the dataset, notably features which do not contribute any useful information or would actually impact the performance of our training or resultant model. Those features (dataset columns) should be removed altogether. This is an example of what is known as dimensionality reduction. Dimensionality reduction helps us to train models faster and usually results in more accurate models being produced. Another issue can be having continuous features which are in different ranges and different measurement units. This can adversely impact the ability of many learning algorithms to infer useful information and relationships across these variables. Often it is worth transforming continuous feature variables using normalization or a standard score approach.

[9]

Normalization is an way of transforming a feature value range into another range with linear scaling preserved. For example, we may want to transform two or more feature of completely different ranges into the range specified by a high and low value. A common special case is to force our features into the range between zero and one. But this formula is sensitive to outliers in the feature range so an alternative to normalization is the standard score

[10]

The standard score of a variable range measures how many standard deviations from the mean a variable value is. This squeezes the feature distribution into a scaled distribution having a mean of zero and a standard deviation of one. The choice of which kind of transformation, if any at all, is appropriate will vary from application to application and from dataset to dataset. The recommendation is to trial different approaches through experimentation and measurement before settling on a final approach.

[11]

Finally, we consider the problem of continuous features with large counts and cardinality. A large cardinality for a feature may not be useful for some learning approaches, for example decision trees. One way to address this is to use a technique called binning. There are two kinds of binning commonly used. These are equal-width binning and equal-frequency binning. Binning requires that we specify the number of bins for our continuous feature values. Too low a number of bins results in a loss of resolution of our feature values. Too high a number of bins may result in some empty bins. Equal-width binning means dividing the feature range onto k, equally sized intervals and categorizing any variable values falling in that interval. Equal-frequency binning means dividing the feature value range having into k intervals having approximately the same number of data points. The same advise applies here as to which binning approach and bin size to choose. The recommendation is to trial different approaches through experimentation and measurement until the best one if found

[12]

For very large datasets, it may be implausible to use all of the available instances for training and testing. In such cases we may choose to subsample the instances instead. Top sampling is simply just selecting some arbitrary percentage of the data set from the top and running with that. However, this is not likely to be representative of the data as a whole so should be avoided. In particular, there is a high risk of getting an unrepresentative sample of the target features doing it this way. Random sampling a subset of instances from the uniform distribution is safer and less likely to introduce sample bias. Stratified sampling is a variation which guarantees that the sample distribution preserves the population distribution, that is the relative frequencies of the sampled feature values is maintained

[13]

In summary, we have considered the commonly used data cleaning and preparation activities prior to the model training phase. The quality of our machine-learned model is directly affected by the quality of the data using in the training and testing. Missing or invalid values and outliers must be removed or replaced. Continuous variables often need to be normalised and/or binned to help with training. Redundant or correlated features should also be eliminated. Failing to carefully clean and prepare our data will almost certainly result in underperforming models so developers must resist the temptation to rush into training before carrying these important activities.