## Intro

Software is eating the world, and AI is eating software. For a technology that is so new (commercial software has only been around for about 30 years), traditional software development is already becoming obsolete. Machine learning, a newer technology, automates what currently takes humans a significant time to accomplish in code. While machine learning appears self-sustaining, it requires significant development and maintenance from a team of highly skilled personnel. The implementation of machine learning today still takes human expertise, and data scientists are in high demand. The next evolution in human intelligence will be automating the creation of machine learning models, or creating AI that designs AI systems. This will broaden machine learning accessibility and lower the technical barrier to complex solution development.

SparkCognition™ has developed a platform that represents the next stage in machine intelligence. Darwin™ is a machine learning product that accelerates data science at scale by automating the building and deployment of models. Darwin provides a productive environment that empowers data scientists with a broad spectrum of experience to quickly prototype use cases and develop, tune, and implement machine learning applications faster than traditional data science methods. Through a blend of evolutionary algorithms and deep learning methods, Darwin accelerates the iterative improvement of models while allowing different degrees of control for customization.

Darwin is a major advancement from traditional machine learning, which operates via manual data cleaning, feature generation, and parameter tuning. These traditional approaches run into several problems: They take time to design and implement, often struggle to scale across large operations, and can be limited by edge cases that occur under extreme or unusual operating parameters. By contrast, Darwin creates dynamic models that are highly accurate and take far less time to develop.

For a better sense of what Darwin is capable of, here are some types of problems that Darwin is designed to solve.

| TYPE OF PROBLEM | DESCRIPTION | EXAMPLES | LEARNING TYPE |
|---|---|---|---|
| **Classification** | Based on a series of inputs, predict a categorical output | Given the features of a mammogram, identify if a patient's tumor is cancerous | **Supervised** |
| **Regression** | Based on a series of inputs, predict a numerical output | Predict the stock price of Microsoft, given the stock prices of Amazon, Walmart, and Twitter | **Supervised** |
| **Forecasting** | Given information about historical events, predict them in advance before they occur | Predict that a critical component in a turbine is trending towards failure | **Supervised** |
| **Clustering** | Finds groupings of data that are similar to each other | Identify the changing operating modes of an offshore oil rig over time | **Unsupervised** |
| **Anomaly Detection** | Identifies data points differentiated from the majority by a significant factor | Find strange user traffic in cyber networks to pinpoint malicious actors | **Unsupervised** |
| **Normal Behavior Modeling** | Given input data over time, build a model of the normal operating state of a system. Identifies behavior over time which deviates from normal | Predict that a critical component in a turbine is trending towards failure | **Semi-supervised** |

## How Does Darwin™ Work?

Darwin automates four major steps in the data science process: cleaning, feature generation, feature selection, and the construction of either a supervised or unsupervised model.

### Cleaning

Data sets are far from homogenous—they come in different file formats and can contain missing data as well as numerous data types. Because of this, Darwin needs to convert any data set into a usable form for algorithmic development.

It is important to handle data in a variety of different forms, including categorical, numerical, and date/time. Darwin converts all of these data types into a single, usable format. Darwin will automatically analyze categorical data and use techniques such as dummy variables (e.g., "0" for "no" and "1" for "yes") or one-hot encoding, a process where a data point is represented using multiple dummy variables with binary traits.

Darwin is also able to work with date/time data by allowing the user to specify a date/time index. The database is then sorted based on that index, preserving temporal data for use in time series problems. Then, for all of the date/time information included in the input data, Darwin will automatically extract powerful features which allow the model to easily learn temporal relationships.

The next step in data cleaning is scaling. Variables within a data set may exist on different scales. For instance, a person may analyze how the daily ambient temperature might change as a function of distance from the sun. One of these variables is measured between -10 and 40 degrees Fahrenheit, and the other variable is measured in millions of miles. Within a function, the two variables are difficult to compare and use together. That is why scaling is useful. Scaling normalizes data sets so that the features are easier to compare to one another. This makes the job of the machine learning algorithm easier and more stable.

### Feature Generation

Once data has been cleaned, data scientists manipulate that data to generate more appropriate features to solve a particular problem. This can be a difficult process, as there is a massive search space for data windowing, new feature selection, and hyperparameter optimization. Darwin can generate more features to drive toward a better solution.

One of the biggest challenges in handling dynamic time series data is determining how to window the time steps for feature generation. How this is done can critically impact not only the time series-derived features, but also the features generated within the Fourier spectrum. Darwin automates this windowing process using one-dimensional convolutional neural networks (CNN). CNNs are a class of deep neural networks. They use a type of multilayer perceptions designed to need only minimal preprocessing. The network instead automatically learns the filters that traditionally would need to be engineered by hand.

Windowing with CNNs is automated, and the user does not need to define inputs. The one-dimensional CNN automatically adjusts the window size for the time frame being used. It uses filters that learn how to extract important features automatically over different windows of data rather than requiring manual specification.

### Feature Selection and Model Building

Once automated cleaning and feature generation have taken place, the data set is ready to be used to build a model. Darwin is capable of building two types of learning models: supervised and unsupervised. Per the examples in Table 1, these methods differ in how they work and the problems they solve. Darwin takes a unique approach to each type of problem in order to achieve an optimal solution.

#### *Supervised Learning Models*

For supervised learning problems, the goal of Darwin is to ingest the cleaned input data and automatically produce a highly optimized machine learning model which can accurately predict a target of interest specified by the user. Darwin accomplishes this using a patented evolutionary algorithm which simultaneously optimizes and compares various machine learning models, from decision tree-based methods such as the Random Forest and XGBoost to state-of-the-art deep learning models.

Darwin begins by analyzing characteristics of the input dataset and the specified problem, and then applying past knowledge to construct an initial population of machine learning models which are likely to produce accurate predictions on the problem. In the case of neural networks, these models are then passed to an optimizer which judiciously applies popular techniques such as backpropagation to train the models.

Darwin then employs principles from evolutionary biology to further optimize the models. Elite models in the population which provide the highest accuracy on the input problem are identified, and the traits from these models are combined to yield even better models. Throughout this process, models are segmented into groups, or species, which allows Darwin to preserve diversity and avoid local optima. This process continues over a user-specified time window.

Darwin also knows how to handle your data. It will automatically segment your uploaded data into train and validation splits during the model creation process. If your data includes a time component, then the ordering of samples is maintained, and if your data contains more than one timestamp, Darwin provides advanced options which allow you to specify sorting indices. If your data is not time series, Darwin will employ techniques such as random or stratified shuffling to ensure that your model is trained and evaluated using stable distributions of data. Additionally, Darwin computes and reports performance metrics on holdout data, so that the user can feel confident that the behavior will generalize well to new and unseen data. All of these steps ensure that Darwin will produce a machine learning model highly optimized to the specified supervised learning problem. Since Darwin starts simple and adds complexity over time, the models produced are as simple as possible while also providing the highest accuracy on the given problem.

#### *Unsupervised Learning Models*

For unsupervised modeling, Darwin first performs anomaly detection using varying techniques, depending on whether the data is time series or not. Once anomaly detection is performed, users have the option of clustering their data. This process is the same for both time series and non-time series data.

To cluster data, Darwin first labels points in a data set by using a standard k-nearest neighbors algorithm, which pairs each point

with its closest neighbors in the set. The points are given a numeric label based on their degree of similarity.

The data is then fed through a neural network trained to determine whether two data points are similar or dissimilar. This network makes use of SpectralNet, a neural network version of spectral clustering.

SpectralNet approximates the relative similarity of any two data points using a multilayer neural network. This network simultaneously approximates data similarity, extracts relationships, and projects the data into a lower dimension. This process has several benefits over standard spectral clustering: It can be saved and continuously improved with future incoming data, and its neural architecture can be modified to accommodate more complex data structures.

Once the data has gone through SpectralNet, a standard Gaussian mixture model is employed to fit the data to multiple Gaussian distributions, forming clusters. However, at this point there will be data points that do not cluster, and that can therefore be determined to be anomalies that do not belong to any normal operating state.

### *Normal Behavior Modeling*

Normal behavior modeling is a powerful tool that improves on the traditional risk index to allow for anomaly detection and prediction of impending asset failures. It is a semi-supervised process in which the user inputs time series data of, for example, sensor data from a specific asset in the time leading up to a failure.

Darwin analyzes the data from before the failure occurred using an autoencoder. An autoencoder is a neural network-based approach towards dimensionality reduction. Autoencoders compress data to reduce the feature set to the smallest size possible, and then decompress it with as little loss as possible. For example, an autoencoder might take a set of 100 inputs X1, X2,...X100, encode them to a lower-dimensional latent space (Y1,Y2,...Y10), and then decode them back to reproduce the inputs as the network outputs (X1, X2,...X100).

Like any other neural network, autoencoders have numerous hidden layers, a defined latent space, and different activation functions in their encoding/decoding process. Once the autoencoder's architecture is defined, the network can undergo backpropagation with dropout to reduce the output loss via weight optimization.

The autoencoder takes a data point of some time or window of time from the pre-failure data, compresses it, and then reconstructs it. The autoencoder thus learns to accurately reconstruct data representing normal functionality of the asset. The difference between x, the original data point, and x1, the reconstructed data point, is represented by a value known as the residual.

For the normal behavior data the model has trained on and is skilled at compressing and reconstructing, the residual will be low, meaning there is little difference between x and x1. For data points closer to the failure, where there are greater deviations from the norm and the data looks different than what the model trained on, the residual will increase.

The residual's significance is determined using a Hotelling's T-squared distribution, and these analyses are then presented to the user as a quantitative evaluation of how close the given data point is to normal operation. This information can be used to identify anomalous operations and predict asset failures in advance.

## Conclusion

Darwin amplifies a data scientist's work by enabling more effective model building. It gives data scientists the ability to iterate through thousands upon thousands of different models and architectures in rapid fashion. Darwin enables organizations to build, deploy, and maintain all of their models from a single configuration-based environment, allowing data scientists and non-data scientist model builders alike to focus on their data problems instead of parameter tuning within models. By leveraging the best of genetic algorithms and deep learning, Darwin provides the next step into the future of artificial intelligence.

***FIGURE 1***
*How an autoencoder works*