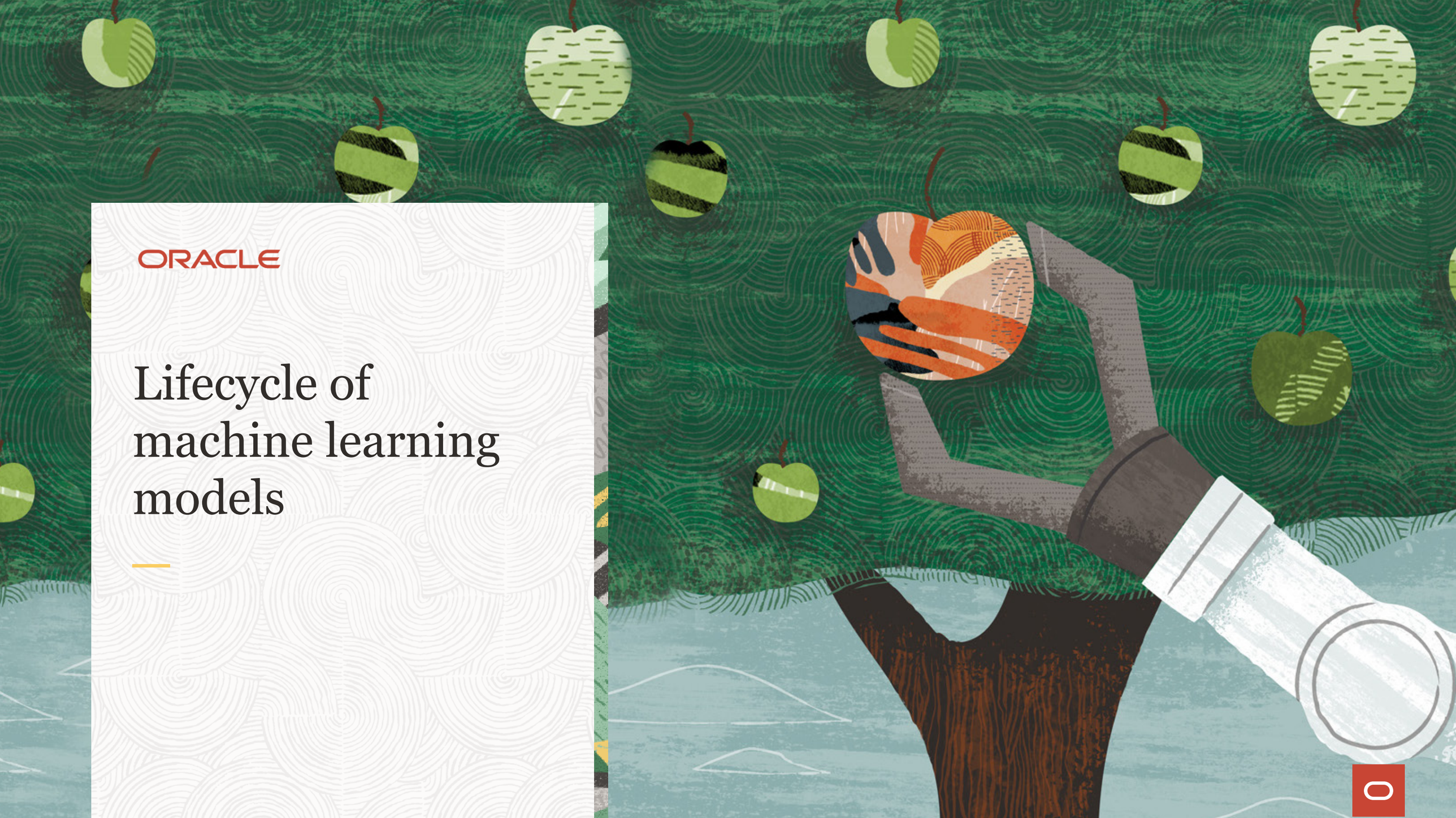
ORACLE

Lifecycle of machine learning models

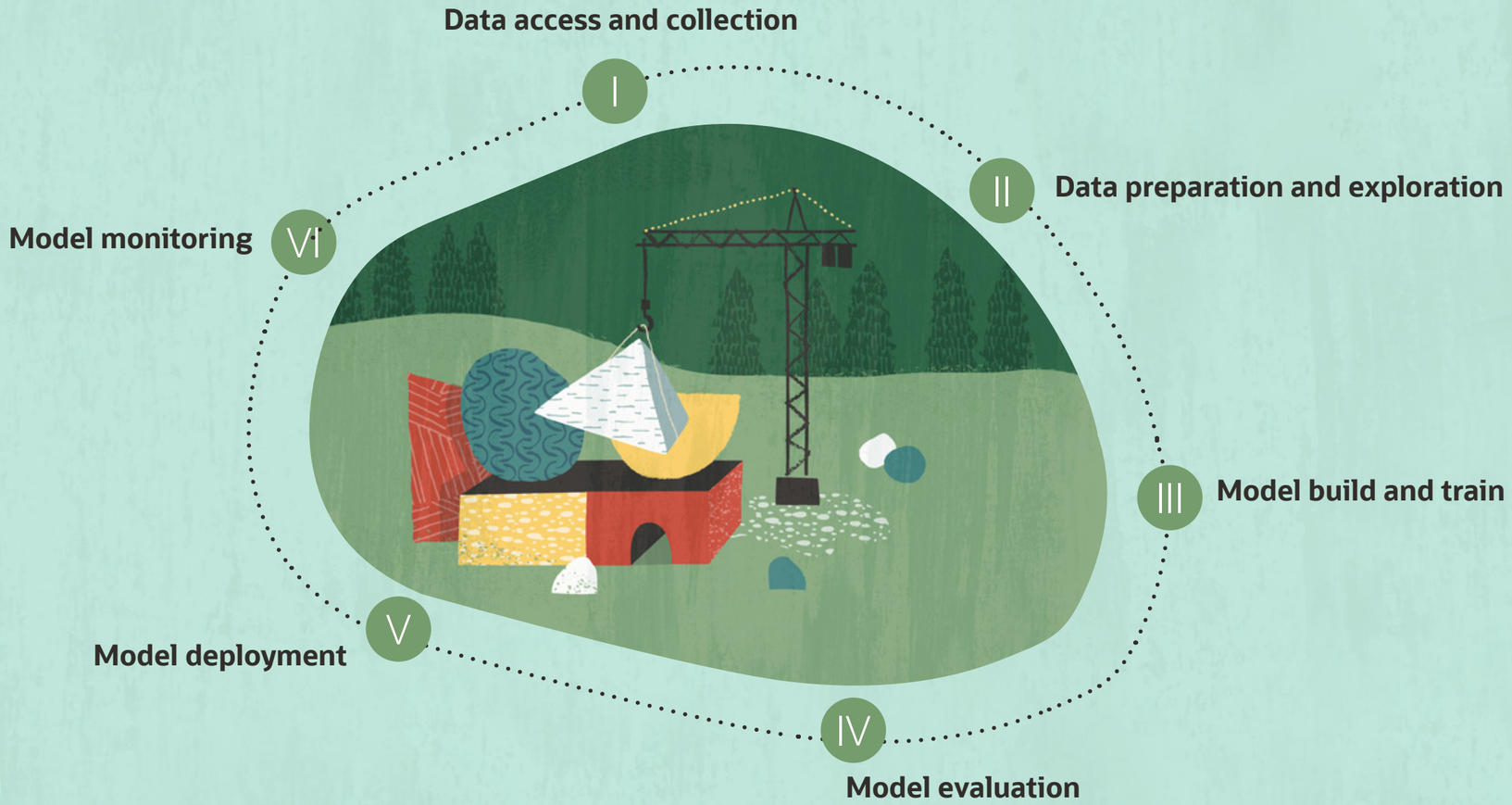


Introduction

Machine learning is an area that enterprises are increasingly investing in or identifying as a potential area of growth. There are many reasons enterprises invest in machine learning, from being able to leverage data to find insights about their customers to making processes more efficient. In this book, we break down how machine learning models are built into six steps: data access and collection, data preparation and exploration, model build and train, model evaluation, model deployment, and model monitoring.

Building a machine learning model is an iterative process. Many of the steps needed to build a machine learning model are reiterated and modified until data scientists are satisfied with the model performance. This process requires a great deal of data exploration, visualization and experimentation as each step must be explored, modified and audited independently.

Steps of building machine learning models



I. Data access and collection

The first step to a machine learning problem is accessing the data. Typically, data scientists will obtain the data for the business problems they are working on by querying the databases where their companies store their data. In addition, there is a lot of value in unstructured datasets that do not fit well into a relational database (e.g. logs, raw texts, images, videos, etc.). These datasets are heavily processed via Extract, Transform, Load (ETL) pipelines written by data engineers and data scientists. These datasets either reside in a data lake or in a database (either relational or not). When data scientists do not have the data needed to solve their problems, they can get the data by scraping data from websites, purchasing data from data providers or collecting the data from surveys, clickstream data, sensors, cameras, etc.

II. Data preparation and exploration

After getting the data, data scientists have to prepare the raw data, perform data exploration, visualize data, transform data and possibly repeat the steps until it's ready to use for modeling. Data preparation is cleansing and processing raw data before analysis. Before building any machine learning model, data scientists need to understand the available data. Raw data can be messy, duplicated or inaccurate. Data scientists explore the data available to them, then cleanse the data by identifying corrupt, inaccurate and incomplete data and replacing or deleting it.

In addition, data scientists need to determine if the data has labels or not. For example, if you have a series of images and you want to develop a detection model to determine whether there is a car in the image, you need to have a set of images labeled whether there is a car in them and most likely need bounding boxes around the cars in the images. If the images lack labels, data scientists will have to label them. There are open source tools and commercial vendors that provide platforms for data labeling, as well as human labelers for hire.

After data is cleansed, data scientists explore the features (or the variables) in their dataset, identify any relationship between the features and make decisions about any necessary data transformations. There are various tools data scientists can use for exploratory data analysis in open source libraries and analytics/data science platforms. A tool that performs statistical analysis of the dataset and creates data visualizations to generate plots of the features is useful in this step.

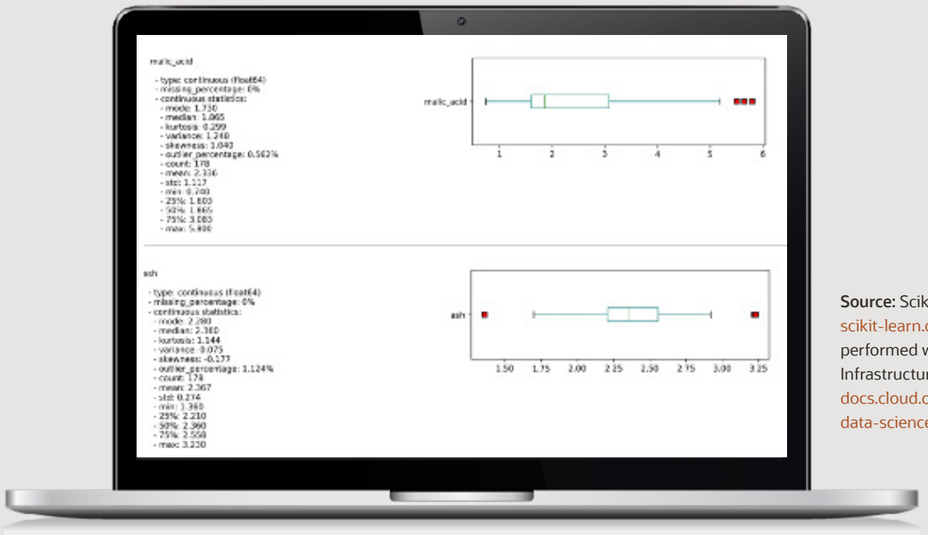
It is important to see what types of features are in the dataset. Features can be numerical, which can be a floating point or integer. Categorical features have a finite number of possible values, typically assigning data into groups. For example, if you have a dataset from a customer survey, the respondent's gender (male or female) is a categorical feature. Ordinal features are a categorical feature with a set order or scale. For example, customer satisfaction response: very satisified, satisfied, indifferent, dissatisfied, and very dissatisfied has a set order to it. You can convert that ordering into an integer scale (1->5). After determining what kind of features there are, obtaining a distribution of values that each of the feature has and getting summary statistics of each feature would be next. Doing so would help answer the following questions about the dataset:

- Is the dataset skewed towards a range of values or a subset of categories?
- What are the minimum, maximum, mean, median and mode values of the feature?
- Are there missing values or invalid values such as null? If so, how many are there?
- Are there outliers in the dataset?

During the data exploration step, it is helpful to plot the features and also plot the features against each other to identify patterns in the dataset. This helps to determine the need for data transformation. Some of the questions you need to answer are:

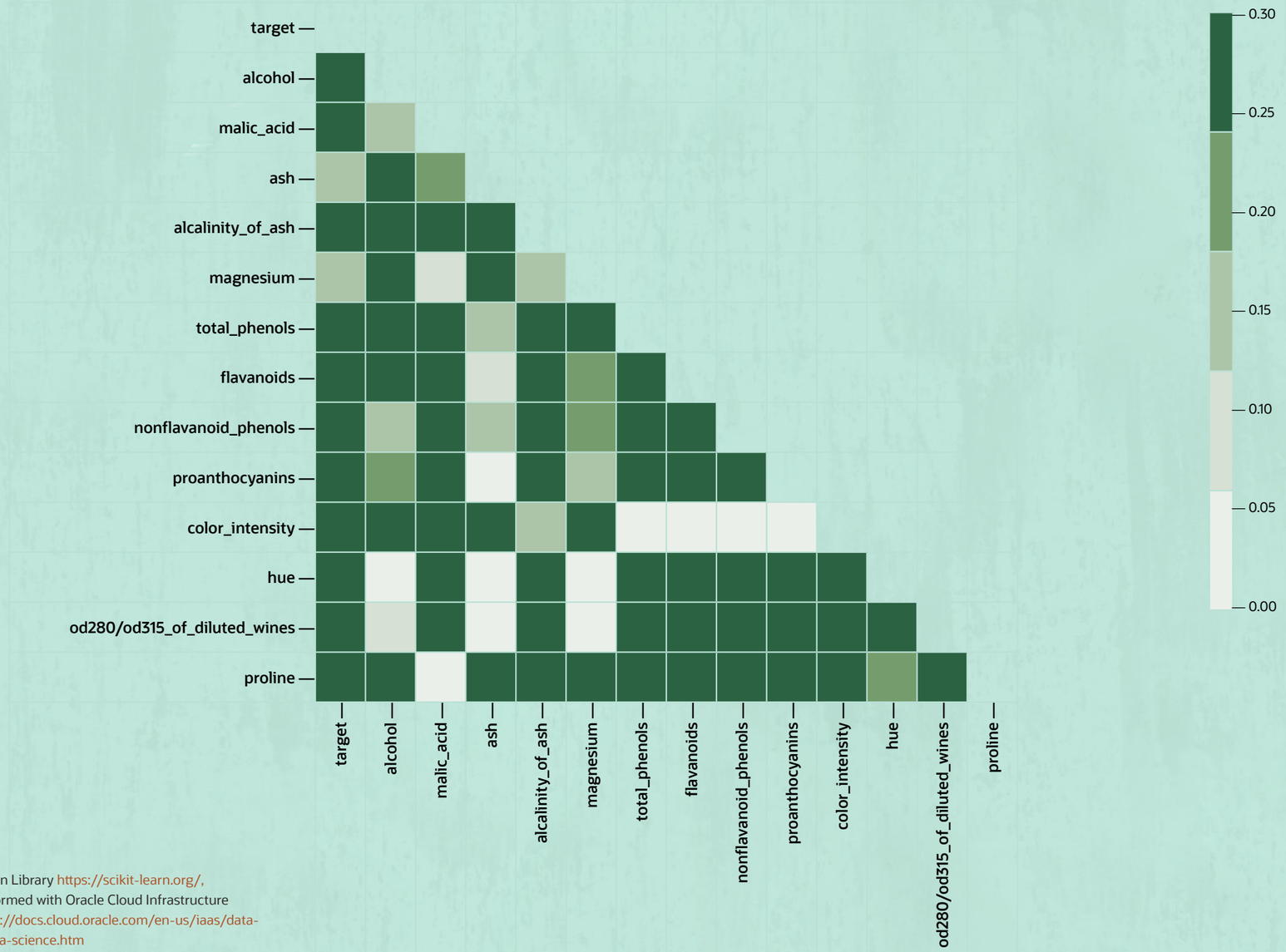
- How do you handle missing values? Do you want to fill in the values and if so, what approach do you plan to take to fill in for the missing value? Some approaches include taking the mean value, the median, the mode, nearby entry's value and average of nearby entries' values.
- How will you handle outliers?
- Are some of your features correlated with each other?
- Do you need to normalize the dataset or perform some other transformation to rescale the data (e.g. log transformation)?
- What is your approach to a long tail of categorical values? Do you use them as-is, group them in some meaningful way or ignore a subset of them altogether?

Summary statistics and visualization of features in a dataset of three types of wine and features of each wine.



Source: Scikit Learn Library https://scikit-learn.org/, Visualization performed with Oracle Cloud Infrastructure Data Science https://docs.cloud.oracle.com/en-us/iaas/data-science/using/data-science.htm

Heatmap of how correlated the features are to each other, from a dataset with three types of wine and features of each wine.



Source: Scikit Learn Library https://scikit-learn.org/, Visualization performed with Oracle Cloud Infrastructure Data Science https://docs.cloud.oracle.com/en-us/iaas/data-science/using/data-science.htm

During the data exploration step, you can identify patterns in your dataset for ideas about how to develop new features that would better represent the dataset. This is known as feature engineering. For example, if you have a traffic dataset for the number of vehicles passing through a major intersection at every hour, you might want to create a new feature categorizing the hour into different parts of the day, such as early morning, mid-morning, early afternoon, late afternoon, and nighttime.

For categorical features, often it is necessary to one hot encode the feature. One hot encoding means turning a categorical feature into binary features, one for each of the categories. For example, suppose you have a dataset of customers, and we have a feature on which states the customer comes from: Washington, Oregon, and California. One hot encoding would produce two binary features where one feature is whether a customer is from Washington state or not, and the second feature is whether a customer is from Oregon or not. It is assumed that if the customer is not from Washington or Oregon, he / she would be from California, so there is no need for a third feature.

III. Model build and train

Model build consists of choosing the correct machine learning models to solve the problems and features that go into the models. In the first step of model build, data scientists need to decide what might be the appropriate machine learning model to solve the problem. There are two main types of machine learning models: supervised and unsupervised. Supervised learning involves modeling a set of input data to an output or a label. Classification and regression are supervised learning problems. Unsupervised learning involves modeling a set of input data without a label. For example, customer segmentation is an unsupervised learning problem. You do not know a priori what customer segment a customer belongs to. The segment will be assigned by the model.

Different classes of machine learning models are used to solve unsupervised and supervised learning problems. Typically, data scientists will try multiple models and algorithms and generate multiple model candidates. Data scientists do not know a priori what model will perform best on the dataset, so they experiment with several of them. During the model training, a data scientist might do feature selection which is the process of selecting only a subset of features as input to the machine learning model. The benefits of reducing the number of input variables are to reduce computational cost of model training, make the model more generalizable and possibly improve model performance.

During the model training, the dataset is split up into training and testing sets. The training dataset is used to train the model, and the testing dataset is used to see how well the model performs on data it has not seen. Model evaluation will be discussed in more detail below.

Model hyperparameter tuning is a major task in the model training process. Models are algorithms, and hyperparameters are the knobs that a data scientist can tune to improve the performance of the model. For example, the depth of a decision tree is a hyperparameter.

You can choose to have a very deep or very shallow decision tree. This will affect the bias and variance of your model. Bias is the error from underfitting or the error from not capturing the relation between the features and the outputs. Variance is the error from overfitting where the model does well in the training dataset but does not perform well to unseen data. Tuning the hyperparameters of a model can be partially automated, although data scientists should always be involved in the process.

Data scientists also have to decide what kind of compute resources they need for training their models. You can prepare the data and train the models locally on your computer. However, depending on how much data there is to prepare and then used to train the model, your computer may not be enough. You may have to transition the workload to the cloud where you can have access to a broader selection of computing resources including GPUs.

Some models can be trained faster on specialized hardware (e.g., training perceptrons/deep neural network models on GPUs.) You may also explore distributed training environments that can speed up the process, especially when the amount of data cannot fit in the memory of the largest machine available, through splitting and distributing the data across multiple machines, or when you want to simultaneously train multiple model candidates in parallel on separate machines.

AutoML

AutoML has garnered quite a bit of attention over the past few years due to the promise of being able to make machine learning more accessible to a larger audience. AutoML stands for automated machine learning. It automates the process of feature selection, model/algorithm selection, and hyperparameter tuning. It is a feature that all major data science platforms have. Users can feed a dataset to AutoML, and it will train multiple machine learning models, tune the hyperparameters for those models, and evaluate their performance against each other.

AutoML can improve the productivity of data scientists by automating the training process. It also allows data analysts and developers to build machine learning models without tweaking every aspect of the model training process that comes with data science expertise. Most AutoML capabilities support tabular data for classification and regression problems while others have more advanced offerings that support images and text data, as well as time series forecasting.

The drawback to AutoML, or any complicated model, is it can seem like a blackbox solution making it difficult for users to understand how the models arrive at the predictions. Users should look to the model explainability offering of the AutoML system to see what capability exists to help users interpret the models and understand how the selected models arrive at the predictions.

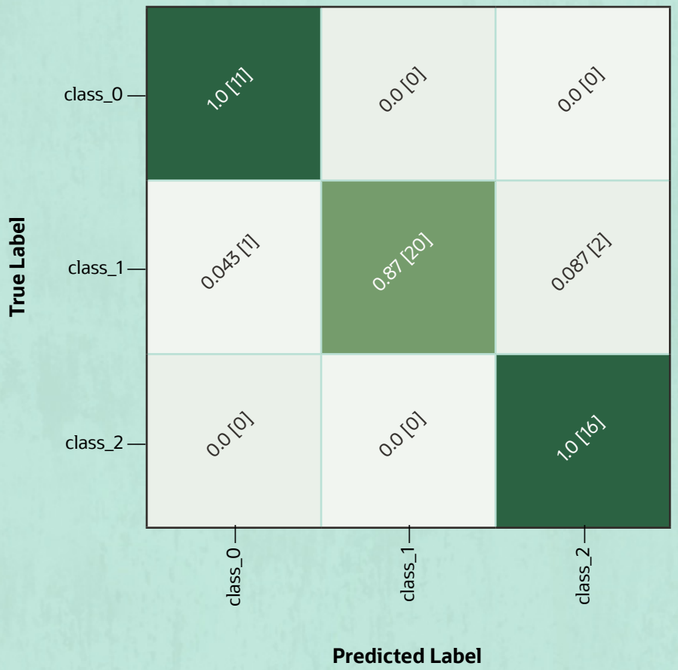
Model explanations typically fall into global explanation and local explanation. Global explanation is understanding the general behavior of a machine learning model as a whole. This includes explaining how important each feature is in contributing to the model predictions. Local explanation provides an understanding on why the machine learning model made a particular prediction for one data sample. For example, why did a fraud detection algorithm predict a particular transaction as fraudulent?

IV. Model evaluation

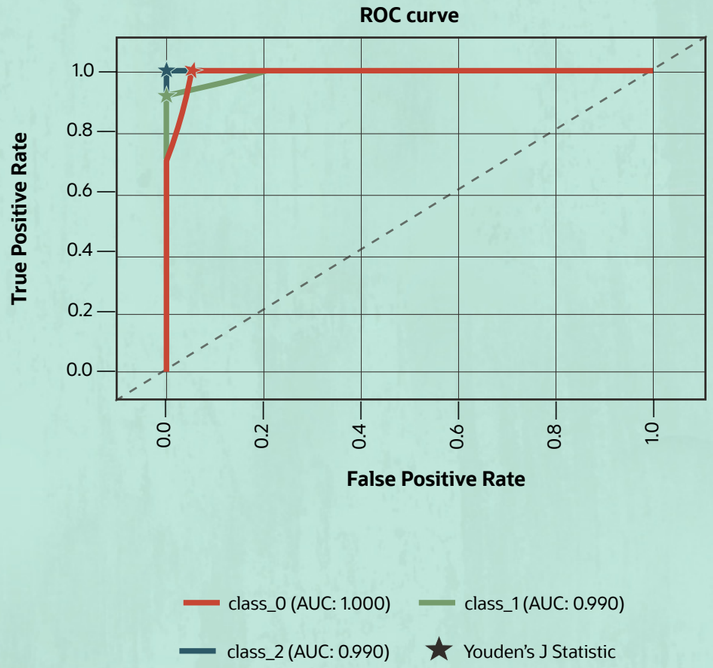
There are many open source tools that help data scientists calculate the metrics for evaluating machine learning models and help them visualize the metrics (e.g., AUC-ROC curve, gain and lift charts.) When evaluating machine learning models, data scientists need to decide which metrics are important for the business problem they are trying to solve.

For classification problems, one can use accuracy for model evaluation, but sometimes it might not be the optimal metric choice. If a problem involves detecting whether someone has a rare illness, a better metric might be how many people with the illness are accurately diagnosed divided by all the people with the illness. In that case, it would be more useful to look at a confusion matrix which shows the number of true positives, true negatives, false positives, and false negatives, and calculate precision and recall. For regression problems, you can use metrics such as root-mean-square error, mean absolute error or calculate the coefficient of determination r^2. For unsupervised problems, a set of clusters with high cohesion within the clusters and separation between the clusters is considered ideal. This can be measured with metrics such as silhouette score and Calinski-Harabasz coefficient.

Confusion matrix for multiclass classification for the results from a random forest model on predicting the type of wine based on the features of a wine, from a dataset containing three types of wine and features of each wine.



ROC curve for multiclass classification constructed for the results from a random forest model on predicting the type of wine from a dataset containing three types of wine and features of each wine.



Source: Scikit Learn Library https://scikit-learn.org/, Visualization performed with Oracle Cloud Infrastructure Data Science https://docs.cloud.oracle.com/en-us/iaas/data-science/using/data-science.htm

V. Model deployment

After the model training and evaluation processes are complete, the best candidate models are saved. Models are usually saved in Pickle, ONNX and PMML format. Depending on the objectives, data scientists might work on a machine learning problem for proof of concept, experimentation or to deploy it to production. Model deployment is consuming the predictions made by the machine learning model in some way. Most likely, the pipeline of data transformations have to be deployed also. Typically, data scientists will work with engineers on model deployment.

Depending on how you intend to consume the predictions, you can deploy for batch consumption or real time consumption. For batch consumption, the predictions can be scheduled (e.g., every hour, every day.) The predictions can then be stored in a database and consumed by other applications. Typically, the amount of data you process is larger than for real-time prediction. A use case would be if you run an e-commerce site and you want to send a weekly email to the customers about recommended products for them based on past purchases. The machine learning models can be scheduled to run ahead of time.

For real-time consumption, a trigger would initiate the process of using the persisted model to serve a prediction. For example, deciding whether a transaction is fraudulent or not when payment is initiated, requires real-time prediction. You have to consider how quickly you have to serve the predictions (milliseconds, seconds?), the volume of demand for the service, and the size of data to run predictions on. Minimizing latency to serve prediction is important. You can improve the serving latency by using a smaller model in size, using accelerators such as GPU and improving how features related to the entity are retrieved for real-time prediction (e.g. If you are recommending products to a user as the user is browsing a site, improvements on how information on past purchases of the user is fetched can improve the latency.)

There are different tools and cloud platform offerings for model deployment such as Functions-as-a-Service (FaaS) platforms, fully managed deployment of models as HTTP endpoints, DIY with flask or Django in a container orchestration platform such as k8 and docker swarm, etc.

VI. Model monitoring

Model monitoring is a challenging step that is sometimes forgotten by organizations without mature machine learning and data science initiatives. Model retraining and redeployment requires time from the data science and engineering team and compute resources. Model monitoring helps the team decide if and when it is necessary to retrain the model and redeploy. Model monitoring can be broken down into two components: drift/statistical monitoring of the model performance and ops monitoring.

After models are deployed, the metrics by which the models were measured and trained go down in production. This is because data is non-stationary. The non-stationarity can manifest in many ways: features in production data can take values outside of the range in the training dataset; there can be a slow drift in the distribution of the values, etc.

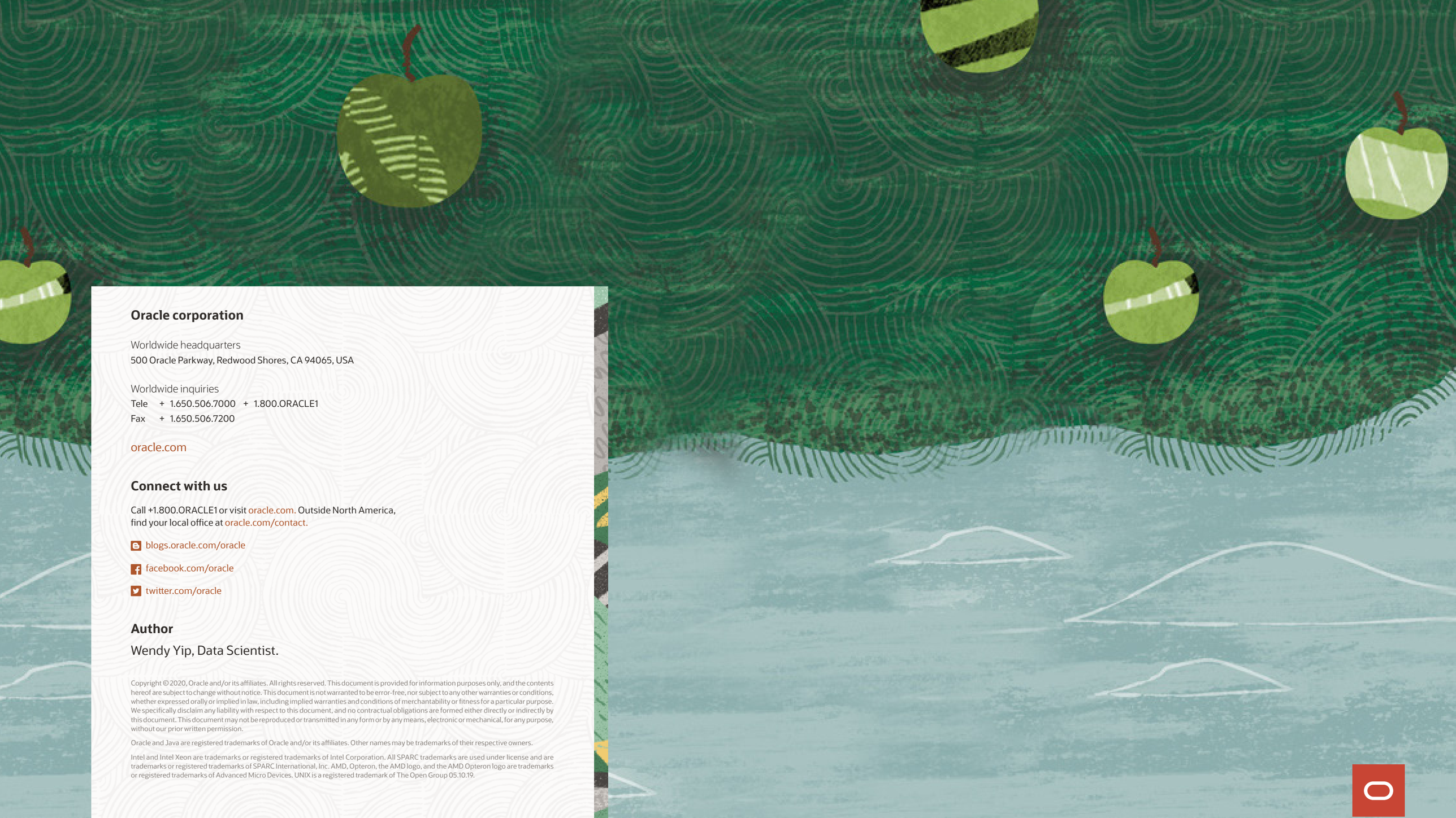
Because of the model degradation, the quality of the models need to be monitored to decide if and when to retrain the model and redeploy. Sometimes, it is not possible to immediately obtain the prediction accuracy of live data going to a production system. For example, it might take some time before you can decide whether a churn prediction model or a fraud detection model provided an accurate prediction. However, it is possible to look at the statistics and distribution of the training data compared to live data and also compare the distribution of the model predictions with training and live data. For example, if you are working with a customer churn model, you can compare the features of your customers used to train your model compared to the features of customers in the production system. Also, you can look at the percentage of customers predicted to churn in the training sample compared to the live production.

Ops monitoring of the machine learning system will require partnership between the data scientists and engineering team. Things to monitor include serving latency, memory/CPU usage, throughput and system reliability. Logs and metrics need to be set up for tracking and monitoring. Logs contain records of events, along with the time when they occurred. They can be used to investigate specific incidents and figure out the cause of the incident. Kibana is an open-source tool used for searching and viewing logs. Metrics measure the usage and behavior of the machine learning system. Prometheus and Grafana are tools for monitoring metrics.

Conclusion

We hope this has been a useful guide on the steps it takes to build a machine learning model. It is important to remember that machine learning is a very iterative process, and the steps outlined in this book will be reiterated and improved upon many times.

There are many resources available that dive deeper into each of the steps covered in this book, and you can learn more about them as you make decisions about your enterprise's data science strategy. If you're ready to get started, Oracle offers hands-on labs so you can experiment with building your own data science models.



Oracle corporation

Worldwide headquarters
500 Oracle Parkway, Redwood Shores, CA 94065, USA

Worldwide inquiries
Tele + 1.650.506.7000 + 1.800.ORACLE1
Fax + 1.650.506.7200

oracle.com

Connect with us

Call +1.800.ORACLE1 or visit oracle.com. Outside North America, find your local office at oracle.com/contact.

blogs.oracle.com/oracle

facebook.com/oracle

twitter.com/oracle

Author

Wendy Yip, Data Scientist.