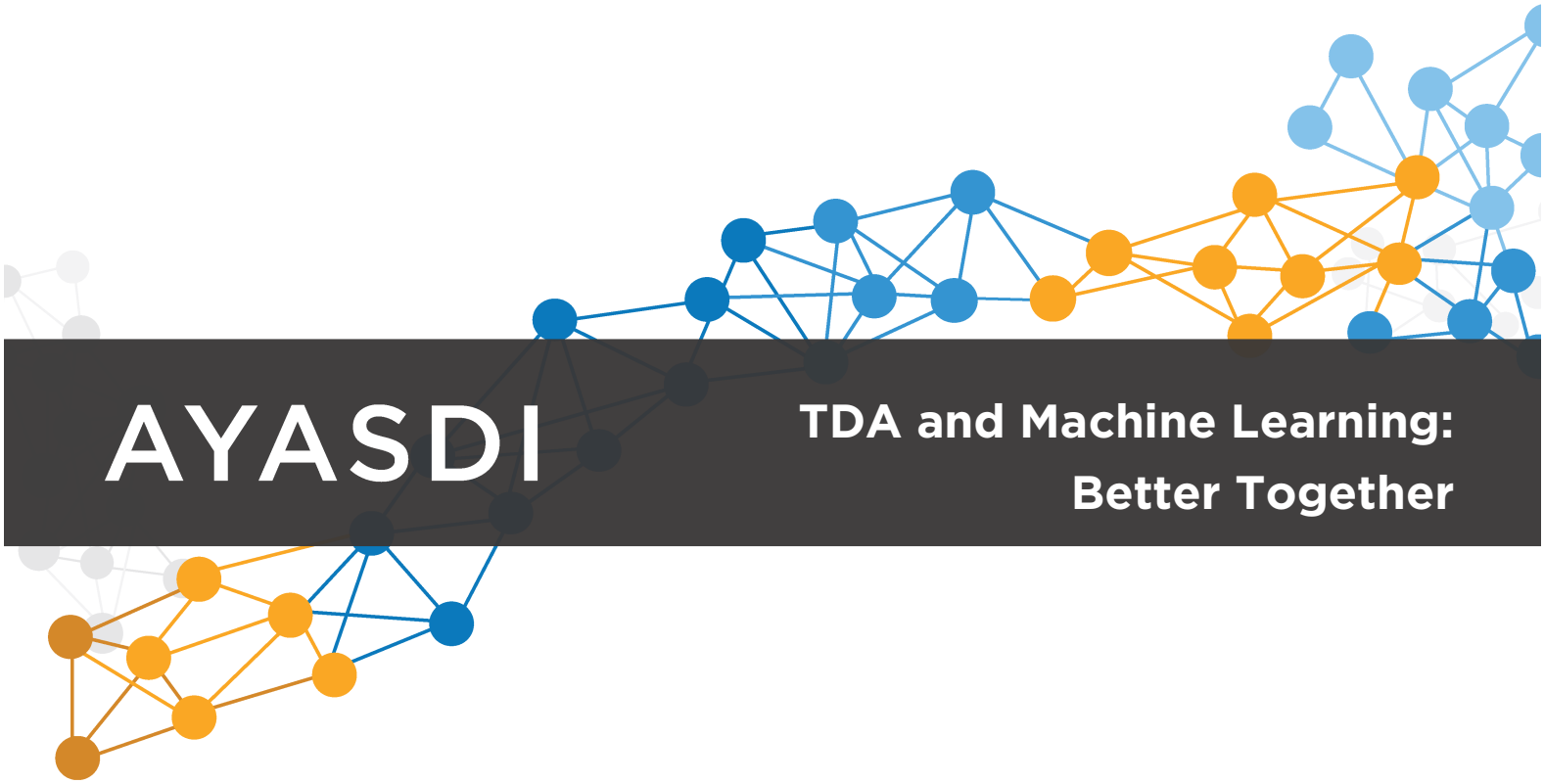

AYASDI

TDA and Machine Learning: Better Together

TABLE OF CONTENTS

The New Data Analytics Dilemma

Now more than ever, organizations rely upon their data to make informed decisions that can affect millions of lives and billions of dollars of revenue. The collection and analysis of data from transactions, sensors, and biometrics continues to grow at a prodigious rate, taxing the analytic capabilities of even the most sophisticated organizations.

The quantity of possible insights in a given dataset is an exponential function of the number of data points. On top of this, aggregate data growth is an exponential function with time. Unfortunately, we cannot train enough data scientists to meet this runaway, double-exponential demand curve.

This is driving scientists and mathematicians to examine new approaches to improve both the quality and the speed of their analytics engines. Today's hypothesis-driven analytics will not suffice. High-performance machines and algorithms can examine complex data far faster and seek insights more comprehensively than ever before. However, we need to find exponential improvements in analysis techniques to meet the growing demand.

Introducing Topology and Topological Data Analysis

Topology is a mathematical discipline that studies shape. TDA refers to the adaptation of this discipline to analyzing highly complex data. It draws on the philosophy that all data has an underlying shape and that shape has meaning. Ayasdi's approach to TDA draws on a broad range of machine learning, statistical, and geometric algorithms. The analysis creates a summary or compressed representation of all of the data points to help rapidly uncover critical patterns and relationships in data. By identifying the geometric relationships that exist between data points, Ayasdi's approach to TDA offers an extremely simple way of interrogating data to understand the underlying properties that characterize the segments and sub-segments that lie within data.

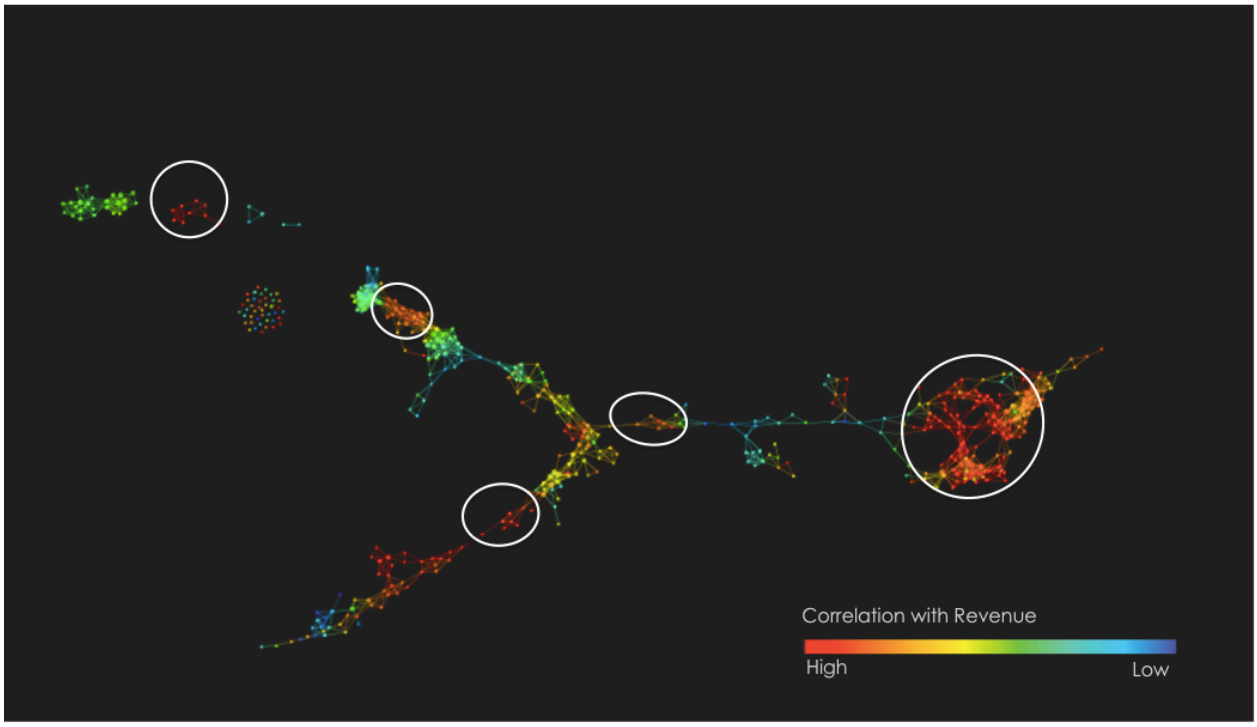

Figure 1: Creating a Compressed Representation of Data to Uncover Patterns and Subgroups of Interest

The Promise of Machine Learning

Machine learning is a class of algorithms that adjust and learn from data to take or suggest actions in the future. It promises to help companies achieve the following goals:

1. Effectively segment existing data
2. Identify the key attributes and features that drive segmentation
3. Find patterns and anomalies in the data
4. Precisely classify new data points as they arrive

There are two classes of machine learning techniques – unsupervised and supervised. Unsupervised learning helps with discovering the hidden structure in data. Supervised learning helps with the construction of predictive models. Innovations in these techniques promise to help drive new revenue streams, forge stronger customer relationships, predict risk, and prevent fraud. However, analyzing complex data using these methods is constrained by certain intrinsic issues as well as a dependency on scarce machine learning expertise.

Machine Learning – Mind the Gap

There are two types of unsupervised learning algorithms:

1. Clustering – These algorithms discover the underlying sub-segments within data by grouping sets of data points in such a way that those in the same group (called a cluster) are more similar to each other than to those in other clusters.

2. Dimensionality Reduction - These algorithms are especially useful for reducing the number of properties or attributes (data columns) required for describing each data point while retaining the inherent structure of the data.

UNSUPERVISED LEARNING - CLUSTERING

Clustering methods segment a dataset into smaller datasets. Different clustering algorithms draw on different techniques to cluster data. Take the example of a hierarchical clustering algorithm such as single-linkage clustering. In the beginning, each data point is its own cluster. The clusters are combined into larger clusters by sequentially fusing pairs that are the most similar to each other by some measure. The process continues until all the data points are fused into one large cluster. The clustering hierarchy can be visualized as a dendrogram that shows which clusters were fused together to produce new clusters. Knowing the sequence and distance at which cluster fusion took place can help determine the optimal scale for clustering.

In general, there are two key issues with clustering algorithms:

1. The Number of Clusters – Some clustering algorithms require that the number of clusters to be returned be determined in advance of applying the algorithm to the data. While a machine learning expert might use some informed criteria (such as a “Bayesian information score”) to make an educated guess at the number of clusters, typically this is an arbitrary choice that can greatly impact conclusions.

2. Continuous Data Sets - Clustering methods work well when data sets decompose cleanly into distinct groups that are well separated. However, many data sets are continuous and exhibit progressions rather than sharp divisions. Clustering methods can create spurious divisions in such data sets thereby obscuring the real structure.

UNSUPERVISED LEARNING – DIMENSIONALITY REDUCTION

Dimensionality Reduction methods make it easier to visualize data sets that have a large number of data columns. For example, consider credit card transactions that have thousands of attributes that are represented as data columns. Visualizing these transactions can be extremely difficult given that we cannot see more than three dimensions at a time. Principal Component Analysis (PCA) is a good example of Dimensionality Reduction. Other examples of Dimensionality Reduction methods include Multi-dimensional Scaling, Isomap, t-Distributed Stochastic Neighbor Embedding, and Google's PageRank algorithm.

Dimensionality Reduction methods are extremely powerful as they can reduce the number of dimensions required to describe data while still revealing some inherent structure in that data.

However, there are two issues with Dimensionality Reduction methods:

1. Projection Loss – Dimensionality Reduction methods compress a large number of attributes down to a few. As a result, data points might appear in clusters that they are not actually a part of. Distinct clusters might overlap. This increases the chances of missing out on subtle insights.

2. Inconsistent Results - Different Dimensionality Reduction algorithms produce different projections because they encode different assumptions. None of the results are wrong; they are just different as different algorithms accentuate different aspects of the data. Choosing a single algorithm might result in missed critical insights.

SUPERVISED LEARNING - REGRESSION AND CLASSIFICATION

Supervised learning algorithms are used for producing predictive models. There are two types of supervised learning algorithms:

1. Regression algorithms
2. Classification algorithms

Regressors predict real-valued variables (e.g., profit margins, stock prices). Classifiers predict discrete variables (such as fraud or customer churn). Examples include Linear and Logistic Regression, Support Vector Machine, and Artificial Neural Networks.

There are two phases in supervised learning:

1. Training - In this phase, the algorithm analyzes a training data set to produce parameters for a function that is assumed to represent the data. This phase needs historical data for which the predictions are known ("ground truth").

2. Prediction - In this phase, the inferred function that was produced in the training phase is used to predict the values for new data points.

Supervised learning algorithms have certain inherent issues that need to be taken into consideration:

1. Assumptions - The choice of algorithm entails an assumption about the shape of the underlying data. For example, linear regression assumes that the data is planar (possibly higher dimensional) and tries to find the best plane that fits the data. If the actual shape of the underlying data is not planar, then the analysis will produce incorrect results. There is a heavy reliance on a machine learning expert knowing which algorithm to choose.

2. Global Optimization - All supervised algorithms try to find parameters for a function that best approximate all of the data. However, data is rarely homogeneous. It is unlikely that there is a single shape that fits the entire dataset.

3. Generalization - A model may perform well with test data, but produce inaccurate results with new data. This is known as a generalization error and it occurs because the model has more parameters than are actually required. This issue is also known as overfitting.

WHAT IS MISSING

1. Successful implementations require experts in Machine Learning, an increasingly scarce resource to find.

2. It is easy to miss important insights in data by choosing the wrong algorithm or by not trying enough algorithms.

3. Each class of machine learning algorithms has it own set of intrinsic issues that need to be taken into account.

The next section details how TDA enhances standard machine learning methods.

How TDA Improves Machine Learning Algorithms

TDA makes machine learning algorithms dramatically more effective. All machine learning methods produce functions or maps. For example:

1. Clustering maps an input data point to a cluster.
2. Dimensionality Reduction maps an input data point to a lower dimensional data point.
3. Supervised Learning algorithms map an input data point to a predicted value.

TDA uses these maps or functions as input to produce a superior output.

UNSUPERVISED LEARNING - CLUSTERING

TDA uses clustering as an integral step in building a network representation of data. As opposed to trying to find disjoint groups, TDA applies clustering to small portions of data. It then combines these “partial clusters” into a network representation that gives an overview of the similarity between the data points. This makes TDA more appropriate for constructing a connected representation of continuous data sets and data with heterogeneous densities.

UNSUPERVISED LEARNING - DIMENSIONALITY REDUCTION

TDA supports the automatic execution and synthesis of Dimensionality Reduction algorithms. The key benefits include the following:

1. TDA eliminates the projection loss issue typical of Dimensionality Reduction methods wherein data points that were well separated in higher dimensions end up overlapping in a lower dimensional projection. TDA achieves this by clustering the data in the original high dimensional space. As a result, data points that were

well separated in the original space will typically be well separated in the TDA output. This enables the easy identification of distinct segments and sub-segments within data that might have been missed using Dimensionality Reduction methods.

2. TDA is able to synthesize the results of various Dimensionality Reduction algorithms into a single output. This eliminates the need to know or guess the correct sets of assumptions for a Dimensionality Reduction method.

SUPERVISED LEARNING - REGRESSION AND CLASSIFICATION

TDA augments supervised learning algorithms in the following ways:

1. Eliminates systematic errors - Most supervised learning algorithms are based on global optimization. They try to assume a shape for the underlying data and then find the parameters that best approximate all the data, thereby making mistakes in some regions. TDA uses the output of these supervised algorithms as an input to discover areas of the underlying data where errors are being made systematically.

2. Optimized for local data sets - As opposed to making global assumptions regarding all the underlying data, TDA effectively constructs a collection or ensemble of models. Each model is responsible for a different segment of the data. This eliminates the need to create a single model that works well on all of the data. A collection of models can be much more accurate. This approach works for any supervised algorithm.

TDA reduces the possibility of missing critical insights by reducing the dependency on machine learning experts choosing the right algorithms. It often uses current machine learning techniques as input to find subtle patterns and insights in local data. In general, TDA enhances any algorithm that it is paired with.

Creating Ayasdi Core Networks with TDA

TDA identifies data points that are related to each other. It then pieces these regions of data together to build a global, compressed summary of the data in the form of a network. Ayasdi Core uses a function on the data (call it *f*) and a measure of similarity to generate compressed representations of the data. The resulting visual network consists of nodes that represent data points with similar function values and that form a cluster based on a measure of similarity.

Consider two simple examples to illustrate how Ayasdi Core creates networks. The first example steps through the general methodology and the second example demonstrates how Ayasdi Core enhances machine learning.

Take a data set that is represented by a circle in the *xy*-plane. We will then use a function *f* that maps each point in the data set to its *y*-coordinate value (Figure 2).

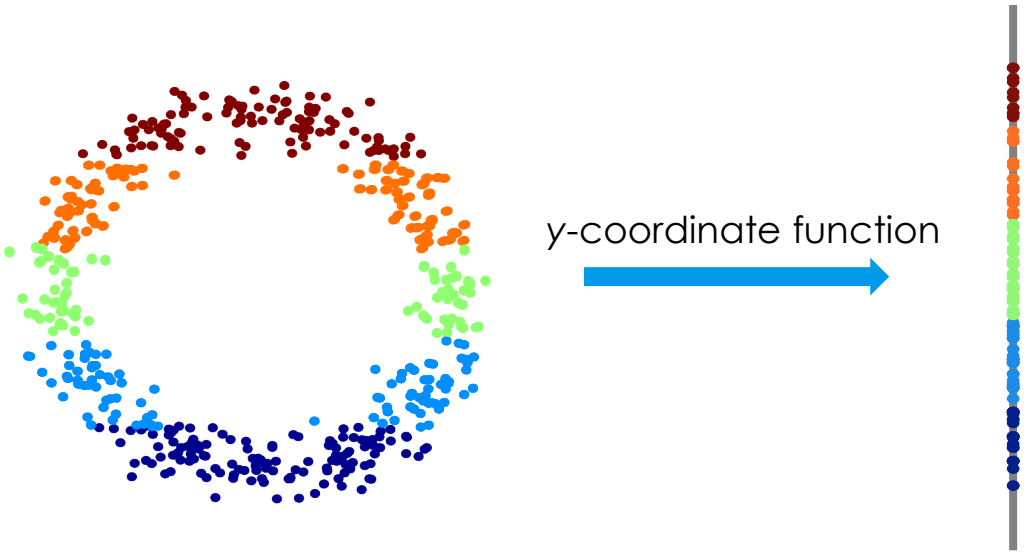

Figure 2: Using a Function to Map Data Points in the Shape of a Circle to their *y*-Coordinate Values

Ayasdi Core subdivides the image of the function into overlapping sets of nearby values. In this example, the points are divided into four overlapping groups that have similar *y*-coordinate values (Figure 3).

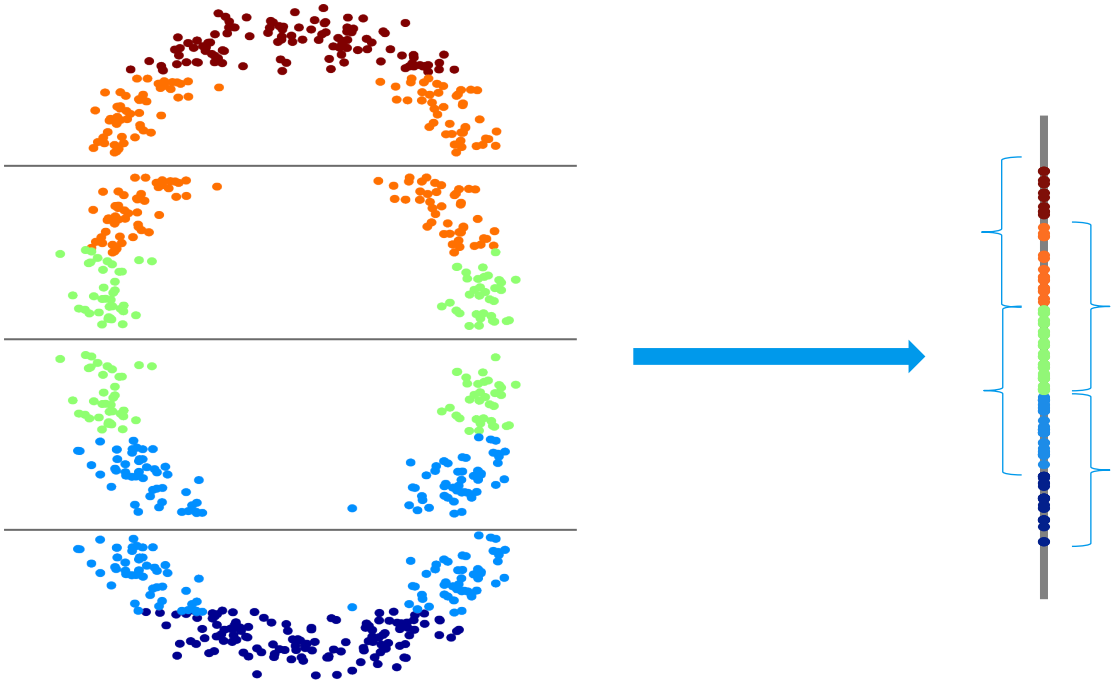

Figure 3: Dividing Data Points into Overlapping Sets with Similar *y*-Coordinate Values

Next, Ayasdi Core clusters each group of data points independently using a measure of similarity. In this example, similarity is defined using the standard Euclidean distance. Each cluster is represented as a node. A node represents a set of data points that are similar with respect to the measure of similarity (Euclidean distance) and the function value (*y*-coordinate). The size of the node reflects the number of data points within. Notice that the top node represents both red and orange data points (Figure 4). The second set from the top containing two distinct regions of data points produces two separate nodes.

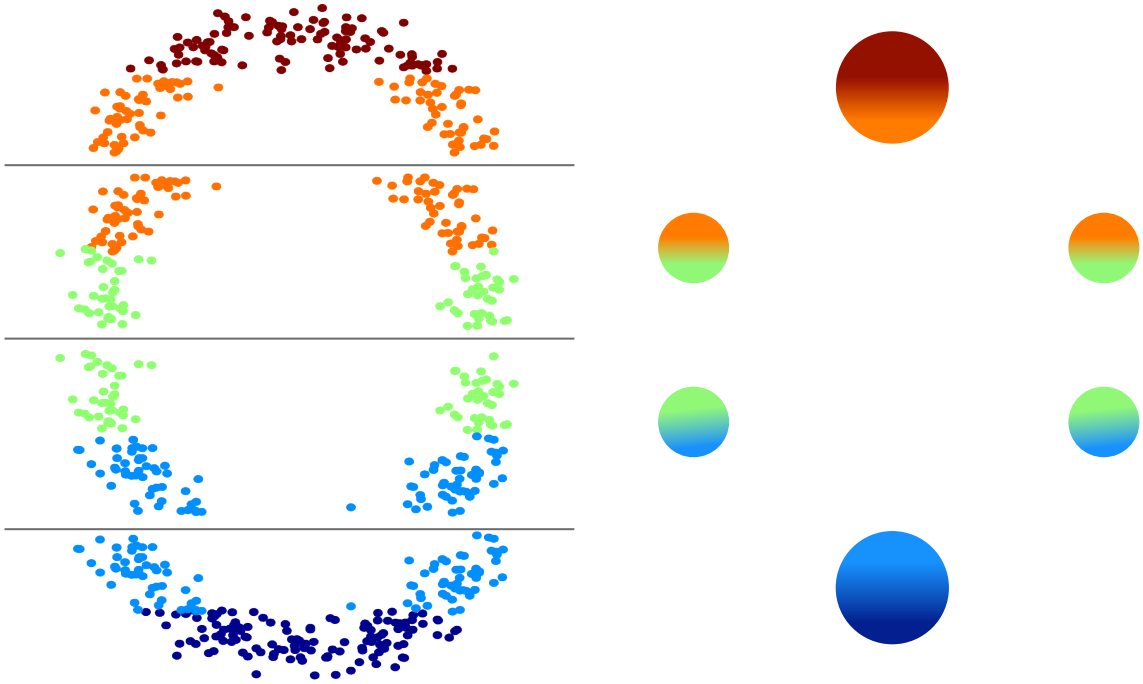

Figure 4: Nodes Represent Clusters of Data Points with Similar Function Values and Measures of Similarity

Nodes with data points in common are connected by edges in the Ayasdi Core network. Since the data set was divided into overlapping sets, a data point can be represented in multiple nodes. In this example, the orange data points on the left are represented in both the top red node as well as the orange node on the left (Figure 5). These nodes are connected by an edge because they contain data points in common.

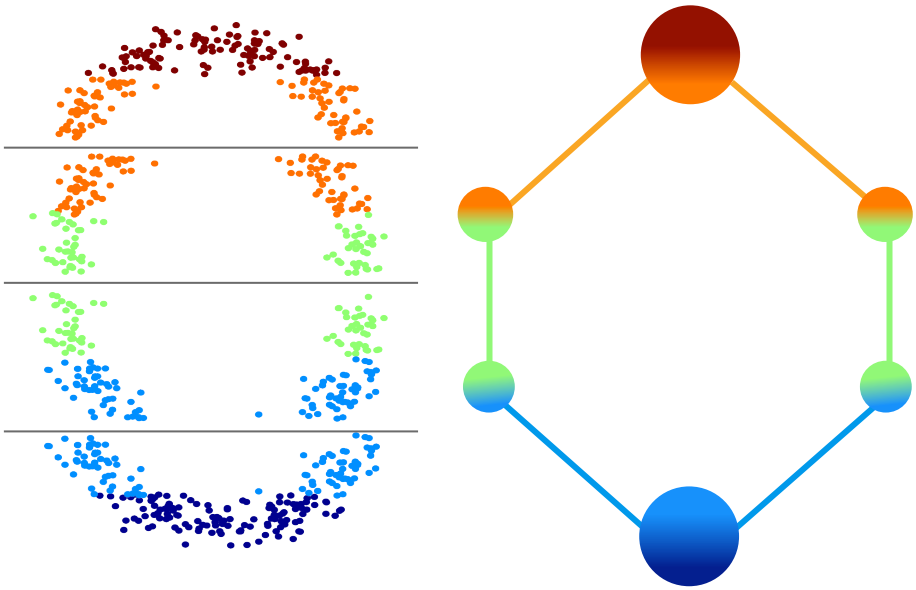

Figure 5: Nodes with Data Points in Common are Connected by Edges to Form a Network

The resulting network is a compressed representation of the original data set that retains its fundamental circular shape. The network is much simpler to visualize and work with than the original data, yet it captures the essential behavior of the data.

In the second example, we will examine a data set that is sampled in the two-dimensional Euclidean plane from four Gaussian distributions. In Figure 6, we color the data points by the values of the density estimator function. Ayasdi Core then divides the data set into overlapping sets with similar function values (in this case, density estimations). Each subset of the data is clustered to create nodes that represent data points with similar function values and measures of similarity.

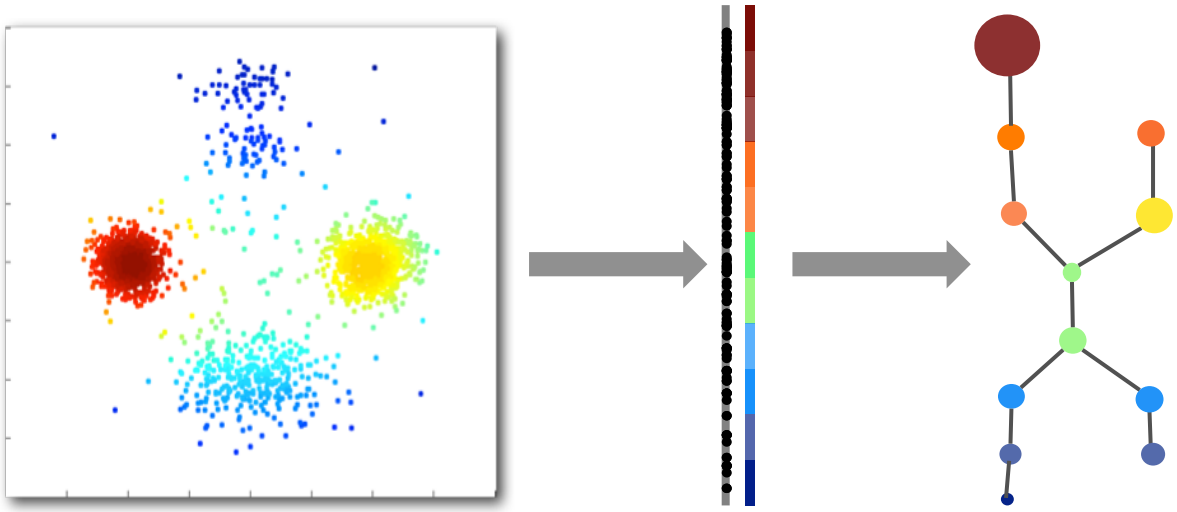

Figure 6: TDA Enhances Machine Learning By Capturing the Overall Structure and Fine-Grained Behavior

The resulting network captures both the overall structure of the data as well as its fine-grained behavior (Figure 6). The four flares in the network correspond to the four regions of varying densities. The flares in the network are connected to each other because of the data points that are common to these regions with varying degrees of density.

Standard machine learning techniques would have identified the four regions but they would have lost the continuous transitions between them. Ayasdi Core captures both the differences and the similarities in the data.

Complex data holds useful information that could go undetected when using standard machine learning and statistical techniques. Ayasdi Core begins by understanding data at a small scale. It then stitches together these pieces of information to create a topological summary or compressed representation of the entire data set. The networks surface the subtle insights in the data while also representing the global behavior of the data.

Ayasdi Core can draw on the power of the function *f* to incorporate virtually any machine learning, statistical, or geometric technique into the creation of compressed representations of the data visualized as networks. Principal component analysis, autoencoders, random forests, and density estimators are some examples of functions *f* that Ayasdi Core uses to derive insights from your data. In this way, TDA is a framework for advanced analytics.

Exploring and Using Ayasdi Core Networks to Understand your Data

Ayasdi Core uses TDA to create a visual representation of data in the form of a network. A network comprises of the following:

- Nodes that represent collections of similar data points
- Edges that connect nodes that share data points

TDA helps automatically discover these networks that reveal the underlying structure of a data set.

The networks produced by TDA are simple, yet extremely powerful representations of the data. The following section outlines various techniques for exploring, understanding, and using the insights uncovered from data.

1. Exploring Data
    a. Visualize - Ayasdi Core presents the output of TDA as an interactive visual network. Tugging and pulling at the nodes changes the orientation of the network. Changing the appearance of a network on the screen by moving nodes around or changing their colors does not impact the insights it represents. In fact, visualizing a network can help with selecting regions of interest and creating node groups that can be inspected further using the “Explain” and “Export” operations.

    b. Color – The color of the nodes and edges of a network can be changed to allow for the quick exploration of data. When coloring a network by a particular data column, Ayasdi Core computes the mean value of the specified data column for all the data points within each node individually. It then maps all these values to a color palette.

    c. Find - With Ayasdi Core, specific conditions can be applied to a data column that evaluate to either true or false. For example, to find all your customers whose “Net Worth” (the data column) is greater than $500,000, the software creates a color scheme that highlights nodes with data points that meet this condition.

    d. Contrast - Ayasdi Core also lets you explore the differences between two specified color schemes.

2. Understanding and Making Use of the Insights
    a. Explain - Ayasdi Core uses the "Explain" operation to help you find the data columns or attributes that differentiate node groups. It runs a wide array of tests (e.g., KS tests, T-test, P-value, hypergeometric enrichment). It returns a list of data columns that are ordered by their statistical power in differentiating the specified node groups.

    b. Resolution - Ayasdi Core provides a multi-resolution view of data that supports the discovery of subtle, otherwise hard-to-find signals in data.

    c. Export - The software also allows for the export of data points along with the associated list of points that belong to a node as file for use in downstream operations (e.g., to view using a BI tool).

How Ayasdi Core Uses TDA to Make Complex Data Useful

SEGMENTATION

Data segmentation involves grouping data points that are more similar to each other in comparison with the remainder of the data. The most common approaches involve either a data scientist manually generating and testing hypotheses or the use of clustering algorithms.

The manual testing of hypotheses can be a huge undertaking even when dealing with small data sets. Typically, a domain expert starts by choosing a logical attribute of the data to create segments. However, while segmenting customers by their spend, for instance, might seem like a good idea, it ignores the impact of other factors such as demographics.

In comparison, standard clustering methods for segmentation produce better results. However, these methods still suffer from the issues described earlier such as the need to know the number of clusters in advance of applying the algorithm and its unsuitability for tackling continuous data sets.

For instance, a financial institution can profit from segmenting their clients by their investment behavior under specific market conditions and then precisely targeting them with tailored recommendations, at the right time. Typically, there are macro-trends in client buying behavior. However, there can also be subtle trends hidden in the data that are driven by specific regional events. For instance, such an event might result in a particular group of clients trading in a specific class of products. In addition, their response is more likely on a continuum, with some clients responding more significantly to the event than others. If the number of clients in this particular region is small compared to the total number of clients in the data, given that they also respond to the same broader market conditions as the rest of the investors, this subtle regional trend will likely be ignored by standard clustering methods.

Ayasdi Core, on the other hand, would discover that while these regional investors are similar to the majority of the clients in the data, they are much more similar to each other. This subtle signal would be captured in Ayasdi Core as a flare in the network thereby informing the bank's sales force of this subpopulation's presence. Moreover, the flare would capture the continuum of responsiveness within this regional group, enabling the sale force to prioritize and target highly responsive clients with tailored recommendations, ahead of those that are more similar to the majority of the clients in the data.

FEATURE DISCOVERY

Understanding the underlying features or attributes of the data that drive segmentation can be invaluable when it comes to pinpointing the factors that impact business outcomes. Ayasdi's software helps with feature discovery by automatically producing a list of the attributes (data columns) that drive segmentation, ranked in order of statistical significance.

Take the example of using Ayasdi Core to understand the reasons for customer churn. While the ability to predict churn is useful, being able to get to the root causes for churn is significantly more important as it often brings systemic issues to the surface.

Identifying the features that result in customer churn with Ayasdi Core involves the following steps:

1. Construct a data set with data columns (in this case, customer attributes) of interest. Optionally create an outcome data column by which data can be segmented. In this example, an output data column tracks whether a customer churned or not.

2. Segment the data set using all the data columns. In this case, the output data column that tracks customer churn serves as an additional data lens through which data is viewed.

3. Create clusters of data points or node groups that form a network, in this case, using the outcome data column for tracking churn.

4. Use the "Explain" operation in Ayasdi Core to get a tabular listing of the underlying features or attributes of the node groups that represent customers that have churned, in statistical order of importance.

CLASSIFICATION

Recommendation engines are designed to help organizations drive more revenue by precisely targeting customers with products and services that other customers with similar profiles purchased in the past. Ayasdi Core serves as an ideal foundation for a recommendation engine. Using Ayasdi Core to deliver tailored recommendations involves the following steps:

1. Create precise sub-segments of a customer base by correlating and analyzing a wide range of client-related data including demographic, buying behavior, market, CRM, and social media information (Figure 7).

2. Assign all newly arriving customer data points to a specific node or group of nodes (sub-segments).

3. Look up the buying behavior of the other customers that are represented in these sub-segments.

4. Present tailored recommendations based on what similar customers have bought to the sales force or directly to customers via a dashboard or through targeted alerts.

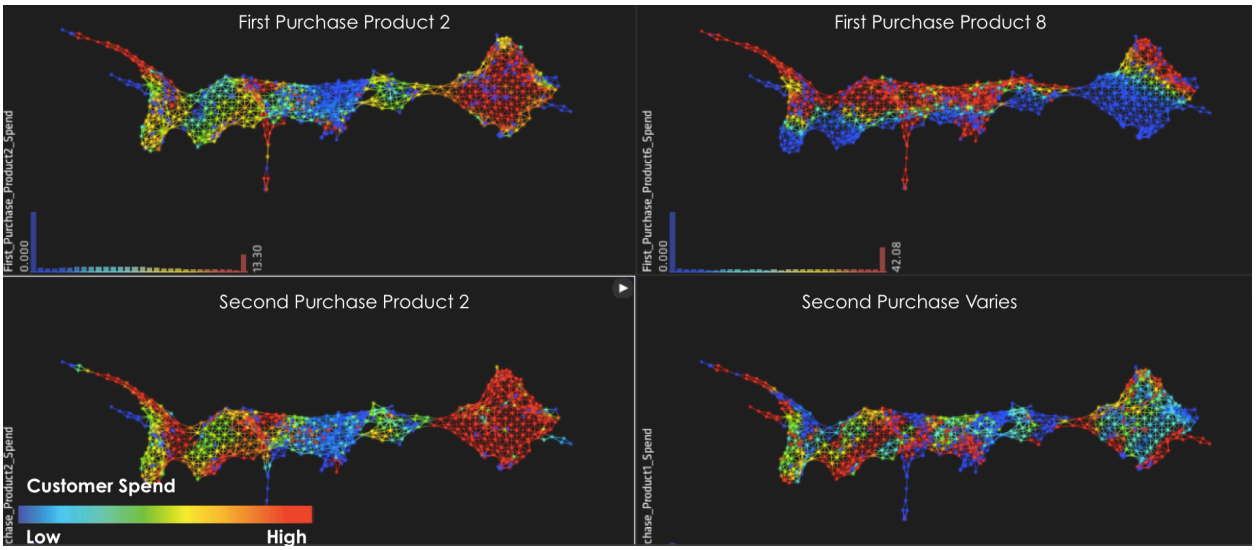

Figure 7: Analyzing Returning Customers by Buying Patterns and Spend

MODEL CREATION

Supervised learning methods are typically employed to create models that can predict future actions or behavior. Ayasdi Core leverages TDA to support the creation of a collection of models, referred to as piecewise or ensemble models, that best represent all the data. These models tend to be far more accurate as they are each optimized for different segments of the data.

An example workflow for creating a model with Ayasdi Core involves the following steps:

1. Construct a data set with data columns (attributes) of interest as well as an outcome data column.
2. Segment the data set without using the outcome data column.
3. Create node groups within the network.
4. Create a simple, distinct model for each node group using standard supervised learning methods like linear regression.
5. Use the model associated with each node group to accurately predict the placement of newly arriving data points within these node groups.

MODEL VALIDATION

Most organizations rely on a plethora of automated models to help with fraud detection, compliance, regulatory risk management, network security, and client relationship management. These models range from simple rule-based systems to those that are the results of supervised learning algorithms. One of the primary steps involved in validation or auditing exercises is the discovery of systematic errors or biases in the model. Typically, models created by supervised learning algorithms produce systemic errors as a result of incorrect assumptions about the shape of the underlying data. Ayasdi Core uses TDA to uncover these errors in models.

Consider the process of validating models used to detect fraud in credit card transactions. Identifying issues in these models using Ayasdi Core involves the following steps:

1. Construct a data set where each data point is a transaction. Create two additional data columns:
    a. The predicted outcomes from the model
    b. The actual ground truth - were the transactions fraudulent or not?

2. Segment the transaction data using all but the columns that track predicted outcomes and the ground truth.

3. Color the network by both the model estimation and the ground truth.

4. Focus on the subgroups of transactions in the network where the model made mistakes.

5. Use the "Explain" operation in Ayasdi Core to get a list of the data columns (features) associated with these subgroups. This helps identify combinations of features that indicate fraud that previously went undetected.

ANOMALY DETECTION

The traditional approach to detecting new patterns of fraud can be manually intensive. These manual investigations often result in the creation of new fraud rules that then need to be incorporated into the fraud detection models.

Ayasdi Core uses TDA to help automatically detect patterns of fraud in data. Detecting anomalies using Ayasdi Core involves the following steps:

1. Construct a data set of transactions. Unlike model validation exercises, anomaly detection does not require knowledge of the ground truth or any other information from the current models

2. Segment the data set based on all data columns

3. Explore regions of the network that represent low density points or points far away from the central core of the data set.

SUMMARY

While organizations have successfully tackled the challenge of storing and querying vast amounts of data, they continue to lack the necessary tools and techniques for extracting useful information from highly complex data sets. Topology and TDA are well suited for analyzing complex data with potentially millions of attributes. Ayasdi Core uses TDA to bring together a broad range of machine learning, statistical, and geometric algorithms to create compressed representations of data. This advanced analytics software creates highly interactive visual networks that allows them to rapidly explore and understand critical patterns and relationships in their data. Ayasdi Core's use of TDA augments current machine learning techniques by ameliorating some of their intrinsic issues and reducing the dependency on increasingly scarce machine learning expertise. Innovative companies have used TDA and Ayasdi Core to 1) precisely segment their data; 2) identify the underlying features that drive segmentation; 3) create more effective predictive models and tailor product recommendations; 4) develop, validate and improve models; and 5) detect subtle anomalies in their data.

ABOUT AYASDI

Ayasdi is on a mission to make the world's complex data useful by automating and accelerating insight discovery. Our breakthrough approach, Topological Data Analysis (TDA), simplifies the extraction of intelligence from even the most complex data sets confronting organizations today. Developed by Stanford computational mathematicians over the last decade, our approach combines advanced learning algorithms, abundant compute power and topological summaries to revolutionize the process for converting data into business impact. Funded by Khosla Ventures, Institutional Venture Partners, GE Ventures, Citi Ventures, and FLOODGATE, Ayasdi's customers include General Electric, Citigroup, Anadarko, Boehringer Ingelheim, the University of California San Francisco (UCSF), Mercy, and Mount Sinai Hospital.

AYASDI

CONTACT US

Ayasdi, Inc.
4400 Bohannon Drive
Suite #200
Menlo Park, CA 94025

sales@ayasdi.com
visit ayasdi.com

@ayasdi