Datameer®

White Paper

Creating a Faster Cooperative Data Science Workflow with Datameer and Amazon SageMaker

Introduction

If you're going to do machine learning work right, you're going to need well-honed data sets to build your models on. It is not just about cleaning up the data. It is also about discovering data that is relevant to the problem at hand, finding more data to increase accuracy, integration data from multiple sources, aggregating values etc.

Four of the seven stages in a data science lifecycle involve getting intimate with your data (see image). But in terms of time, these four stages can consume 90% or more of the data science analytic lifecycle.

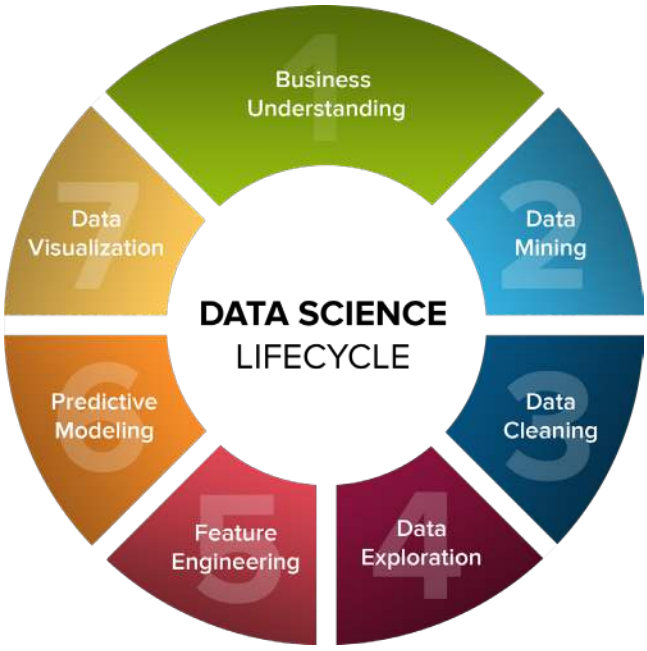

Figure 1: Data Science Lifecycle

And getting there isn't just about preparing the data. It's about exploring it and understanding it – achieving intimacy with the data, grasping its content and meaning in a fundamental way. It takes way more than simple preparation to do that. It takes deep analysis and exploration. And that requires data preparation, data exploration and machine learning work to be done in tandem and in harmony.

The good news here is that the integration of analytics and machine learning can be done relatively easily and effectively in a practical, matter-of-fact fashion. We'll illustrate how to create a more efficient and effective ML workflow that can be done with a rich data preparation and exploration platform – in this case Datameer – combined with a machine learning platform – in this case Amazon SageMaker. We'll highlight the specific features of the Datameer platform that complement the machine learning process in SageMaker,

how a strong symbiosis between Datameer and SageMaker can be fashioned to create a faster workflow that produces better ML models and results.

We'll also show how the cooperative relationship can work both ways – with data being sent from Datameer to an ML platform to build and test models, then the results of those tests and predictive output from the models fed back into Datameer for further exploration. We'll show how this loop can repeat iteratively – setting up a virtuous analytics-ML workflow cycle.

Today's ML Workflow

In the world of data science today, much of the work tends to happen inside a programming and data manipulation environment called a notebook. Notebooks are Web browser-based documents that contain a combination of text (in markdown format, specifically), code, and output generated by the code.

Often times that code will be written in Python, which is a very popular language for data science work. The Python code can then be applied to different platforms and APIs containing various algorithms in the ML platform (in our case SageMaker).

Notebooks and data science tools can be used to explore, query, profile and transform data. But data scientists must write code. This has three important consequences:

1. Only data scientists or data engineers, with the right coding skills, and sufficient file and database permissions can perform these tasks
2. Coding can be error prone and, in the best case, is far less efficient in preparing and exploring data for data science
3. It is difficult to get reuse and collaboration in such a coding environment to help the overall team be more productive across all analytic projects

With this in mind, a more efficient workflow would be to use a platform specifically designed to accelerate the difficult preparation and exploration tasks – finding the right data, exploring it to understand its' relevance to the problem, shaping it in the right way, and engineering the features needed to make the ML model hum.

The data scientist can then use the notebook for what it is good for - creating the ML model - and using the ML platform for testing it and running in production. Once done with the ML modeling, one could also use the data exploration capabilities of the data preparation platform to validate the ML model across a number of data sets, to ensure its' accuracy.

All three aspects – preparation, exploration and feature engineering – are essential to the ML workflow. And to perform each of these tasks requires a key set of features we will discuss shortly.

Key Features for the ML Workflow

Datameer is an advanced data preparation and exploration platform that works especially well with big data. The features are designed to help you understand your data better, shape it in the right manner, and to explore it on a self-service basis. As it happens, many of these features make Datameer a great tool to prepare data for machine learning workloads. We review some of those features here. In the next section we'll apply them to a particular dataset that we'll then use to build a machine learning model.

Data profiling

Datameer's Inspector provides data profile information in a visual format, requiring little effort to see the shape of your data. Users can see vital profiling statistics (like number of rows, number of distinct values, and the maximum, minimum and mean) on any column, just by clicking it.

Users can also see data profiling information for the entire dataset by going into Flip Sheet view. The Flip Sheet view provides a deep range of information for each column to provide a richer profile of the data.

Data profiling, especially the histograms that visualize the distribution of values in a column, is important functionality in the data science preparatory workflow. Knowing that a given column has a small number of distinct values may provide a clue that its value can be predicted by a machine learning model. If the column has a large number of distinct values, or a small number if they're evenly distributed, then it may be important to predicting the value of another column.

Visual Data Exploration

Visual Explorer is Datameer's patent-pending technology for performing visual exploration on large volumes of data with amazingly fast interactive performance. This allows customers to see patterns in the data and to do so in an iterative fashion.

Visual data exploration allows a data scientist to dig deeper into their data and become intimate with each attribute and value. Easy to understand charts show interesting aspects of the data and unveil patterns within the values. Unconstrainted exploration and drill down allows a data scientist to work through any paths in the data and quickly switch between different attributes and metrics for truly agile discovery.

The ability to do this on large volumes of data is very helpful in the creation of accurate data models for which mere sampling of data must be avoided. This quickly encourages users to investigate their data tenaciously, since they can ask question upon question with little if any waiting time. Once a user has explored a specific path in the data and

found something valuable to their model, they can then create a refined dataset with the click of a single button.

Algorithmic Exploration

Another way to explore and shape your data is to apply algorithms and statistical functions that reveal patterns. Using this approach, a data scientist can unveil patterns that they would never had seen with standard aggregations and can use these patterns for feature engineering tasks.

Datameer's Smart Analytics offers a focused set of built-in algorithms that reveal advanced patterns in the data. For example, one can view the relationship between columns and the impact it has on others with column dependencies. Users can also create decision tree to see how data columns or values relate to an outcome.

Datameer also offers a rich suite of statistical and advanced grouping functions that can help users explore the shape and values of their data. All of this is extremely important to aid the feature engineering process.

Easy to Follow Workflow

Beyond these exploration features, working with data in a spreadsheet and formula environment is highly congruent with the machine learning preparatory workflow. In Datameer, data is loaded into one sheet then gradually transformed and summarized in successive sheets in the workbook.

This leaves the lineage of the data fully visible and discoverable. In effect, each tab of the workbook is a chapter in a story of the data's evolution, essentially a presentation of the progressions in the analyst's thought process in working with the data.

In some ways, the cells of a Datameer workbook are like the cells in a Data Scientist's notebook. Each one allows a set of transformations on, and/or analyses of, the data. The difference is that notebooks have code, while a Datameer workbooks has data rows and columns with easily readable spreadsheet formulas as well as simple filters, sorts, joins and unions.

Rich Suite of Functions

Another major part of feature engineering is creating new columns from the source data to feed the right columns and values to the AI/ML model. Doing so requires a comprehensive and large suite of easy to apply functions to that transform and aggregate data as well as calculating new values based on statistical or calculation functions.

Datameer offers over 270 powerful, yet simple to use functions to transform and massage the data into the right shape. These elements provide much of the same expressiveness and power of code, but on declarative functions instead of imperative stepwise lines of code. They're accessible to more authors and more readable for a broader set of consumers.

These powerful spreadsheet-style formulas will come in handy when performing feature engineering. ML models typically require numerical, statistical and encoded values. We can apply transformation, calculation and statistical functions to the core data to produce new columns that provide highly tuned values to the ML model in the right format.

Doing it in Datameer and SageMaker

Notebooks are a great place to build and test models. We believe the best results can be achieved by using data preparation and machine learning platforms together. In this section we'd like to show you how this can be done.

Our Example

To flesh this out with an example, we will use a data set of telematics data than can be applied to areas such as usage based insurance (UBI) and driver behavior analysis, risk assessment and pricing of auto insurance. In our specific example, we will prepare, explore and feature engineer an optimal dataset, then use a random forest model to identify trips that are not driven by the insured driver to determine cases where this occurs and identify where this might introduce increase risk.

The original dataset only contains geolocation information that is hard to use directly in predictive models. The high level workflow will be:

- Use Datameer to shape, cleanse and explore the data, then extract features from the data that are most optimal for our model.
- Use a notebook in SageMaker to apply a random forest model, then train and test the model using the SageMaker platform.
- Take test results back into Datameer and explore the results across various attributes to see where it is accurate and improvements can be made.

Our Architecture

In our example, we use an S3 bucket for storage, an EC2 instance to run Datameer, an EMR cluster for data transformation, and a SageMaker cluster to train and host the ML model. Our data files are contained in the S3 bucket, which will also be used to exchange files between Datameer and SageMaker.

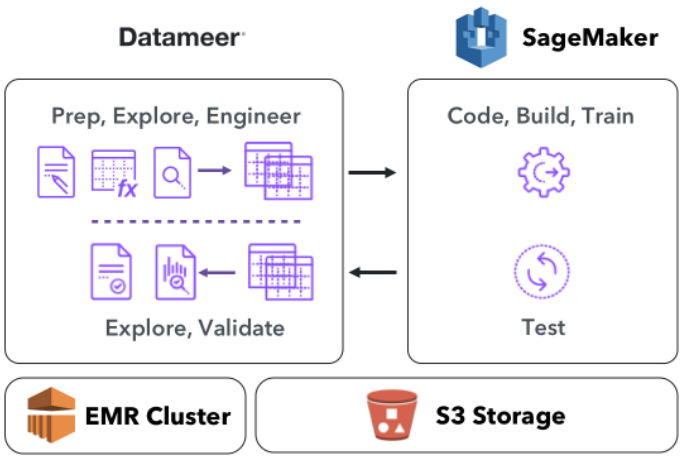

Figure 2: Data Preparation and Science Architecture

Preparatory Workflow in Datameer

First let's start with data acquisition or ingest. Unlike the notebook case, where explicit code must be written to connect to a database or file, Datameer provides a file browser metaphor, similar in concept to Windows' Files Explorer or Mac's Finder. Data stewards, database administrators or personnel in a central analytics group can set configure the population of the folder structure in the browser.

Connections can be passive links to remote databases, import jobs that bring data into the Datameer workbook, or file uploads that contain data from files contributed by users from their own storage media. Data can live in local storage, on-prem server or in the cloud.

A specialist can set up a connection to an S3 bucket. A data scientist can then import data from a file in that S3 bucket, simply by selecting the connection, and now the data comes right in. We can then create a new Workbook based on the data we imported.

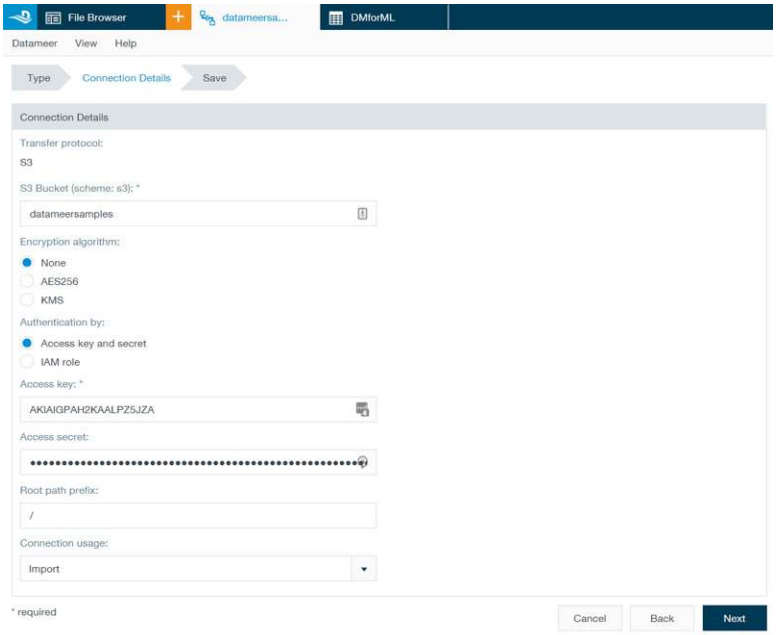

Figure 3: Connection to S3 in Datameer

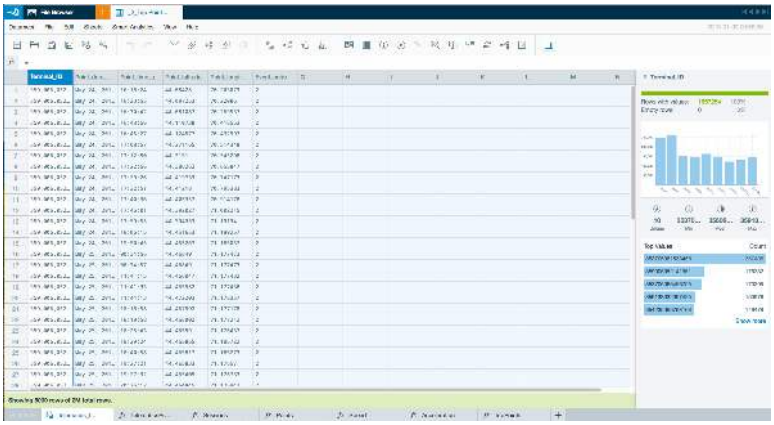

Figure 4: Working with a Datameer Workbook

Once we bring the data in, we notice it is very raw with simple event data and locations. We will need to shape and engineer the dataset so that we can provide a much wider suite of features to the model. With coding, this would take a good deal of work. Luckily Datameer makes it easy. The features we are trying to engineer are:

Time	Acceleration Average	Length
Speed Average	Acceleration Std. Deviation	Distance Per Second
Speed Std. Deviation	Acceleration Minimum	Distance Std. Deviation
Speed Minimum	Acceleration Maximum	Distance Minimum
Speed Maximum	Acceleration 10th Percentile	Distance Maximum
Speed 10th Percentile	Acceleration 30th Percentile	Distance 10th Percentile
Speed 30th Percentile	Acceleration 70th Percentile	Distance 30th Percentile
Speed 70th Percentile	Acceleration 90th Percentile	Distance 70th Percentile
Speed 90th Percentile	No. of Fast Accelerations	Distance 90th Percentile
	No. of Slow Accelerations	

Our first task is to sessionize the data so we can group it into specific trips. Once we've done that, we can then use Datameer inspectors to see some inconsistencies in the data.

As we continue examining the data, with Datameer Inspector, which provides us a profile of the data, it tells us there are some rows in which the Latitude, Longitude, Previous_ Latitude and Previous_Longitude columns have missing (blank) values. So, with a series of simple mouse-clicks on those values, we can create a filter to remove those dirty rows from our dataset.

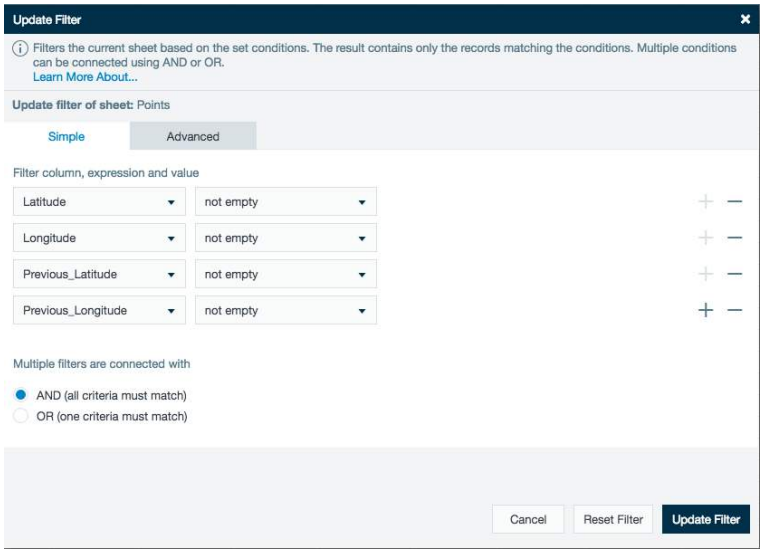

Figure 5: Filtering Data in Datameer

We can then create a series of calculated columns using some of the over 270 built-in spreadsheet-style functions inside of Datameer to calculate Distance (in miles), Time Duration (in Hours), Speed, and Acceleration. For example, to calculate the distance in Miles, we can use HAVERSINE function.

Figure 6: Calculated Columns in Datameer

Our next step is to engineer the potential features we want to feed into the ML model. Once again, we can use a number of Datameer's built-in point-and-click functions to generate new calculated columns that represent these features. Many of these features look at where the driver session landed in various percentiles for different attributes (Speed, Acceleration, Distance, etc.).

Figure 7: Feature Engineering in Datameer

Once we've created these value added columns, we can then explore the data with Visual Explorer to see if there are other patterns in the data that would be meaningful to our problem.

We can also use Smart Analytics algorithm functions to see patterns in the data that lead to certain outcomes. For example, we applied a decision tree algorithm to see certain values that led to the outcomes of OwnerDriven and OtherDriven.

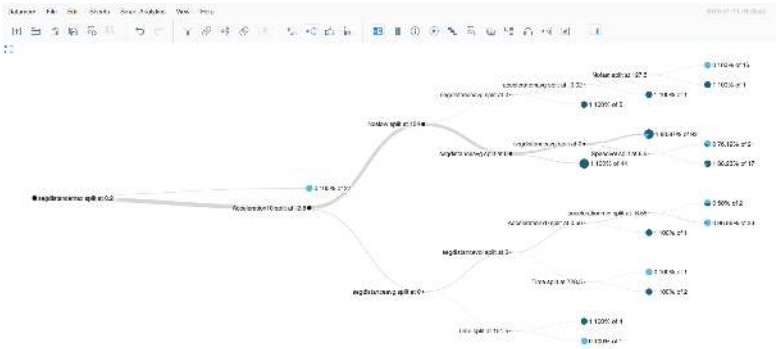

Figure 8: Decision Tree Graph in Datameer

We can also apply the Column Dependencies algorithm to see how different columns are related to one another and find ones that have greater affinity to our outcome.

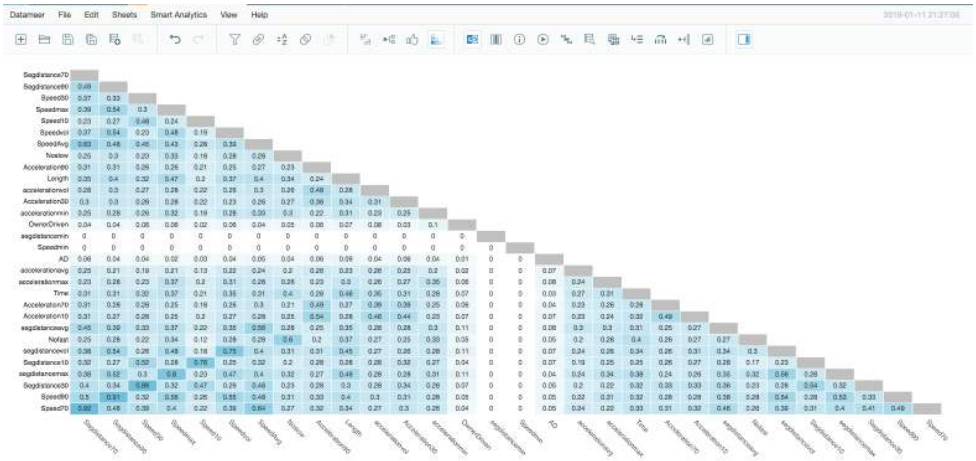

Figure 9: Column Dependencies Graph in Datameer

Finally, we can split the prepared dataset into files for training, test and validation data, and then create an export job to feed our shared S3 bucket where SageMaker can access it.

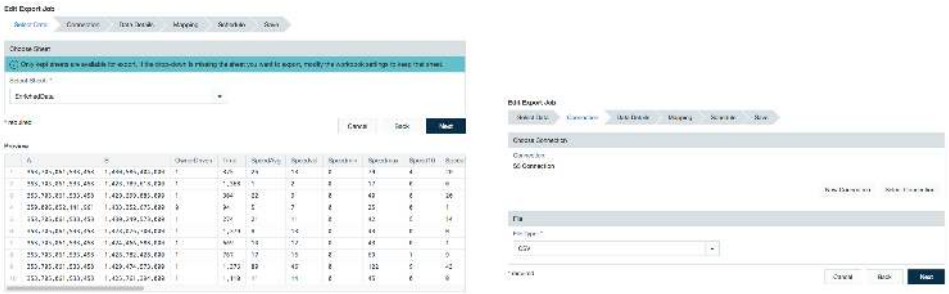

Figure 10: Datameer Export Job

Model Definition, Training and Testing in SageMaker

Now that we have the engineered dataset from Datameer, we can focus our effort to build, train and test the ML model in SageMaker. To do this, we've written some very basic Python code in a Jupyter notebook that uses a SageMaker built-in algorithm.

We'll now show you highlights of what we did there, including defining, training and testing the model, then pushing the "scored" data (the test data along with the predictions it ran on it) back into S3 bucket for us to further explore.

We can use the AWS console to create a new SageMaker Jupyter notebook instance.

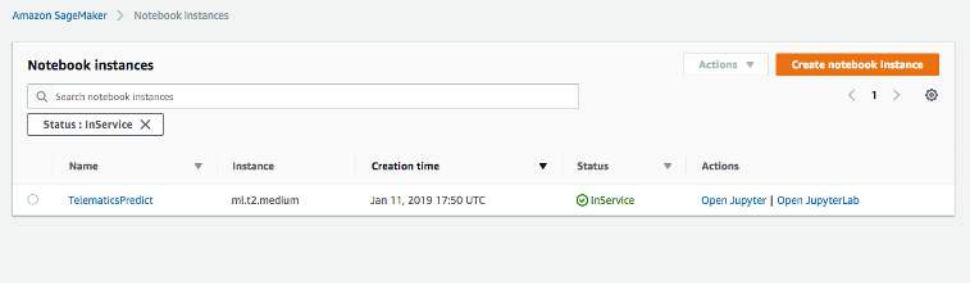

Figure 11: Notebook Instances in SageMaker

We can then use the notebook environment inside of SageMaker to write our code.

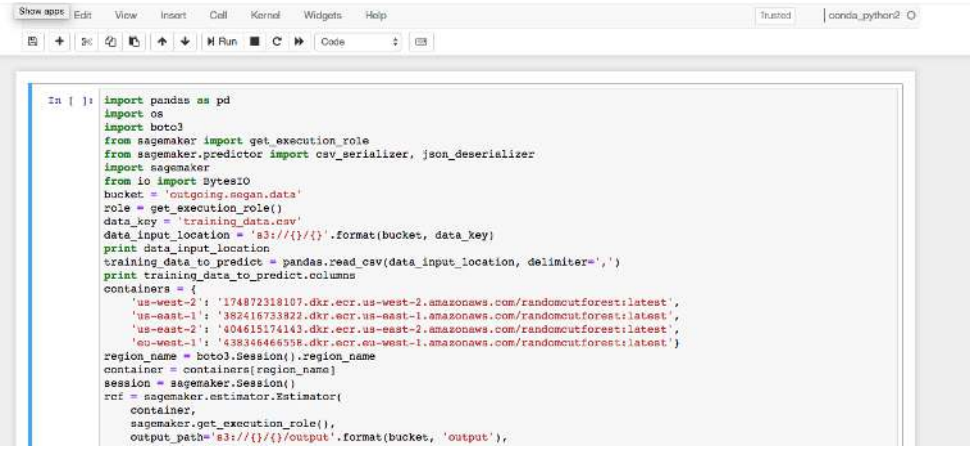

Figure 12: Jupyter Notebook in SageMaker

Our first task is to set up our environment and read the data exported by Datameer from our shared S3 bucket:

```
import pandas as pd
import os
import boto3
from sagemaker import get_execution_role
from sagemaker.predictor import csv_serializer, json_deserializer
import sagemaker
from io import BytesIO

bucket = ‚outgoing.segan.data‘
role = get_execution_role()
data_key = ‚training_data.csv‘
data_input_location = ‚s3://{}/{}‘.format(bucket, data_key)
```

```
print data_input_location

training_data_to_predict = pandas.read_csv(data_input_location, delimiter=‘,‘)
print training_data_to_predict.columns
```

Now that we have the data, we are ready to define and train the model! We are using a SageMaker built-in algorithm, in our case Random Cut Forest. We could also use deep learning frameworks like TensorFlow, MXNet, PyTorch etc. or other ML frameworks like scikit-learn too. Using a SageMaker built-in algorithm means the amount of code required is minimal. There are other benefits too, these algorithms are cloud optimized for improved performance and they also enable distributed training for large datasets (you can read more about them here).

SageMaker makes this very easy by giving us a direct API to use the Random Cut Forest algorithm, define our parameters – in our case the number of samples per tree at 200, the number of trees at 50 and the feature dimensions at 29. The modeling and training process requires a very straight-forward, compact set of code where we can set up my containers, sessions and parameters, then train our model:

```
containers = {
    ,us-west-2‘: ,174872318107.dkr.ecr.us-west-2.amazonaws.com/
randomcutforest:latest‘,
    ,us-east-1‘: ,382416733822.dkr.ecr.us-east-1.amazonaws.com/
randomcutforest:latest‘,
    ,us-east-2‘: ,404615174143.dkr.ecr.us-east-2.amazonaws.com/
randomcutforest:latest‘,
    ,eu-west-1‘: ,438346466558.dkr.ecr.eu-west-1.amazonaws.com/
randomcutforest:latest‘}

region_name = boto3.Session().region_name
container = containers[region_name]

session = sagemaker.Session()

rcf = sagemaker.estimator.Estimator(
    container,
    sagemaker.get_execution_role(),
    output_path=‘s3://{}/{}/output‘.format(bucket, ,output‘),
    train_instance_count=1,
    train_instance_type=‘ml.c5.xlarge‘,
    sagemaker_session=session)
```

```
rcf.set_hyperparameters(
    num_samples_per_tree=200,
    num_trees=50,
    feature_dim=29)

s3_train_input = sagemaker.s3_input(
    s3_data=data_input_location,
    distribution='ShardedByS3Key',
    content_type='text/csv;label_size=0')

rcf.fit({,train': s3_train_input})
```

Once we've trained the model, we can compute scores for each of the training data points. First, we create an inference endpoint using the model we generated, then we run the model across the entire dataset and use standard deviations to determine anomalies in the data:

```
from sagemaker.predictor import csv_serializer, json_deserializer

rcf_inference = rcf.deploy(
    initial_instance_count=1,
    instance_type='ml.c5.xlarge',
)

rcf_inference.content_type = ,text/csv'
rcf_inference.serializer = csv_serializer
rcf_inference.deserializer = json_deserializer


results = rcf_inference.predict(training_data_to_predict.as_matrix())
scores = [datum[,score'] for datum in results[,scores']]
training_data_to_predict[,score'] = pandas.Series(scores, index=training_data_to_
predict.index)

score_mean = training_data_to_predict.score.mean()
score_std = training_data_to_predict.score.std()

score_cutoff = score_mean + 3*score_std
anomalies = training_data_to_predict[training_data_to_predict[,score'] > score_
cutoff]
```

In our final step, we write the test results to a CSV file suitable for us to explore and drill into model accuracy inside of Datameer Visual Explorer. We will save the file in our shared S3 bucket:

```
print anomalies
#results_csv = anomalies.to_csv(sep=‘,‘)

csv_buffer = BytesIO()
anomalies.to_csv(csv_buffer, index=False)

s3client = boto3.client(„s3“)
response = s3client.put_object(
 Body=csv_buffer.getvalue(),
 Bucket=bucket,
 Key=‘model_results.csv‘
)
```

Exploring Test Results in Datameer for Accuracy

With our model now trained and tested, we can explore how our model behaved across the various attributes inside of Datameer using Visual Explorer. To get the data back in, we could create a new Import job that connected to the CSV file directly. We could also download the file manually and use File Upload job in Datameer to push it back into the workbook. The end-result of that process is shown below:

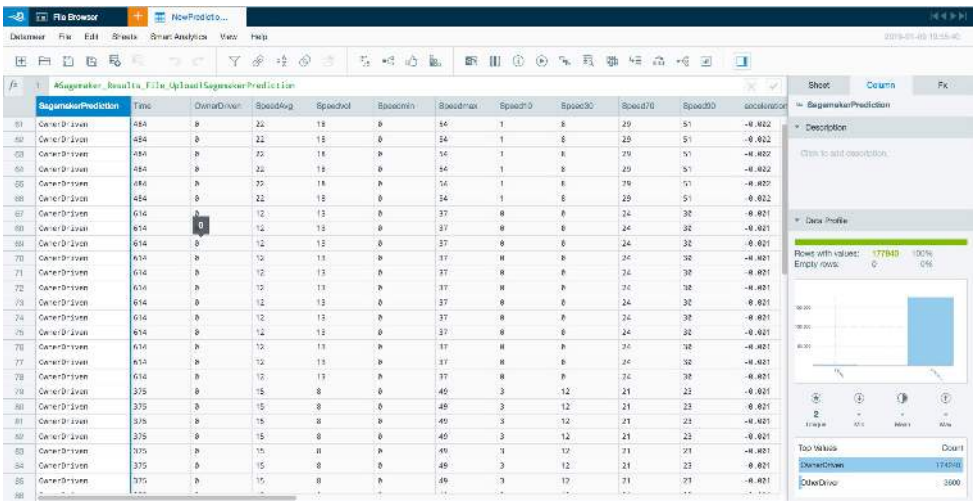

Figure 13: Worksheet with Test Results Data in Datameer

In our model we are trying to predict if the ride was owner driven or driven by another person. We have two key columns here we want to explore – SageMakerPrediction which tells us what SageMaker predicted, and PredictionCorrect which is a Boolean value which compares the SageMaker prediction to the actual driver state. As we can see from our analysis of the prediction data back in Datameer, the model we built is about 87% accurate. Not bad!

PredictionCorrect	Count
False	23,040
True	154,800

Figure 14: Prediction Accuracy Calculated in Datameer

We can now use Visual Explorer to drill down and examine the accuracy of the model across any attributes to see how well it behaves on those dimensions. The most obvious place to look is how accurate the model is based on the actual driver status.

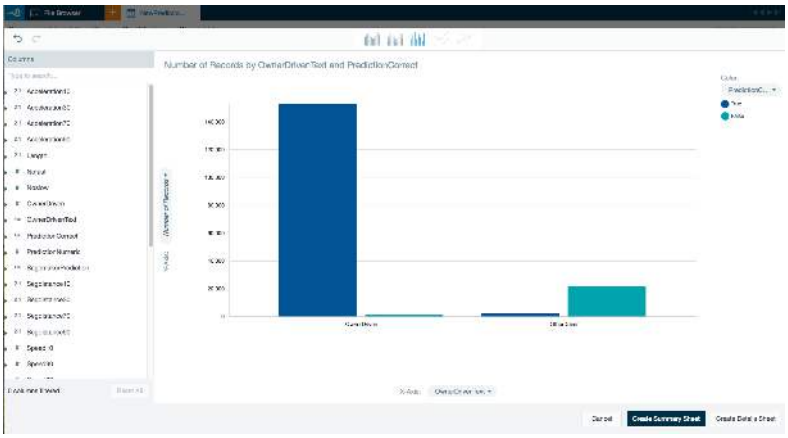

Figure 15: Exploring Test Results Data in Datameer Visual Explorer

As we do this, we can see that our model improves accuracy in cases where the car was owner- driven. However, in cases where there was another driver, the model is not predicting accurately. In fact, the accuracy of the model is only 9% when there was another driver. This definitely tells us we need to explore additional features or add additional data to try improve the accuracy along this dimension.

This insight proves that using a tool like Datameer in tandem with a machine learning platform like SageMaker is very effective. After training and testing the model in SageMaker, we were able to send the scored data to Datameer and analyze patterns in the model's accuracy.

Summing Up

Data preparation and data science tools are highly complementary, with each offering critical functionality to form a more optimal workflow to speed time to AI and ML insights. In particular, a data preparation platform that also offers scalable data exploration in a variety of ways – data profiling, algorithmic and visual exploration – can truly work in tandem with the data science platform to also help feed better and more data to the ML models to increase their accuracy and value to the business.

The combination of Datameer and SageMaker can deliver a more optimized overall workflow from shaping the data, performing feature engineering, training and deploying the ML model, and examining its' overall behavior. The combination takes many of the code-intensive aspects of the process away to not only speed the total workflow, but also gain re-use across projects and collaboration among the data science teams. Datameer's integration with the rest of the AWS platform, including EMR, provides the scalability, security and governance for extensive data science programs.

About Datameer

Datameer is an analytics lifecycle platform that helps enterprises unlock all their raw data. The cloud-native platform was built for the complexity of large enterprises—yet it's so easy to use that everyone from business analysts to data scientists to data architects can collaborate on a centralized view of all their data. Without any code, teams can rapidly integrate, transform, discover, and operationalize datasets to their projects. Datameer breaks down data siloes, gets companies ahead of their data demands, and empowers everyone to discover insights. Datameer works with customers from every industry including Dell, Vodaphone, Citibank, UPS, and more. Learn more at datameer.com.