**Cloud Machine Learning as a Service (MLaaS):**

*“Machine Learning as a Service (MLaaS) is a umbrella term used by Cloud-based platforms that provide services like data preprocessing, model training, model evaluation and predictions packaged as one service“*

There are many companies who are offering extensive services in this field of Machine Learning targeting different users and businesses (Enterprise, Midlevel, and Startups). As per the market analysis Amazon, Microsoft, and Google are the three leading cloud MLaaS service providers that allow either automated or semi-automated plus flexible service for fast model training and deployment.

**Platform Assessment**

Recently in April 2018, Google has deprecated their prediction API which caused many users to switch their models to other platforms or rewrite their models. Considering potential risks with Google, I have inclined my research towards Amazon and Microsoft only.

Both Amazon and Microsoft Azure provide services for newcomers and an expert data scientist. In case of deadline-sensitive projects where we have unprocessed data to make some predictions, automated services come handy with the price tag.

Below Figure 1 from Alexsoft website depicted a basic comparison between Amazon, Microsoft, and Google

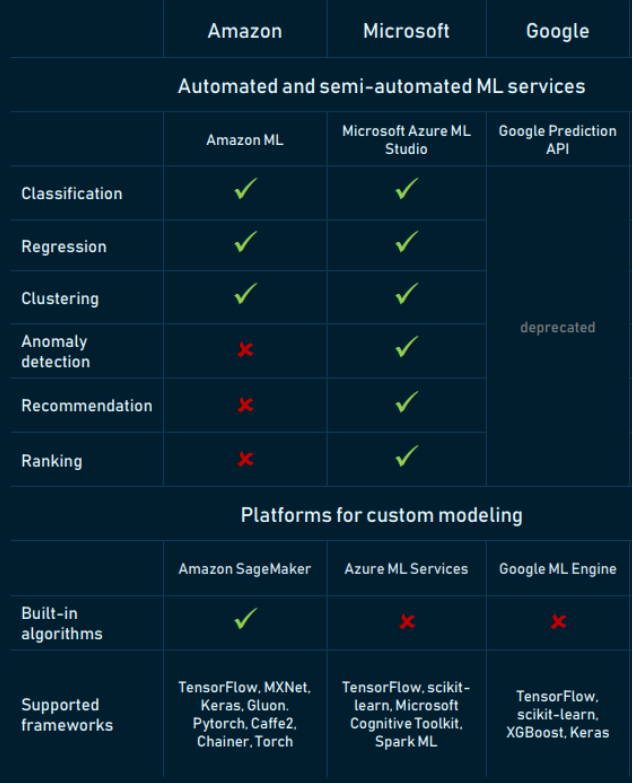


Figure 1: Alexsoft Cloud Machine Learning Service Comparison

From the above figure, we can see Microsoft ML Studio (automated system) provides more algorithm then Amazon where Amazon is very limited and restricted when comes to automated AWS Machine Learning service.

I have developed **POC** on Amazon and Microsoft, testing their both automated and semi-automated services.

**Automated Service**

I have used my own dataset of 40 rows to test the services. It took me around 5-10 mins on both platforms to upload the data , model training and make predictions. Amazon and Microsoft have provided almost the same predictions.

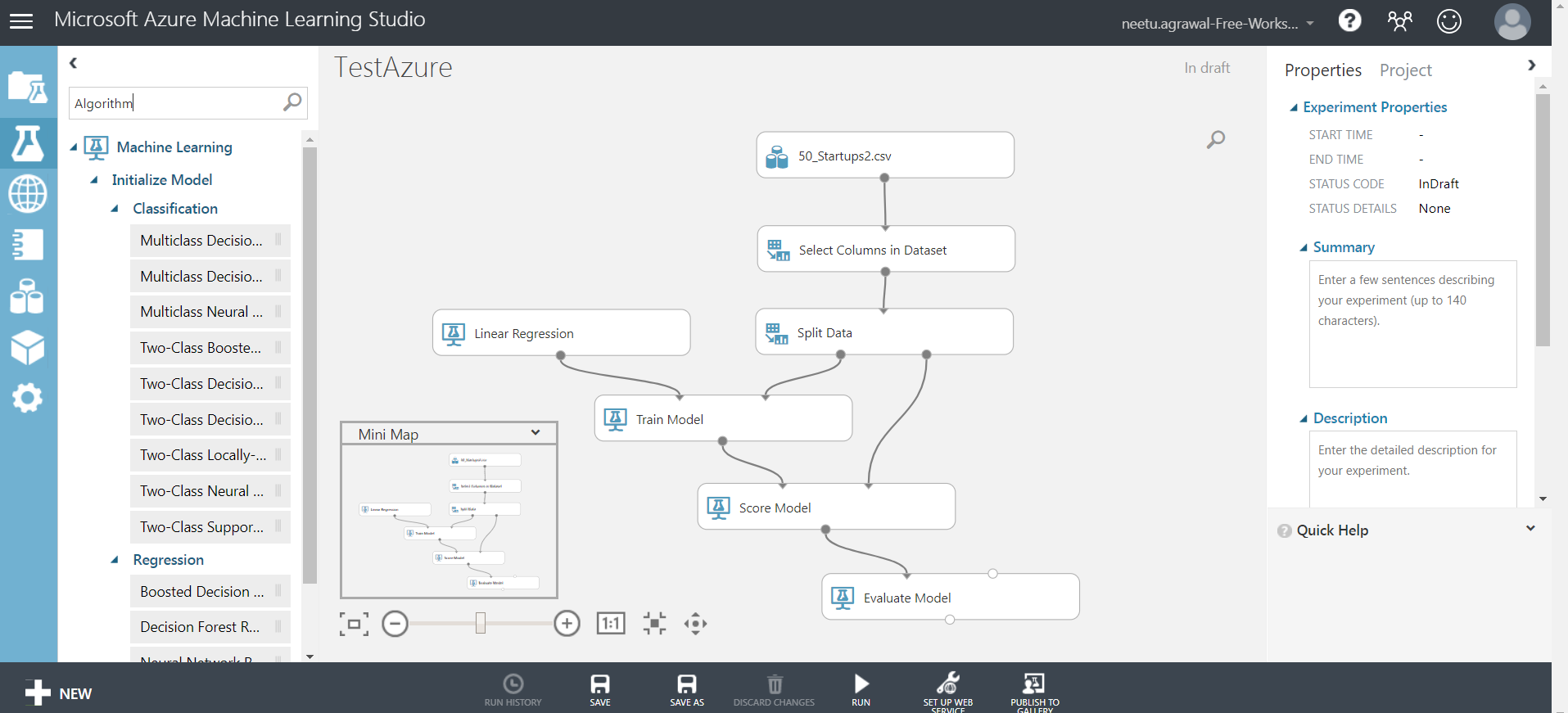
**Amazon** has automatically preprocessed my data and categorises the fields (Categorical, Numerical, Binary or Text) and in next step, it showed me chose Algorithm which was Regression and was asked to either go with the default settings or provide custom settings like Learning rate, number of iterations. To achieve this I have created two service :

* Amazon Machine Learning and
* AWS S3, (Amazon automatically created EC2 instance to deploy my model).

It cost me 0.86 in total excluding S3 charges. Amazon Machine Learning services doesn’t come under Free Tier

**Microsoft** allows me to upload the data directly in the Workspace from my local computer which skips the steps of uploading in Data Storage and all. Through its quite intuitive drag and drop interface, I was able to preprocess my data and apply algorithm from the list to make predictions *Although* here I chose the algorithm to train my model, and for which you need to have some knowledge of data science. Interestingly graphical visualisation of each step of your workflow makes it easier to take your next step and gives your ability to trace your steps back in case you missed something instead of fiddling around. It cost me nothing as it got covered up in Free tier

**My Experiment in ML Studio**



**Comparison of Automated Services :**

|  |  |
| --- | --- |
| AWS Machine Learning | Azure ML Studio |
| Supports only supervised learning that includes Binary , Multi-class classification and regression | Supports 100 methods that address classification (binary+multiclass), anomaly detection, regression, recommendation, and text analysis. It’s worth mentioning that the platform has one clustering algorithm (K-means). |
| Automated Data pre-processing  The service identifies which fields are categorical and which are numerical, and it doesn’t ask a user to choose the methods of further data preprocessing (dimensionality reduction and whitening). | Provides direct drag and drop options to pre-process data. |
| Amazon choose the algorithm itself after processing the data | Drag and drop the algorithms from the menu on data box to train the model |
| Supports only two data sources: S3 and AWS Redshift otherwise  To get data from other AWS sources, use data pipelines to convert or transfer data in S3 | Allow to import data directly from Local computer otherwise  Import data from the following options in a module and then add that module in your experiment  -WebURL via HTTP  -Azure Blob storage  -Azure Cosmo DB (NoSQL)  -Azure SQL DB  -On-Premise SQL Server DB |
|  | **Allow to add your own scripts in Python and R,** but it has some limitations. Please check the note below to get more information |
|  | The Azure ML graphical drag and drop interface visualizes each step within the workflow. This includes data exploration, preprocessing, choosing methods, and validating modeling results. |

Note: Ml Studio does allow to incorporate your own python scripts but it comes with some limitations like The Python runtime is currently sandboxed and, as a result, does not allow access to the network or to the local file system in a persistent manner. All files saved locally are isolated and deleted once the module finishes. You can find them more in details here : <https://docs.microsoft.com/en-us/azure/machine-learning/studio/execute-python-scripts#limitations>

**Semi-Automated Service**

I have used my own dataset of 40 rows to test the service in Amazon but I have used the sample dataset and sample project to test the Microsoft service.

**Amazon** provides semi-automated service called **Sagemkaer**. Since Sagemakerprovides built-in algorithms, I have used Linear Learner algorithm for Regression in Sagemaker. It took me around 12-15 hours to achieve data preprocessing, model training , model deployment and final predictions. The services I have used to achieve this are: Amazon Sagemaker - Jupyter Notebook instance, ECR(Elastic Container Repository) , S3, DynamoDB, Data Pipeline. It cost me 0.18 as most of the tasks got covered up in Free Tier. I have used following steps :

1. Created Notebook instance ml.t2.medium (because it is the cheapest and comes under Free Tier)
2. Created S3 bucket and uploaded my CSV file data
3. Code my program in Notebook using Python Conda 3.
4. Code : Labeled my data as per LinearLearner algorithm requirement and fetch the data from S3 and push it in the algorithm
5. Created repository in Elastic Container to train and deploy my model
6. Code - train and deploy model and final make predictions

Most of the time I spent on processing my dataset (LinearLearner raised issue in having Categorical feature within my dataset, eventually I have removed the column ) and later training my model. Amazon doesn’t provide good logs to test your code so everytime I had to go to the training jobs and check the failed jobs to trace their logs. Please find my python script here [LinearLearner.py](https://drive.google.com/open?id=1s4TzJDPRyam38vsYrcj3SvAlmS1hGF_y)

**Microsoft** ML Studio provides flexibility to upload your R or pythons scripts, it is feasible for small modules but it is cumbersome for large modules. So Microsoft does provide a semi-automated service called **Microsoft Azure Machine Learning Services(Azure MLS)**. I have used sample dataset and sample project to test this service, a reason behind doing so as MLS doesn’t provide any built-in algorithm and it requires own algorithm to test this service.

To run a sample project it took me few hours to do all setup and installation. Semi-automated services requires installation of Workbench on your local computer and create and run your project locally. I have followed below steps :

1. Create service accounts for Azure Machine Learning services
2. Create resource group and then resource called Experimentation
3. Install and log in to Azure Machine Learning Workbench.
4. Select the sample project in Workbench
5. Run a script in that project
6. Access the command-line interface (CLI)

I have followed the steps till 5, after that the next moves were to deploy the model which includes create real-time web services. It cost me nothing as Workbech is free, it costs only when you deploy the model and start using web -services.

**On-Premise**

**Amazon** now provides a Local Mode of Sagemaker to train our notebook instance where you can do write your own scripts using Sagemaker Python SDK and test your work locally on small dataset and then scaling to train on full dataset on Cloud. To do this, it uses Docker compose and NVIDIA Docker.

We can also use Sagemaker Notebook instance as a local environment, all we need to run this script “pip install -U sagemaker” and it will install necessary prerequisites.

Note : Even though we’re training locally, we’ll still access data from Amazon S3 to maintain consistency with SageMaker training.

**Microsoft** semi-automated service only supports local environment for now. To access their service we need to install Workbench locally and create our scripts and train our model locally before creating real-time web service and deploying it on a cloud.

Note: In terms of platforms for custom modeling, both providers provides more or less similar services.

**Comparison of Semi-Automated Services :**

|  |  |
| --- | --- |
| AWS | Microsoft Azure |
| Built-in algorithm | No |
| Provides cloud and on-premise environment | Only On-premise environment |
| Best scenario for those companies who are already using Amazon and not planning to switch to other cloud platforms | Best Scenario for Enterprise businesses |
| AWS provides Free digital training and certification to help in learning AWS foundation | Microsoft provides Free digital training |
|  | Provides very organised Architecture to manage your resources under one group and subscription |

**Pricing Model**

|  |  |
| --- | --- |
| AWS | Azure |
| Free Tier | |
| <https://aws.amazon.com/free/faqs/?ft=nf>  **Sagemaker First 2 months**   * 250 hours of t2.medium Notebook usage each month for first two months * 50 hours of m4.xlarge for training * 125 hours of m4.xlarge for deploying ML models   **12 months Free Tier with Amazon Sign Up**   * 5 GB S3 storage (but they do charge for Usage Values) * 750 hours RDS per month * 750 hours EC2 per month * 1 millions API thru API Gatway per month * 1 GB Quicksight   **Always free**   * 25 GB of Dynamo (NoSQL) * 1 Million AWS Glue ETL service * 1 million per month AWS Lambda | **ML Service Free tier**   * First 2 seats * 10 GB Storage * 100 Modules per experiment * 1 hour duration for each experiment * Execution on one node * No Production Web API   **12 months Free Tier**   * 5 GB Blob Storage * 5 GB File Storage * 250 Gb SQl DB * 15 GB Data transfer bandwidth * 64 GB x 2 SSD |
| Standard Tier | |
|  |  |
| Amazon SageMaker Batch Transform only charges you for the instances used during while your jobs are executing. If your data is already in Amazon S3, then there is no cost for reading input data from S3 and writing output data to S3 |  |

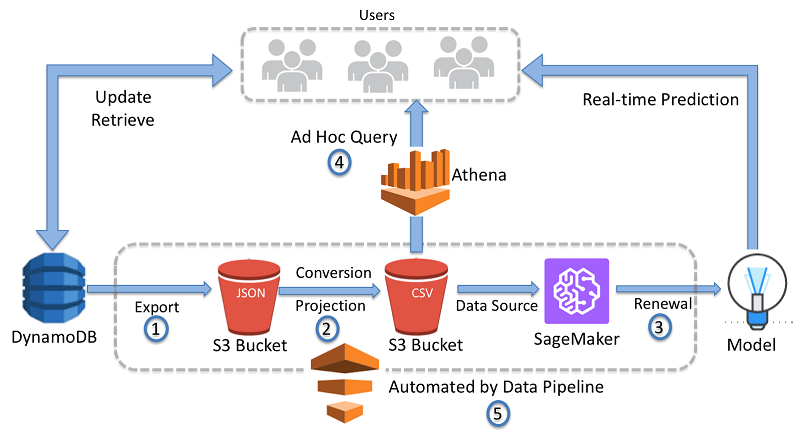
**DataStorage:**

If we choose the same provider both fro Storage and Machine Learning , this will reduce the time spent on configuring a data source.

*Note: Amazon SageMaker uses Amazon S3 as the primary storage for training data*

AWS Dynamo DB :

* NoSQL DB
* Provide SDK for man languages to do CRUD operations and run simple queries
* AWS provide Downloadable version of DynamoDB or DynamoDB webservices
* Amazon offers free digital training for DynamoDB
* Auto Scaling Feature or Set Read And Write capacity Feature
* https://boto3.readthedocs.io/en/latest/guide/dynamodb.html



**Database Structure:**

NoSQL database is best structure to store Big data as big data is unstructured. NoSQL Database over SQL database because it is quickly scalable and very quick in scanning table with millions and millions of entries as they do indexed search.

**External Application :**

<https://aws.amazon.com/blogs/apn/integrating-with-amazon-sagemaker-using-built-in-algorithms-from-external-applications/>

**Connect to Core Devices:**

Amazon Greengrass

If we are planning to connect our IOT gateway running on a site to AWS , we need AWS IOT services for it.

**Personal Notes :**

Amazon does provide a proper documentation with Cleanup details so if you are testing their services you wouldn’t get charged once you are done. This is very helpful. Like I have tested their ML and Sagemaker services and after I was done with my evaluation I have deleted all the entities (as mentioned in the document) which might cause AWS to charge us in future.

**Reference Documents:**

* <https://www.altexsoft.com/blog/datascience/comparing-machine-learning-as-a-service-amazon-microsoft-azure-google-cloud-ai-ibm-watson/>
* <http://blog.exposedata.com.au/2016/09/19/the-battle-of-the-amls-amazon-machine-learning-vs-azure-machine-learning/>
* <https://blog.pythian.com/comparative-analysis-amazon-sagemaker-google-datalab/>
* <https://www.youtube.com/watch?v=MxULtApNYCk>

**Credentials :**

AWS IAM Role:

Email : [neetu.agrawal@qbots.ai](mailto:neetu.agrawal@qbots.ai)

Password : agw@Qbots

AWS Developer Role:

Account ID : 771669846174

Username: Developer

Password : Qbots2018

Instance name : **Testing**

Note : you need to start the instance first and then you will have the access of all the files I have created within it called LinearLearner.py. Data files are in S3 bucket under qbots-bucket/Example-datasets/