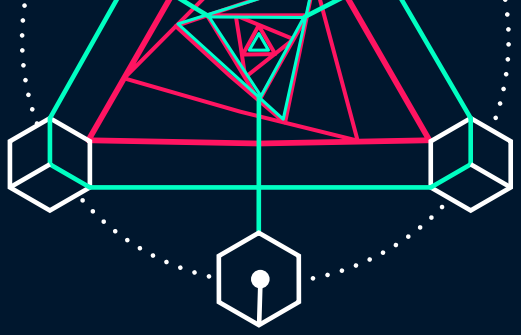
data iku

HOW TO:
EXECUTE ANOMALY DETECTION AT SCALE

For Discovering Fraud, Network Monitoring, Health Care, Uncovering Emerging Trends, and More

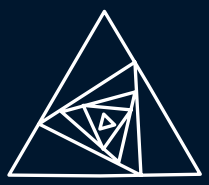
A GUIDEBOOK BY DATAIKU

www.dataiku.com



About the Guidebook

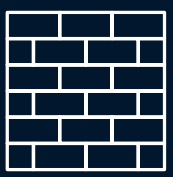
This guide is intended to provide a high-level overview of what anomaly detection is, how it works, and when it can be applied. By the end, readers should have an understanding of:



What anomaly detection is and how it differs from other similar systems.

The three types of anomalies that can be detected.

Use cases and examples of where anomaly detection is employed.



How to build a basic anomaly detection system.

For those completely unfamiliar with data science in the context of anomaly detection, this guide will provide a short introduction to the topic and walk through the core aspects.

But on top of that, for those already familiar, the guide includes some code and practical examples for execution.

What?

Anomaly detection is all about finding patterns of interest (outliers, exceptions, peculiarities, etc.) that deviate from expected behavior within dataset(s). Given this definition, it's worth noting that anomaly detection is, therefore, very similar to noise removal and novelty detection. Though patterns detected with anomaly detection are actually of interest, noise detection can be slightly different because the sole purpose of detection is removing those anomalies - or noise - from data.

As with most data science projects, the ultimate end goal or output of anomaly detection is not just an algorithm or working model. Instead, it's about the value of the insight that outliers provide. That is, for a business, money saved from preventing equipment damage, money lost on fraudulent transactions, etc. In health care, it can mean earlier detection or easier treatment.

ANOMALY DETECTION REQUIRES A SYSTEM THAT IS AGILE AND CONSTANTLY LEARNING BECAUSE:

The very nature of the use cases for anomaly detection (specifically in the IT and finance sectors) means often times fraudsters are specifically and deliberately trying to produce inputs that don't look like outliers. Adapting to and learning from this reality is critical.

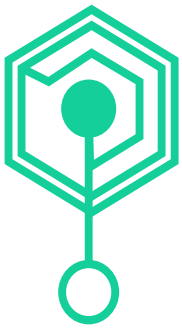
Aside from malicious intents, datasets generally tend to change over time as users change, so a system needs to evolve along with those users. Anomalies, by their nature, are unexpected, so it's important that any methods used are adaptive to the underlying data.

Many use cases are extremely time sensitive, and businesses (or patients, customers, etc., of those businesses) can't afford to wait. Detecting patterns early based on a combination of data points can help anticipate issues before it's too late.

It is important to note that, despite the most common use cases being detection of fraud or system intrusion, anomalies are not always bad - that is, they don't always have to indicate that something is wrong.

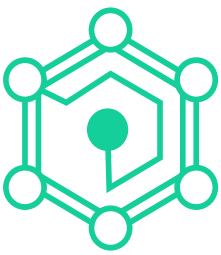
Anomaly detection can also be used, for example, to detect or predict slight changes in customer or user behavior that may then result in a shift in selling, development, or marketing strategy, allowing businesses to stay a step ahead of new trends.

There are three basic types of anomalies that may be detected:



Point anomalies:

Point anomalies are simply single, anomalous instances within a larger dataset. For example, a temperature of 60 degrees Centigrade in a dataset would be a point anomaly, as that would be the highest temperature ever recorded on Earth. Anomaly detection systems often start by identifying point anomalies, which can be used to detect more subtle contextual or collective anomalies.



Contextual (or conditional) anomalies:

These are points that are only considered to be anomalous in certain context. A good example is temperature again; while 30 degrees Centigrade is considered to be within the range of possible temperatures, given the context of December in New York City, this data point is certainly an anomaly. With spatial data, latitude and longitude are the context, while with time-series data, time is the context.

Collective anomalies:

When related datasets or parts of the same dataset taken together are anomalous with respect to the entire data set (even when individual datasets don't contain anomalies). For example, say there is data from a credit card making purchase in the US, but also a dataset showing money being taken out of ATMs in France at the same time. A collective anomaly may occur if no single anomaly happens in any one dataset, but all datasets measuring various components taken together signal an issue.

Why?

Anomaly detection is an approach that can be useful across an array of industries and for a variety of purposes. But the underlying, unifying factor is the ability to detect small changes or differences in a system that might otherwise go unnoticed. Uncovering anomalies using machine learning allows humans (or other systems) to take action based on these outliers.

Specifically, a non-exhaustive look at use cases for anomaly detection systems include:

INDUSTRY	USE CASE(S)	CHALLENGE(S)
IT, DEVOPS	Intrusion detection (system security, malware), production system monitoring, or monitoring for network traffic surges/ drops	**Need for a real-time pipeline to react; huge volumes of data plus unavailability of labeled data corresponding to intrusions making it difficult to train/test; usually have to adopt a semi-supervised or unsupervised approach.**
MANUFACTURING AND INDUSTRY, CONSTRUCTION, AGRICULTURE, AND MORE	Predictive maintenance, service fraud detection	**Industrial systems produce data from different sensors that varies immensely - different levels of noise, quality, frequency of measurement.**
HEALTH CARE	Condition monitoring, including seizure or tumor detection	**Costs of misclassifying anomalies are very high; also, labeled data more often than not belongs to healthy patients so usually have to adopt a semi-supervised or unsupervised approach.**

INDUSTRY	USE CASE(S)	CHALLENGE(S)
FINANCE AND INSURANCE	Fraud detection (credit cards, insurance, etc.), stock market analysis, early detection of insider trading	**Financial anomaly detection is high risk so it must be done truly real time so that it can be stopped as soon as it happens. Also, it's more important perhaps than other use cases to be careful with false positives that may disrupt user experience.**
PUBLIC SECTOR	Detection of unusual images from surveillance	**Requires deep learning techniques, making this type of anomaly detection more expensive.**

Anomaly detection can be useful in a number of other fields and industries where rare events are very important or impactful, but they are hard to find within data.

Because of its wide array of applications, mastering anomaly detection from a data scientist's perspective is an incredibly applicable use case.

How?

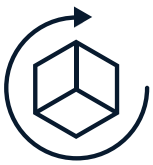
If you're familiar with the seven fundamental steps to building a data project, then you already know the basics for how to get started using anomaly detection to benefit your team or business. But there are also several particularities to bear in mind when working with anomaly detection:

1 Understand the business

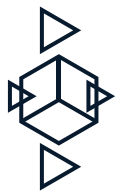
The first step in successful anomaly detection is to really understand what kind of a system the business needs and to lay out framework for the requirements and goals of anomaly detection before diving in. These are important preliminary discussions because not all anomaly detection work is the same; exactly what qualifies as an anomaly and the subsequent processes kicked off by anomaly detection vary vastly by (and even among) use cases.

Notably, the nature of the data, of the problem at hand, and the goals of the project necessarily dictate the techniques employed for anomaly detection. Even within a single, specific industry like health care, different projects will have different definitions of what makes a datapoint an anomaly. Very small fluctuations in a system tracking body temperature, for example, could be considered anomalies, while other systems could tolerate a much larger range of inputs. So it's not as easy to universally apply a single approach as it is for other types of data projects.

To ensure the success of a project involving anomaly detection, data team members will need to work directly together, collaborating with other non-data teams (business, operations, legal, etc., depending on the domain) to:



Define and continually refine what constitutes an anomaly. It might constantly change, which means continual re-evaluation.



Determine, once an anomaly is detected, what the system will do next. For example, send anomalies to another team for further analysis and review, automatic actions on an associated asset/account, etc.

Define goals and parameters for the project overall. For example, the end goal is probably not just to detect anomalies, but something larger that impacts the business, like block fraudulent charges, cure more patients by detecting health conditions earlier, increase revenue by anticipating future trends, etc. Having larger goals will allow you to better define the scope of the project and the expected output.

Develop a plan to monitor and evaluate the success of the system going forward.

Identify what anomaly detection frequency (real time vs. batch) is appropriate for the business and the use case at hand.

2 Get your data

Having as much data for anomaly detection as possible will allow for more accurate models because you never know which features might be indicative of an anomaly. Using multiple types and sources of data is what allows a business to move beyond point anomalies into identifying more sophisticated contextual or collective anomalies. In other words, variety is key.

For example, looking at fraud detection, it's possible that transaction data isn't anomalous because the fraudster has stayed within the "normal" range of the actual user's habits. But data from ATM use or account weblogs may reveal anomalies.

GO FURTHER

For the sake of simplicity, this guidebook, will walk through a simple case of anomaly detection, more specifically a fraud detection example where the goal is to predict whether or not a mobile payment transaction is fraudulent.

The theoretical dataset contains several fields of information regarding the transactions themselves, the recipients, and the clients that made the payments. The schema would look like this:

```
transaction_id | transaction_date | transaction_type | transaction_amount | recipient_
id | recipient_country | client_ip_address | client_card_mask | client_card_country |
client_email_address | is_fraudulent
```

In a supervised context, the is_fraudulent **column represents the target variable, namely the actual status of the transaction (fraudulent or legit).**

3 Explore, Clean, and Enrich Data

When doing anomaly detection, this stage is even more important than usual, because often the data contains noise (usually errors, either human or not) which tends to be similar to the actual anomalies. Hence, it is critical to distinguish between the two and remove any problematic data that could produce false positives.

In an ideal world, you'd have a sufficient amount of labeled data from which to begin; that is, you'd be able to enrich the datasets you have with information on which records represent anomalies and which are normal. If possible, starting with data you know is either anomalous or normal is the preferred way to begin building an anomaly detection system because it will be the simplest path forward, allowing for supervised methods with classification (as opposed to unsupervised anomaly detection methods).

For some of the use cases detailed above, this is likely very attainable. Specifically in financial for fraud detection or manufacturing/industry for predictive maintenance because there is a clear mechanism for feedback on which cases where anomalous (customer relationship manager data detailing fraud complaints or maintenance records). For other use cases, like condition monitoring in health care or intrusion detection, having enough labeled data from which to begin could be difficult, though it's still possible to successfully detect anomalies without labeled data.

GO FURTHER

In most data science use cases, and especially in anomaly detection, the data preparation portion can have a significant impact on the result of the model. Applying the right processing steps and engineering relevant features are indeed very good ways to better characterize potential outliers.

In the case of fraud detection and given this particular dataset, several useful operations can be performed on the initial dataset to create additional information on each transaction, for example:

- Parse transaction_date and extract date features (e.g., day of week, week of year).
- Derive the client's country from client_ip_address via IP geolocation.
- Extract the email domain from client_email_address.

More advanced feature engineering operations will also prove useful here. In the case of fraud detection, it is common to compute rank features by leveraging window functions (performing a calculation across a set of table rows that are somehow related to the current row).

For example, in this fraud detection case, it's important to know how many different IP addresses were used by a given client (identified by his email address) for his purchases. The corresponding PostgreSQL query is:

```
SELECT "transaction_id",
        SUM(CASE WHEN "first_seen" THEN 1 ELSE 0 END)
                                OVER (PARTITION BY "client_email_address" ORDER BY "transaction_date")
AS "distinct_ip"
FROM(
 SELECT "transaction_id",
          "client_email_address",
          "transaction_date",
                                    "transaction_date" = MIN("transac_date") OVER (PARTITION BY "client_
email_address", "client_ip_addr") AS "first_seen" FROM "transactions_dataset"
```

Other examples of typical rank features for fraud detection include:

- How many times did a client make a transaction from a given country?
- How many times were a given user and a given beneficiary involved in a common transaction?

4 Get Predictive

There are two primary architectures for building anomaly detection systems:

Supervised anomaly detection, which you can use if you have a labeled dataset where you know whether or not each datapoint is normal or not.

Unsupervised anomaly detection, where the dataset is unlabeled (i.e., whether or not each datapoint is an anomaly is unreliable or unknown).

When using a supervised approach, you'll apply a binary classification algorithm. Exactly which algorithm is less important than making sure to take the appropriate measures regarding class imbalance (i.e., the fact that for anomaly detection, it's highly likely that you have far more "normal" cases than anomalous ones).

When using an unsupervised approach, there are two ways of training your algorithms:

NOVELTY DETECTION:

The training set is made exclusively of inliers so that the algorithm learns the concept of "normality" (hence the prefix "one-class" found in some methods). At test time, the data may also contain outliers. This is also referred to as semi-supervised detection.

OUTLIER DETECTION:

The training set is already polluted by outliers. The assumption is made that the proportion of outliers is small enough so that novelty detection algorithms can be used. Consequently, those algorithms are expected to be robust enough at training time to ignore the outliers and fit only on the inliers.

GO FURTHER

In a supervised context, the `is_fraudulent` label is available, and the prediction problem can thus be seen as a binary classification task that takes a matrix (X) containing all the features in numeric form and a target vector (y) representing the labels.

However, some extra steps have to be taken because this case deals with a highly class-imbalanced problem (i.e., the outliers are vastly underrepresented):

1

During the cross-validation phase, ensure that both train and test sets have the same proportion of outliers. This can be done using a stratified split - here's an example in Python using the Scikit-learn library:

```
from sklearn.model_selection import train_test_split
X_train, X_test,y_train, y_test = train_test_split(X, y,
test_size=0.35, stratify=y, random_state=42)
```

2

An additional option is to use resampling techniques, i.e., only taking a subset of inliers while keeping the full population of outliers.

In a fully unsupervised case, there is no access to the `is_fraudulent` label. Consequently, it's necessary to resort to special outlier detection algorithms that are trained only on the feature matrix (X) and return an anomaly score for each data point at evaluation time.

Following the example for an unsupervised case, first train the Isolation Forest algorithm on the transaction data:

```
from sklearn.ensemble import IsolationForest
dtr = IsolationForest().fit(X)
df["y_scored"] =  -dtr.decision_function(X)
```

Then compute the anomaly rank of each transaction and append it to the original dataset (in a pandas DataFrame format). By doing so, it will then be possible to sort the transactions by decreasing anomaly level for direct access to the most suspicious ones (as defined by the model) at the top of the list:

```
df["anomaly_rank"] = df["y_scored"].rank(method="dense", ascending=0)
df.sort_values(by=["anomaly_rank"], ascending=True, inplace=True)
```

In some cases, such a ranking method can be improved by adding a projected damage column, i.e., the amount of money that would be lost in a given transaction if it were to be fraudulent.

5 Visualize

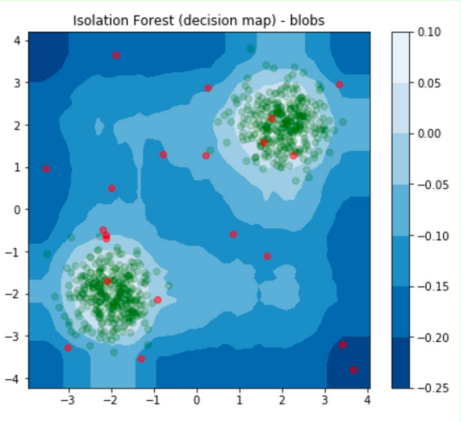
Visualizations are especially useful in the process of building and testing anomaly detection models because sometimes they are the clearest way to see outliers, especially in very large datasets.

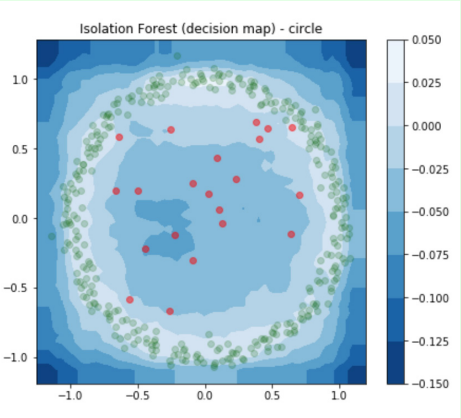
GO FURTHER

In an unsupervised context, it is possible to build a decision map where the anomaly score is computed at each point of a grid that spans over the feature space. In practice, this allows the observation of the zones where inliers are likely to be regrouped according to the model. Any point lying outside of those areas has a higher probability of being an anomaly.

In the following examples, decision maps are built after training an Isolation Forest algorithm on simple two-dimensional datasets with various cluster shapes for the inliers (in green), and a randomly uniform repartition for outliers (in red). Data located in an area with darker shades of blue is more likely to be an anomaly.

Note that in practice, real-life datasets usually have more than two features, so to be able to apply a decision map methodology, there is a prerequisite of applying feature reduction to the initial dataset.





6 Deploy and Iterate

To have real impact with an anomaly detection system, your model should be scoring data real time in production. Anomaly detection is generally time sensitive, so going to production to make predictions on live data rather than retroactively on test or stale data is more important than ever.

But putting a model in production isn't the end. Iteration and monitoring of anomaly detection systems is critical to ensuring that the model continues to learn and be agile enough to continue detecting anomalies even as user behaviors change. However, unlike other types of machine learning models, accuracy is not a viable metric for anomaly detection. Since the vast majority of data is not composed of anomalies (i.e., there could be hundreds of thousands of "normal" data points), the system could achieve a very high accuracy but still not actually be accurately identifying anomalies.

GO FURTHER

Instead of accuracy, anomaly detection systems may rely on the following evaluation methods:

Recall:

The ratio of correctly detected anomalies to total anomalies.

False Positive Rate:

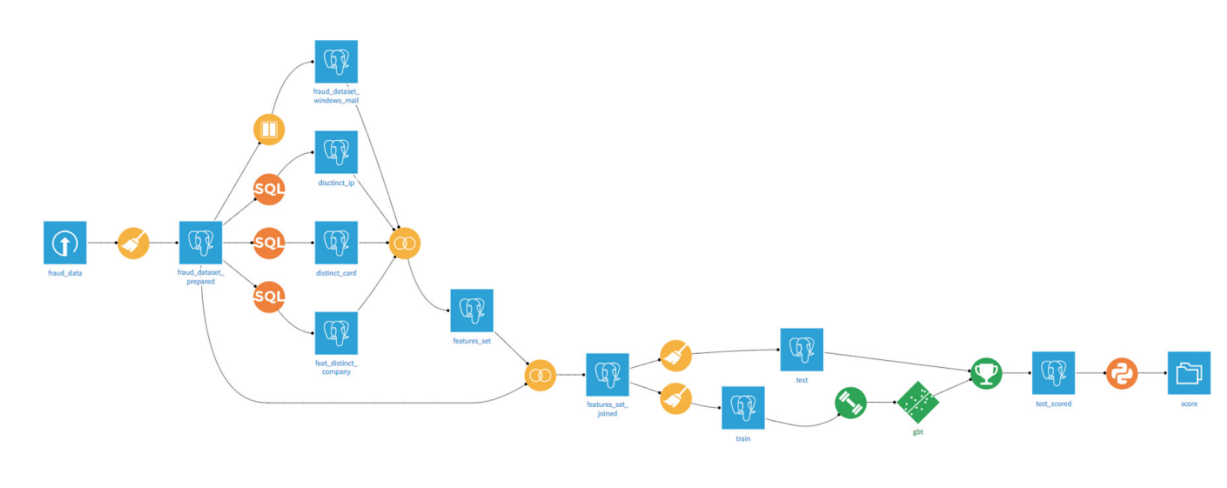
The ratio of misclassified anomalies to total records.

Receiver Operator Characteristics (ROC) curve:

Tradeoff between detection and false alarm rate.

Regarding iteration, keep in mind that by far the most laborious step when it comes to anomaly detection is feature engineering. Continuing to iterate until false positives/negatives are reduced and the system is effective yet agile is a time consuming yet critical part of the process.

What can be helpful here is having a visual representation of the entire process so that iteration is simpler and faster every time, even once the model is in production; for example, here's an overview of our fraud detection example that we worked through in Dataiku Data Science Studio (DSS), but any visual representation can be effective:



What Are the Pitfalls?

TOO MANY FALSE NEGATIVES/POSITIVES:

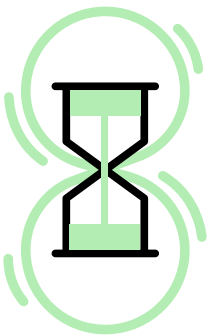
Evaluating anomaly detection is a particularly fine balance. False negatives can be detrimental, of course, but on the other hand, a system that identifies too many false positives is of almost no use either. In a real time system, there is no room for second review of any potential anomalies if the system is producing too much nose. And in many use cases, false positives in anomaly detection could destroy the reputation of the company and cause lost business (think of the frustration, for example, if your bank was constantly blocking your funds due to false positive fraud detection).

Solution: Spending the time upfront on feature engineering to make sure there aren't too many false negatives or positives and continuing to iterate, refine, and make improvements even after the model is in production: both are critical.

MISSING OR UNRELIABLE DATA:

A system that is not robust enough is not the only cause of false positives. Another can be simply unreliable data, which is unfortunately a problem across many different industries, but particularly health care and manufacturing.

Solution: Investing in improvement in systems and sensors to ensure complete and reliable data is an essential component of accurate anomaly detection. Because no matter how good your model is, it will not perform well if it's based on poor quality data.



LACK OF AGILITY TO HANDLE CHANGES IN THE NORM:

In today's world, change is the only constant. Human beings change over time, so the idea of normal behavior in the context of anomaly detection will continue to shift. In addition, systems also change over time, but gradual change doesn't always equate to anomalous behavior.

Solution: Any anomaly detection system you build, no matter what the use case, needs to be agile enough to shift with changing norms. Similarly, there should be a plan in place to continually monitor and review the system to ensure it's still performing as expected over time. Systems should also take into account any seasonality or other patterns - for example, a customer at a bank may generally make larger-than-normal purchases around the holidays, but these should not be necessarily flagged as fraudulent.

FAILURE TO TIE INTO BUSINESS OBJECTIVES:

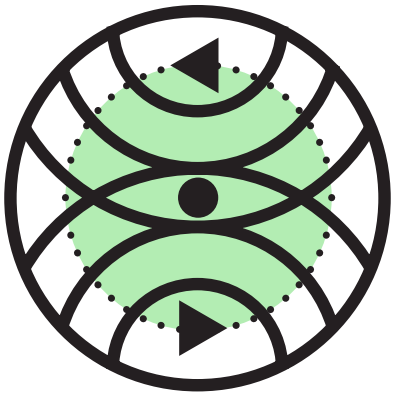
Like many systems of intelligence, it's easy to get disconnected from the business side and develop a system that doesn't take the proper next steps or follow-up based on detected anomalies.

Solution: Whether it's sending the case for review by a customer service team, notifying a particular staff member, or countless other next-step actions, anomaly detection systems need to actually take some action in the end to be effective. Building a system in a vacuum of a data science team won't do any good.

Looking Ahead

UNSTRUCTURED DATA

Because of the breadth of its utility, especially as fraudulent activity and attacks on systems become more pervasive, anomaly detection will continue to become more sophisticated. There have been, and will continue to be, developments in the realm of anomaly detection with unstructured data (like images and text). Developments in anomaly detection using deep learning in imaging will be particularly influential in health care.



AUTOMATED PREVENTION

Additionally, developments will go from pure detection of anomalies to automated prevention techniques. Being able to stop potentially harmful anomalous behavior before it happens has the potential for broad impact in the years to come, particularly, again, when it comes to cyber security and fraud detection.

MORE SCALABLE PROCESSES

And finally, in the years to come as more and more industries hinge on anomaly detection, expect to see an overall streamlining of anomaly detection processes as businesses continue to scale. This means more and more companies investing in the right architecture to retrieve data critical for anomaly detection work, the means to process it quickly, and apply models for impact in production.

data iku

Your Path to Enterprise AI

Dataiku is the centralized data platform that moves businesses along their data journey from analytics at scale to enterprise AI. Data-powered businesses use Dataiku to power self-service analytics while also ensuring the operationalization of machine learning models in production.

20,000+

ACTIVE-USERS

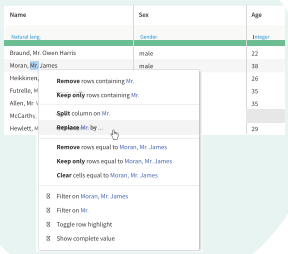
*data scientists, analysts, engineers, & more

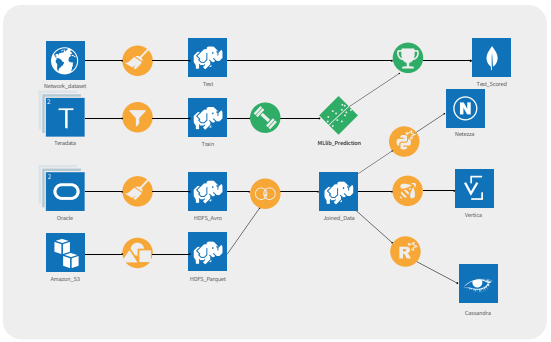
200+

CUSTOMERS

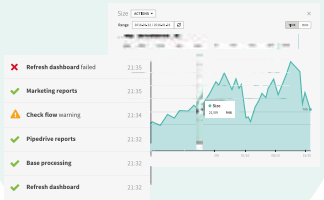
GE · Santander · paloalto NETWORKS · PREMERA BLUE CROSS · FOX NETWORKS GROUP · Unilever · NXP · SEPHORA · KUKA · BNP PARIBAS

1. Clean & Wrangle



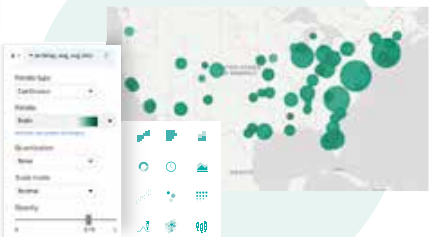


5. Monitor & Adjust

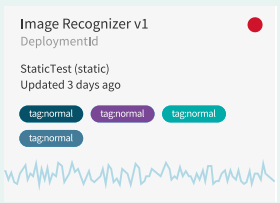


2. Build + Apply Machine Learning

3. Mining & Visualization



4. Deploy to production



GUIDEBOOK

www.dataiku.com