**4.3** **ML training data and web mapping**

**Reading material**

|  |
| --- |
| Machine Learning in Practice  In the previous section we defined ML and discussed the key steps in Machine Learning. In this section, we will discuss the practical aspects of the key steps in Machine Learning in greater detail starting with data collection. |
| **Importance of Data Collection**  In the previous section, we discussed that the role of training data is to serve as the experience a learner uses to produce an expert (classifier). Thus, if we provide the wrong data the learner will produce a classifier that doesn’t do what we want it to do.  Without a foundation of high-quality training data, even the most performant algorithms can be rendered useless. Indeed, robust machine learning models can be crippled when they are trained on inadequate, inaccurate, or irrelevant data in the early stages. When it comes to training data for machine learning, a longstanding premise remains painfully true: Garbage In, Garbage Out. Therefore, ***no element is more essential in machine learning than quality of training data***. |
| Simply put, training data teaches the machine learning model. The learning algorithm (learner) continuously updates the classifier (model) based on the data.  For these reasons, the need for quality, accurate, complete, and relevant data starts early on in the training process.  It is also important to have the right amount of data. For many tasks, even if you have high quality data it is not possible to produce a good classifier with a small amount of data. Greater quantity of correctly sampled data reduces the chances that the classifier we train overfits to the dataset.  All in all, we must ensure we use quality sources of data, collect the right amount of data, and have some sort of quality assurance to ensure a successful training of our ML model. |
| **Data Collection: Sources of Data**  In the Papaya classification example in the previous section, we discussed how to create a training dataset for ML. In many instances one does not need to collect data on their own given that there are a multitude of high-quality open data sources.  It is possible to gather your own data and label it yourself. But since data collection is a lengthy process you can use an in-house team, crowdsourcing, or a data labeling service to do the work for you. You can also purchase training data that is labeled for the data features you determine are relevant to the machine learning model you are developing.  There are quite a few different sources to get the training data sets from, and the choice of these sources depends on your goals, the requirements of your machine learning project, as well as your budget, time, and personnel restrictions. |
| Collecting training data: open-source, web and IoT, ML  Source: <https://labelyourdata.com/articles/machine-learning-and-training-data> |
| **1. Open-source training data sets:** This might be an acceptable solution if you're very lucky or otherwise for smaller businesses and start-ups that don't have enough free resources to spend on data collection and labelling. The great benefit of this option is that it's free and it's already collected. But there's a catch (isn't there always?): such data sets were not initially tailored for your algorithm's specific purposes but for some other project’s. What this means for you is that you'll need to tweak and probably re-annotate the data set to fit your training needs. Many open-source data sets are available on the internet from sources such as World Bank, Yahoo Finance, and Quandl. |
| **2. Web scraping:** Web scraping is the process of extracting data from a website. This is a very common way of collecting training data sets that many machine learning companies use. This means that you use a program to extract data from a web page on the Internet. |
| **3. Internet of Things (IoT):** Sometimes sensors, cameras, and other smart devices may provide you with the raw data that you will later need to annotate by hand. This way of gathering a training data set is much more tailored to your project because you're the one collecting and annotating the data. On the downside, it requires a lot of time and resources, not to mention the specialists that know how to clean, standardize, anonymize, and label the data. |
| **4. Artificial training data sets (Data Augmentation):** This is the way that has started to gain traction in recent years. What it basically means is that you first create a machine learning model that will generate your data. This is a great way if you need large volumes of unique data to train your algorithm. It saves financial and personnel resources as it only needs to spend them on designing the model to create your data. Still, this method of collecting training data requires a lot of computational power, which is not usually freely accessible for small and middle-sized businesses.  Besides, if you need truly vast amounts of data, it will take some time to generate a voluminous high-quality training data set.  In some cases, it is possible to generate artificial data deterministically; that is, without using a machine learning model. One popular application of machine learning that heavily uses artificial data generated deterministically is optical character recognition. |
| **5. In-situ data collection:** any observation taken by an instrument in direct contact with the medium it "senses" is called an in-situ observation. In other words, In-situ data collection is collection of data by direct measurements. Temperatures measured by standard thermometers, wind speeds and directions measured by a cup anemometer and wind vane, and precipitation measured by a rain gauge are all very common in-situ weather observations. You may have even taken your own "homemade" in-situ weather observations before. Picking up blades of grass and tossing them in the air to get a sense for the wind direction, for example, would be an example of an in-situ observation. In remote-sensing, in-situ data is used to verify different measurements taken by remote sensing equipment.  Source: <https://www.e-education.psu.edu/meteo3/node/2224> |
| **Data Collection: Quantity of Data**  How much training data is needed?  There’s no clear answer - no magical mathematical equation to answer this question - but more quality data is better. The amount of training data you need to create a machine learning model depends on the complexity of both the problem you seek to solve and the algorithm you develop to do it. One way to discover how much training data you will need is to build your model with the data you have and see how it performs, by trial and error. |
| There are a few very broad guidelines that might help you get a basic idea about why this question has no answer:   * Usually, more sophisticated models with more attributes and links between them (like the Artificial Neural Networks) will require more data to train properly. * The scope of application along with the complexity of the real-life phenomena that your model is being trained to predict also plays a role in how much training data will be needed. Beware of the exceptions and blind spots. * With time, you will likely need to re-train or tweak your model as the trends that it predicts change, which will require more data in the long-term.   As you can see, a lot of factors play into the understanding of how much training data is enough. As a rule of thumb, experienced engineers have at least a general idea about the amount of data that will suffice to train your model. You should start listening to them, and then get more training data as you go. |
| **Data Collection: Quality of training data**  For a quality training data, the below points should be considered:   1. ***Relevant:*** The very first quality of training data should be relevant to the problem that you are going to solve. It means that whatever data you are using should be relevant to the current problem. For example, if you are building a model to analyze social media data, then data should be taken from different social sites such as Twitter, Facebook, Instagram, etc. 2. ***Uniform:*** There should always be uniformity among the features of a dataset. It means all data for a particular problem should be taken from the same source with the same attributes. 3. ***Consistent:*** In the dataset, the similar attributes must always correspond to the similar label in order to ensure uniformity in the dataset 4. ***Comprehensive:*** The training data must be large enough to represent sufficient features that you need to train the model in a better way. With a comprehensive dataset, the model will be able to learn all the edge cases.   Quality training data is vital when you are creating reliable algorithms.  Source: <https://www.cloudfactory.com/training-data-guide> |
| **Data Collection: Factors Affecting Data Quality**  What affects training data quality?  There are three main factors that can help you predict the level of quality you can expect from the people who work with your data - whether your workers are in-house, crowdsourced, or outsourced teams.   * **People**: The selection, development, and management of workers * **Process**: How workers do the work - from onboarding to task instructions to quality control workflow * **Tools**: The technology to access the work, manage workers, and maximize quality and throughput   factors that affects training data quality  Source: <https://www.cloudfactory.com/training-data-guide> |
| **Data Preparation**  We will now focus on the process of data preparation, the third step in the key steps of ML. In section 4.1, we discussed how the training data is used to teach the model, but a randomly sampled data is used to evaluate the test error. We call the data that is used to evaluate the test error the test data.  **What is a test dataset?**  Once the model is trained with the training dataset, it's time to test the model with the test dataset. This dataset evaluates the performance of the model and ensures that the model can generalize well with the new or unseen dataset. The test dataset is another subset of original data, which is independent of the training dataset. However, it has some similar types of features and class probability distribution and uses it as a benchmark for model evaluation once the model training is completed. Test data is a well-organized dataset that contains data for each type of scenario for a given problem that the model would be facing when used in the real world. Usually, the test dataset is approximately 20-25% of the total original data for a machine learning project. If you have a large dataset (with 1 million+ entries), as small as 2% of the total original data can be used as a test set. |
| At this stage, we can also check and compare the testing accuracy with the training accuracy, which means how accurate our model is with the test dataset against the training dataset. If the accuracy of the model on training data is greater than that on testing data, then the model is said to have **overfitting**.  The testing data should:   * Represent a part of the original dataset. * It should be large enough to give meaningful predictions. |
| **Data Preparation: Train-Test Split**  Splitting the dataset into train and test sets is one of the important parts of data pre-processing.  If the model is trained with a training set and then tested with a completely different test dataset, then the model will not be able to understand the correlations between the features. Therefore, training and testing the model with two different datasets will decrease the performance of the model. Hence it is important to split a dataset into two parts, i.e., **train and test set.**  Training data vs testing data in ML  Source: <https://labelyourdata.com/articles/machine-learning-and-training-data> |
| In this way, it is easy to evaluate the performance of the model. Such as, if it performs well with the training data, but does not perform well with the test dataset, then it is estimated that the model may be overfitted.  Recently, there are different machine learning approaches that are trained with one dataset and tested on a completely different dataset. The task of designing classifiers that work well on data that looks different from the one they were trained on is called domain adaptation. For example, if you have a self-driving car trained with street data from the city of London, you still will want the car to work as well in the streets of Kigali. To achieve this one can use different domain adaptation techniques. |
| **Data Preparation: Recap Training Vs. Test Data**   * The main difference between training data and testing data is that training data is the subset of original data that is used to train the machine learning model, whereas testing data is used to check the accuracy of the model. * The training dataset is generally larger in size compared to the testing dataset. The general ratios of splitting train and test datasets are 80:20, 70:30, or 90:10. * Training data is well known to the model as it is used to train the model, whereas testing data is like unseen/new data to the model |
| In most ML applications, collecting data once and training may not meet performance expectations. Sometimes model evaluation on the test data might indicate more data is needed; hence, the data collection, and data preparation steps might have to be redone after every data collection. We can understand the whole process of training and testing in three steps, which are as follows:   1. **Feed**: Firstly, we need to train the model by feeding it with training input data. 2. **Define**: Now, training data is tagged with the corresponding outputs (in supervised learning), and the model transforms the training data into text vectors or a number of data features. 3. **Test**: In the last step, we test the model by feeding it with the test data/unseen dataset. This step ensures that the model is trained efficiently and can generalize well.   The above process is explained using a flowchart given below:  Train and Test datasets in Machine Learning  Source: <https://www.javatpoint.com/train-and-test-datasets-in-machine-learning> |
| **Data Preparation: Some Data Preparation Steps** A lot of elaborate data preparation techniques can be applied. Here we discuss some of the widely used data preparation steps.   * Data cleaning: cleaning the data to remove unwanted data, missing values, rows, and columns, duplicate values, data type conversion, etc. You might even have to restructure the dataset and change the rows and columns or index of rows and columns. * Data visualization: just to understand how data is structured and understand the relationship between various variables and classes present. * Splitting the cleaned data into two sets - a training set and a testing set. The training set is the set your model learns from. A testing set is used to check the accuracy of your model after training. |
| * Label frequency: If there are 10.000 samples of sweet papayas in 10 samples of not-sweet papayas than model can assume that all papayas are sweet because doing so results in a small training error (< 0.0001). A dataset with skewed label proportions is called imbalanced. The class that has a large proportion in the dataset is called majority class. * Standardization: standardization is the rescaling of features to ensure the mean and the standard deviation to be 0 and 1, respectively. This is important when you have measurements in different units. If you have two columns in your data, one measured in kilometers and another in millimeters, the mm measurements will be large numbers while the km measurements will be small numbers. Consequently, your model might learn to weight the mm measurements more than the km measurements. When this is not desired, the features are standardized so all measurements have similar orders of magnitude. |
| * One hot encoding: One-hot encoding is a way of representing categorical data as an array of zeros and ones. In one-hot encoding, you will always have an array (vector) with length equal to the number of categories. One of the elements in the array will be one while the rest are zero. The position of the one in the array indicates the category the data point belongs to. Look at the following example, In the image below, having a 1 on the first column (and zero elsewhere) indicates the data point is Red. Thus, the array (vector) [1, 0, 0] indicates a Red item.   Source: https://www.kaggle.com/code/dansbecker/using-categorical-data-with-one-hot-encoding   * Augmentation: This is artificially generating training data to your dataset. Section 4.2.12 discusses this in greater detail |
| **Model Choice**  Model choice is the third key step in ML.  After the data is prepared we have to select the kind of model we will train. There are many classes of ML methods to choose from. A big part of the choice depends on the data we collected. If the data collected is unlabelled, then we must use models amenable to unsupervised learning but if we have labels we can use models that require labels. |
| The following image shows different types of ML models that can be used in supervised and unsupervised settings (excluding deep learning models). |
| **Model Choice: Considerations** We have seen that a large part of the model choice depends on the kind of data available. However, there are other considerations as presented below.   1. **Explainability:** In many situations, explaining the results of a model is paramount. Explainability is important in medical applications, legal applications, and critical industrial applications. Unfortunately, many algorithms work like black boxes, and the results are hard to explain regardless of how good they are. The lack of explainability may be a deal-breaker in those situations. Linear Regression and Decision Trees are good candidates when explainability is an issue. Neural networks, not so much. Understanding how easy it is to interpret the result of each model is important before picking a good candidate. |
| 1. **Complexity:** A complex model can find more interesting patterns in the data, but at the same time, it will be harder to maintain and explain. A couple of loose generalizations to keep in mind: More complexity can lead to better performance but also larger costs. Complexity is inversely proportional to explainability. The more complex the model is, the harder it will be to explain its results. Putting explainability aside, the cost of building and maintaining a model is a crucial factor for a successful project. A complex setup will have an increasing impact during the entire lifecycle of a model. 2. **Dataset Size:** The amount of training data available is one of the main factors you should consider when choosing a model. A Neural Network is really good at processing and synthesizing tons of data. A KNN (K-Nearest Neighbors) model is much better with fewer examples. Going beyond the amount of available data, a related consideration is how much of it you truly need to achieve good results. Sometimes you can build a great solution with 100 training examples; sometimes, you need 100,000. Use this information about your problem and the amount of data to choose a model that’s capable of processing it. |
| 1. **Training time and cost:** How long it takes, and how much it costs to train a model? Would you choose a 98%-accurate model that costs $100,000 to train or a 97%-accurate model that costs $10,000? Of course, the answer to this question depends on your individual circumstances. Models that need to incorporate new knowledge in near real-time can’t afford long training cycles. For example, a recommendation system that needs to be constantly updated with every user’s action benefits from an inexpensive training cycle. Balancing time, costs, and performance is crucial when designing a scalable solution. 2. **Inference time:** How long does it take to run a model and make a prediction? Imagine a self-driving system: it needs to make decisions in real-time, so any model that takes too long to run can’t be considered. For example, most of the processing needed to develop predictions using KNN happens during inference time. This makes it expensive to run. However, a Decision Tree will be lighter during inference time and will require more time during training.   Source: https://towardsdatascience.com/considerations-when-choosing-a-machine-learning-model-aa31f52c27f3 |
| **Model Choice** Oftentimes ML engineers train multiple models and use the evaluation results to make a model choice. Although the considerations discussed can be helpful to narrow down model choice, they should not be used to pick a single model. It is better to train multiple models that meet the requirements of the ML task and base your decisions on the performance of the models.  **Training the Model**  The following key step in ML is training the model. The way model training is carried out depends on the model used, the data storage, and the type of computational resource used. But in general, in this step the training data is processed by the learning algorithm to output a classifier. |
| **Hyperparameter Tuning**  The next step in ML is hyperparameter tuning. Before we discuss hyperparameter tuning we must define hyperparameter and distinguish it from parameter.  **Parameter Vs Hyperparameter**   * A model parameter is a configuration variable that is **internal to the model** and whose value can be **estimated from the given data**. |
| * Parameters:   + They are required by the model when making predictions.   + Their values define the skill of the model on your problem.   + They are estimated or learned from data.   + They are often **not set** manually by the practitioner.   + They are often saved as part of the learned model. |
| * A model hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data.   + They are often used in processes to help estimate model parameters.   + They are often specified by the practitioner.   + They can often be set using heuristics.   + They are often tuned for a given predictive modeling problem.   Source: <https://www.datacamp.com/tutorial/parameter-optimization-machine-learning-models> |
| **Hyperparameter Tuning: Comparing Classifiers**   * Let’s say you have trained two same classifiers called SVM-1 and SVM-2 with different hyperparameters for our papaya classification task and the classifiers have the performance shown in the table below. * If you had to use one of the two models for production which one would you use? Why? * Now your boss comes and tells you that a test error of 0.1 or better is required and you must retrain one of the models. Which one would you re-train? |
| * Let’s say you chose to retrain SVM-1 because it has smaller test error and retrained to get a test error of 0.08.  Is the test error of the retrained model a good approximation of the generalization error of the retrained model?   + No, it is not. The reason is by using the information that SVM-1 performs better on the test data to choose it for retraining we can no longer consider the test data unseen.   + If you have used some data to change anything on one of your classifier, learning algorithm, training settings, or model settings it cannot be considered unseen data.   + In this case, the test error was used to inform the choice between the settings used for svm-1 and svm-2.   + So how can such choices about model settings be made and still have a test error that approximates the generalization error? |
| * A third set of data known as validation data is used to make such choices. * The learning algorithm never sees the validation data as its training input. |
| **Training Vs Validation Vs Test**    **Training data:**   * Is the data that will be seen by the learning algorithm. * Used by the learning algorithm to produce a good classifier   **Validation data:**   * Is never used to train the learning algorithm exceptions aside * Used to evaluate a trained model for further use * Used by the machine learning practitioner to determine what training settings, classifier settings, … to use for better generalization   **Test data:**   * Is never used to train the learning algorithm * Is never used to make any decision about classifier choice, classifier settings, training settings, … * Used to obtain an approximation of the generalization error of a classifier     Note: There are other approaches to choosing classifier settings that aren’t discussed here.  Read more in the following link https://www.cs.cmu.edu/~schneide/tut5/node42.html |
| **Evaluating the Model:**  As was previously discussed, we use test data to evaluate the performance of the tuned model. This will give us the final verdict on how well a model does on unseen data. |
| **Deployment:**  Machine learning model deployment is the process of placing a finished machine learning model into a production environment where it can be used for its intended purpose. Models can be deployed in a wide range of environments, and they are often integrated with apps through an API (Application Programming Interface) so they can be accessed by end users. The environments where an ML model can be deployed to range from small microcontrollers, to standalone servers, to distributed server cluster, to cloud environments.  The process of actually deploying the model requires several different steps or actions, some of which will be done concurrently.  First, the model needs to be moved into its deployed environment, where it has access to the hardware resources it needs as well as the data source that it can draw its data from. Let’s take the ever popular ChatGPT model as an example. ChatGPT is deployed on Microsoft Azure cloud infrastructure. Thus, the trained model will have to be configured to work on the virtual machine Microsoft Azure provided.  Second, the model needs to be integrated into a process. This can include, for example, making it accessible from an end user’s laptop using an API or integrating it into software currently being used by the end user. In the case of ChatGPT, the developers have made an API for programmers available at https://api.openai.com/v1/chat/completions. Whenever a request is sent to the url the request is processed and forwarded to the model deployed on Microsoft Azure. The output of the model is forwarded as an HTTP data to the caller. This API is meant for use by developers.  Third, the people who will be using the model need to be trained in how to activate it, access its data and interpret its output. Sometimes we would like to provide access to the power of a model to non-technical users. In such cases, we may design an interface where people can interact with our model by simply using graphical or command line tools. For example, ChatGPT allows interaction with the model through a web interface at https://chat.openai.com  Source: https://www.dominodatalab.com/blog/machine-learning-model-deployment#:~:text=Machine%20learning%20model%20deployment%20is,be%20accessed%20by%20end%20users. |
| **ML Process as a Cycle** Until now we have described the key steps in ML as a straight process. However, the process is actually cyclic that involves multiple loops. Many times the loops in ML cycle might take us as far back as collecting new data. For example, training data can evolve with time.  Training data is used not only to train but to retrain the model throughout the artificial intelligence development lifecycle. Training data is not static: as real-world conditions evolve, the initial training dataset may be less accurate in its representation of ground truth as time goes on, requiring to update the training data to reflect those changes and retrain the model.  AI model development lifecycle for training data and machine learning.  Source : <https://www.cloudfactory.com/training-data-guide> |
| **ML in Remote Sensing**  We have seen the key steps in ML in detail. But why do we use ML in remote sensing?  Environmental remote sensing involves the use of satellites and other air-borne instruments to collect data about the environment.  The rapid increase in the volume of remote sensing data obtained from different platforms has encouraged scientists to develop advanced, innovative, and robust data processing methodologies. Machine learning methods are widely applied to remote sensing datasets; they have been used to classify ships from remote sensing images, determine the distribution of palm trees in a forest from images and so much more.  Machine learning algorithms allow a system to learn and improve from data and experience without being specifically programmed, reducing the level of human intervention. This data-driven approach means valuable information about a natural phenomenon can be extracted from the data alone. |
| Key Points:   * Machine learning with remote sensing can help to improve predictions about the behavior of environmental systems, improve the automation of data analysis, lead to a better management of resources and the discovery of new insights from complex data sets. * Applications include improved weather forecasting, flood and drought prediction, precision agriculture, forest management and in marine conservation and coastal clean-up projects. * Wider implementation for remote sensing is limited by the availability of accessible and representative datasets for training the machine learning algorithm. Specific challenges include: the availability of *Analysis Ready Data* which isdata that have been processed to a minimum set of requirements and organized into a form that allows immediate analysis with a minimum of additional user effort and interoperability both through time and with other datasets- it requires expertise, time and computational power to prepare; the demands on storage, transfer and processing of large data sets; and the demands on having an accurate and well-developed training data set.   In the following section, we will discuss the image classification and discuss some applications of ML in remote sensing. |

**Exercise materials and tasks**

**Quiz questions**

|  |
| --- |
|  |
| Please answer the following questions to test your understanding so far:   1. What is training data?    1. **A dataset used to train machine learning models**    2. A dataset that evaluates the performance of the model    3. A new dataset that the model can generalize well 2. Please connect the steps of the process of training and testing data work in ML with the corresponding activities:  |  |  | | --- | --- | | Step 1 | The model is tested by feeding it with the test data/ unseen dataset | | Step 2 | Training data is tagged with the corresponding outputs | | Step 3 | The model is fed with training input data |   System feedback:   * Student answers different from below: “Sorry, that’s not correct, please check the process of training and testing data work in ML”  |  |  | | --- | --- | | Step 1 | The model is fed with training input data | | **Step 2** | **Training data is tagged with the corresponding outputs** | | **Step 3** | **The model is tested by feeding it with the test data/ unseen dataset** |   System feedback:   * Student answers like follows: ”Correct, these are the steps of the process of training and testing data work in ML:”  1. How much training data do we need for machine learning?    1. Algorithms tell you when it’s sufficient    2. **We don’t know**    3. **You'll get perception with experience** |