# Behavior Analysis using Serverless Machine Learning

Darezik Damkevala
Department of Computer Engineering
Vidyalankar Institute of Technology
Mumbai, India
darezik@darezik.com

Rohit Lunavara
Department of Computer Engineering
Vidyalankar Institute of Technology
Mumbai, India
rohit.lunavara@gmail.com

Mansi Kosamkar
Department of Computer Engineering
Vidyalankar Institute of Technology
Mumbai, India
mansikosamkar@gmail.com

Suja Jayachandran
Department of Computer Engineering
Vidyalankar Institute of Technology
Mumbai, India
suja.jayachandran@vit.edu.in

*Abstract* — **This paper supplies a route for using the Watson Machine Learning API on IBM Cloud to carry out serverless data analytics using machine learning as a service. Transforming the large amount of data produced by an organization into intelligence can be done using advanced analytics methods such as using a modified Mahalanobis Distance algorithm for synthesis of correlation data under the purview of machine learning. Further refinement of correlation data is done using a Multivariate Reliability Classifier model. The consumption of this advanced analytics service can be done in a serverless manner where the developer only must be concerned with how the data is analyzed, i.e. scoring, batch or stream models with a continuous learning system without the outlay of hardware upon which to train those models. This paper examines the usage of such serverless AI systems in the scope of user behavior analysis over varied demographics.**

***Keywords — Data Analytics, REST API, Serverless Computing, Machine Learning Models, Mahalanobis Distance, User Behavior Regression, Demographic Classification***

## I. INTRODUCTION

Data analytics using Machine Learning gives the computers an ability to learn without any specific programming. Adding powerful tools which need less human effort, model creation becomes a lot easier, predictions improve by continuous model retraining, and forecasts get smarter over time as well. We can find unique market needs and market trends which allows change with agility.

The Machine Learning algorithms that we are going to use for Behavior Analysis need correlation data from multiple variables which is obtained from the covariance matrix, which in turn is obtained from Mahalanobis distance. Here, we must empirically estimate the covariance and mean vectors based on the input vectors. However, this approach breaks down as estimating the covariance may lead to an incorrect output if the covariance matrix is non-invertible. The workable solutions to this problem are discussed later.

The Machine Learning service that we are currently surveying will use this refined correlation data. These services are a set of REST APIs that we can call from any programming language which also helps us integrate Watson Studio analytics in our application. The IBM Cloud applications created by us can be bound to the Machine Learning service instance and as such we can generate predictive analysis that our applications need.

We also discuss the strategies that we used for input data normalization [1]. The focus is on the *Category* and *Task* model. The input classification strategy has also been discussed where the idea of relating a user to a demographic and that the change in a user's data will affect the affinity to a demographic.

## II. USER BEHAVIOR ANALYSIS

User Behavior analysis is an overly broad terminology. In this section, we give a brief description of the problem with a specific user behavior use-case such as obtaining poor results which is directly proportional to the low amount of effort put in by the user, etc. as an example.

The variables to compute effort in our use-case are as follows:

- Task completion rates,
- Work period frequency,
- Task category,
- Time spent,
- Time allotted,
- Length of individual work period, and
- Time of day of each work period

There are many other types of user behavior use-cases in our product -e.g., actions that are supposed to be inversely proportional such as scoring low on a test even though the user put maximum effort shown by the time

978-93-80544-34-2$31.00 ©2019 IEEE


Authorized licensed use limited to: ECOLE POLYTECHNIQUE DE MONTREAL. Downloaded on January 16,2022 at 01:14:31 UTC from IEEE Xplore. Restrictions apply.

spent on preparation, results differing based on the location and time of preparation by the user, etc. that are currently supported by fitting machine learning algorithms under the hood. However, we will turn our attention towards the example for clarity and due to domain constraints.

Consider a student who has obtained unsatisfactory results in an examination since they have failed to put in enough effort towards that field of study. We see the parameters in the effort that they had put forth towards the examination. The variables under scrutiny should be the highly weighted variables in their effort. To predict their behavior and supply insights, we should look to define a baseline which will be used for further comparisons. We define a self-baseline, in which we look to judge the user's behavior against their own past behavior.

But, for this we need to clarify what *user behavior* means. This process entails distinguishing the time-granularity for user behavior analysis (second, minute, hour etc.) and recognizing features which help show the composition of the effort put in by the user for each user-task pair. The features which characterize effort have already been mentioned.

The accumulated information from the user over time is the input to the user behavior prediction algorithm. For the self-baseline, data vectors for each task performed by the user since the beginning is chosen as a baseline data vector X.

## III. Finding Correlation Data

The algorithms that we are using rely on calculating the Mahalanobis distance which is explained at length in [2]. It is a multivariate generalization of the z-score (which shows the number of standard deviations from the mean of all observations per observation). "When there are multiple variables in each observation, the Mahalanobis distance shows how many standard deviations away an observation is from the mean value of all observations" [3]. It is not reliant on any scale or unit of measurement. When we have an observation vector of N variables x = {x1..., xN} and an observation set X = {x1..., xK} with the following mean vector μ = {μ1..., μN} and covariance matrix Σ, the Mahalanobis distance is given by

$$\sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

However, the actual values of the mean vector $\mu$ and the covariance matrix $\Sigma$ are not available, therefore, we must use practical estimation based on the observations. To get around the constraints of computing the inverse covariance matrix using the equation leads to some problems if the covariance matrix $\Sigma$ is singular or ill-conditioned. [3] offers insights on the numerical stability of these calculations.

Reference [4] has suggested the following algorithm for calculating the Mahalanobis distance which uses SVD (Singular Value Decomposition).

Also, [5], shows an MRC (Multivariate Reliability Classifier) model which is based on multiple Mahalanobis distances from various sets of observations. It shows that the classifier can both quantitatively measure reliability of the reliable observations and qualitatively screen deviants from the established norm.

$$MD_{i,l} = a_0 + \sum_{m=1, m\neq l}^{k} a_m MD_{i,m} + \varepsilon_i$$

The "PCA-MD-MRC (Principal Component Analysis - Mahalanobis Distance - Multivariate Reliability Classifier) model equation" [4] is as shown above. Here, k is the suitable number of tests, $\varepsilon$ is the sample error in calculation, $a_0$ is the offset value, the Mahalanobis distance $MD_{i,k}$ is then linearly regress as shown in the above equation.

This model uses error distribution to lower the amount of variance in measurements and qualitatively classify the reliability of the product.

## IV. REST API

The reliable correlation data thus obtained can be used as an input to the REST API to analyze user behavior. This REST API is used to interact with the IBM Watson Machine Learning services. In this, using the correlation data and metadata for a predictive model, we then deploy and manage the deployment from the services. There are various models that can be used depending on a specific use case.

- Online Model
- Batch Model
- Stream Model

This paper outlines data points and normalization methods are used to prepare the training data. IBM Watson Studio prepares the data and recommends suitable techniques, building out a model without any deep knowledge of machine learning needed.

For a task management and behavior analysis system we do not need Classification Methods, but we may use them to tie Users whose demographics are unknown to a specific demographic. For this purpose, we must use multiclass classification as binary classification would prove unsuitable for the purpose.

To analyze and predict an individual users' future behavior we must use a Regressive model as we will have many values in our label column.

Another choice would be to build the model in a manual fashion using one of the following three techniques as outlined in [6]:

1. Decision tree classifier
2. k nearest neighbors' classifier



Authorized licensed use limited to: ECOLE POLYTECHNIQUE DE MONTREAL. Downloaded on January 16,2022 at 01:14:31 UTC from IEEE Xplore. Restrictions apply.

3. Naive Bayes

In this specific situation the Decision tree classifier is the best choice for our dataset since it supports multi-class labels along with continuous and categorical features while linking observations on the item (represented in branches) to conclusions on the target value of the item (represented in the leaves).

### A. Online Deployment of Models

The IBM Watson services can be accessed using API access tokens. These tokens need to be generated using the *UserID* and *Password* in the service credentials tab. We can then work with published models using the published model's URL, deployment URL and usage information. The next step is the creation of an online deployment of a predictive model.[6] After successful deployment, we can finally check the parameters of the deployed model status and scoring details.

### B. Batch Deployment of Models

The IBM Watson services can be accessed using API access tokens. User authentication should be performed using the *UserID* and *Password* in the service credentials tab. The token generated is used to interact or work with the published predictive model by using the model URL, deployment URL and usage information. Then, we create the online deployment of a predictive model. Finally, we obtain the information about the mode and then make score requests against it.

### C. Stream Deployment of Models

Like the online and batch deployment of models, here, we start by generating an API access token to access the IBM Watson services using their API. Then, we work with the published predictive models using its URL, deployment URL and usage information. Then, we create a stream deployment using IBM Event Streams. We obtain the deployment details to figure out if the deployment was successful or not. After successful deployment, we can stop, start, or cut the streaming deployment.

### D. System for continuous learning

Like earlier modes of deployment, we start by generating an API access token by using *UserID* and *Password* in the service credentials tab. Then, we start working with the published predictive model by setting up the continuous learning system for it. This is done by assembling the authorization header and the feedback data set. The continuous learning system's iterations are run, and the feedback data is stored.

## V. Input Data Normalization

Getting Things Done with Serverless Home Automation is our homegrown task management and data analytics application for which serverless machine learning would be an ideal use-case.

The data is exposed through controllers since the administrator user interface uses an MVC (Model- View-Controller) pattern and so it can be easily piped through to another API.

The goal is to collect adequate and correct data points which directly affect the results while also keeping some semblance of user privacy.

The data is normalized to values ranging between 0 and 1 by using continuous sliders rather than a discrete integer-based system. Since the aim is to increase the user's productivity level through their satisfaction [7] levels, this supplies a tactile and easily usable interface for all skill sets and age groups. This normalization could alternatively be carried out using the database normalization design pattern outlined in [8].

For this paper, the relevant models with their dependencies are:

### A. Category

Tasks have a category attached to them which includes the fields *Priority* which is a non-normalized user facing field better suited for scheduling than machine learning, however, they also have a normalized weight.

### B. Task

Tasks in addition to the category attached to them also have a *ResultID* and an *Estimate in Seconds*. A single task can also have multiple *periods* attached to it. This allows us to supply insights not only on the time taken to complete the task but also the way it was completed (i.e. short bursts versus long sprints).

Then, these variables are fed together to the IBM Watson API in a stream deployment model which forms the basis for our machine learning model.

### C. Result

Using methods shown in [9], where the authors outline an objective measurement system to measure satisfaction values in the context of tourism [10] or in [11] where service satisfaction values are obtained using fuzzy measures, we find the normalized:

- Perceived satisfaction with the result,
- Score (marks obtained divided by total marks),
- Perceived scope for improvement, and
- Result's weight (how important the specific result is to the user in question)

### D. Demographic

Most internet enabled applications have users from all over the world [12] and as a result they come from a multitude of demographics; our sample set is not much different with different demographics created based on:



Authorized licensed use limited to: ECOLE POLYTECHNIQUE DE MONTREAL. Downloaded on January 16,2022 at 01:14:31 UTC from IEEE Xplore. Restrictions apply.

- Nationality,
- Residency,
- Age lower bound,
- Age upper bound,
- Co-ordinate bounding box:
  - Top left,
  - Top right,
  - Bottom left, and
  - Bottom right

The relationship between demographics and users along with the calculations is defined in the next section.

## VI. Input Classification Strategy

All the entities outlined in the earlier sections are tied to a specific *UserID* and each user's data is a discrete entity from the front-end's perspective.

The user's details and metadata are collected using data-points such as:

- Nationality,
- Age as a range, and
- Location as a latitude-longitude pair

and these data-points are then fit to a specific demographic.

A user to demographic relation is a many to many relations where users and demographics are related to each other using an affinity field, computed at the time of user creation. If the user's personal data changes, his affinity to a demographic shall also change. For example, on the user's birthday, they might be shifted to another age range.

Another alternative is the usage of sensors in the environment or within the smart devices that the user carries which could lead to a more correct classification while removing direct user input from the equation as outlined in [13] where such a system is theorized.

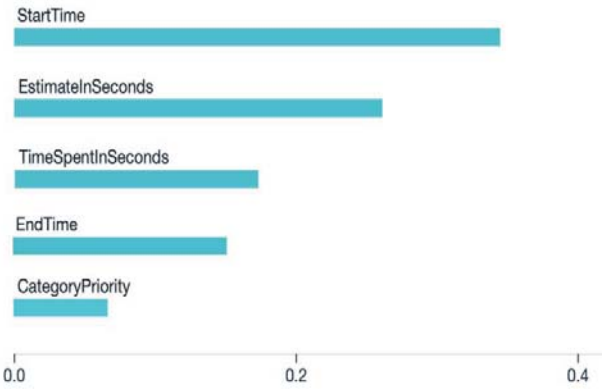


Fig. 2. Obtained predictor importance/weight for the neural network

Added demographic information could also be used to supply insights such as outlined in [14] where individuals have been classified by using the strong relationship between the regional language that the people speak and their demographic classification.

## VII. Results

The machine learning model [11] we have used for predicting the results is a neural network. In Fig 1, we see the network diagram which forms the neural network. The importance of each predictor is also generated and for a single user are categorypriority, categorynormalizedweight, estimateinseconds, timespentinseconds, starttime and endtime.

The results are provided in the 'ResltNrmlzdWght' column which stands for the results normalized weight, 'NormalizedScore' column which stands for the results normalized score, 'NormalzdStsfctn'column which stands for the results normalized satisfaction and 'NormalizedImpro' column which stands for the results normalized weight or importance to the user.

Fig. 2. depicts the final weights arrived at by running the neural network over the training set and Fig. 3. shows the distribution of improvement scores arrived at by the algorithm.

## VIII. Conclusion And Future Scope

The algorithm which finds the Mahalanobis distance using Singular Value Decomposition supplies an alternative to the standard method of finding the Mahalanobis distance which is used to measure correlation between the input and output. This helps us avoid the issue of not being able to compute the correlation data for singular or ill-conditioned multivariate data set matrices. Adding to this, the MRC (Multivariate – Reliability - Classifier) model is used to produce sound output which does not have results biased by extreme outliers to produce a covariance matrix for the Machine Learning algorithm.

The refined correlation data is used as an input for the Serverless Machine Learning API. The Watson Machine Learning API on IBM Cloud is thoroughly studied and found to be the most helpful choice for implementing machine learning based variants of the algorithm without the hardware or maintenance costs of a standard machine learning stack.

The data points and techniques outlined in this paper

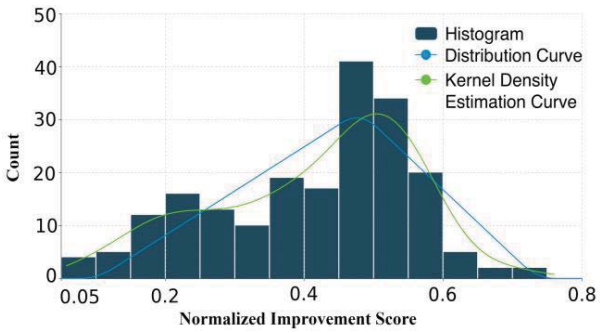


Fig. 3. Histogram depicting the distribution of improvements in results predicted by the neural network



Authorized licensed use limited to: ECOLE POLYTECHNIQUE DE MONTREAL. Downloaded on January 16,2022 at 01:14:31 UTC from IEEE Xplore. Restrictions apply.

can be used to run machine learning upon and classify different users' behavior, provide insights, and predict their future actions while correlating these current and future actions with the results they shall produce.

The algorithm for finding the correlation data can be augmented by adding special constraints which improve the reliability and usability of the output. One such constraint can be changing the weights used to compute the Mahalanobis Distance based on both positive and negative user feedback. This will result in greater user satisfaction and personalization.

We can also allow tuning of the Mahalanobis Distance based on contribution of more important parameters such as Time Spent, User Satisfaction and Task Completion Rates to the output. This will reduce the impact of variability in less important parameters on the overall output.

Using clever refinement methods, we can change the first training set and shorten the training period as well which lowers the possibility of generating insights on users' behavior based on false positives or false negatives. The impact of this constraint will be that there will not be an elongated training period during which the user could lose motivation to keep using the service.

## ACKNOWLEDGEMENT

We take this opportunity to express our sincere gratitude to all the people who have helped us carry out this survey and prepare this paper. We acknowledge the guidance of our professors in the Computer Engineering department at Vidyalankar Institute of Technology who helped us immensely while developing this paper and helped supply us the necessary facilities to successfully carry out this work. Their guidance and invaluable suggestions have been a major motivating factor and we thank them for their comments on an earlier version of the manuscript, although any errors are our own and should not tarnish the reputations of these esteemed people.

## REFERENCES

[1] Huang, Lei, Xianglong Liu, Yang Liu, Bo Lang, and Dacheng Tao. "Centered Weight Normalization in Accelerating Training of Deep Neural Networks." 2017 IEEE International Conference on Computer Vision (ICCV), 2017.

[2] P. Mahalanobis. "On the Generalized Distance in Statistics." In Proceedings of the National Institute of Science, India, volume 2, pages 49–55, 1936.

[3] A. Ker. "Stability of the Mahalanobis distance: A technical note." Technical Report CS-RR-10-20, Oxford University Computing Laboratory, 2010.

[4] M. Shashanka, M. Shen and J. Wang, "User and entity behavior analytics for enterprise security," 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, 2016, pp. 1867-1874.

[5] S. Krishnan and H. G. Kerkhoff, "Exploiting Multiple Mahalanobis Distance Metrics to Screen Outliers from Analog Product Manufacturing Test Responses," in *IEEE Design & Test*, vol. 30, no. 3, pp. 18-24, June 2013

[6] R. Kotecha, V. Ukani and S. Garg, "An empirical analysis of multiclass classification techniques in data mining," 2011 Nirma University International Conference on Engineering, Ahmedabad, Gujarat, 2011, pp. 1-5.

[7] Samal, Biswaranjan, Anil Kumar Behera, and Mrutyunjaya Panda. "Performance Analysis of Supervised Machine Learning Techniques for Sentiment Analysis." 2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS), 2017.

[8] Huang, Lei, Xianglong Liu, Yang Liu, Bo Lang, and Dacheng Tao. "Centered Weight Normalization in Accelerating Training of Deep Neural Networks." 2017 IEEE International Conference on Computer Vision (ICCV), 2017. doi:10.1109/iccv.2017.305.

[9] K. Kumar and S. K. Azad, "Database normalization design pattern," 2017 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON), Mathura, 2017, pp. 318-322.

[10] T. Hasuike, H. Katagiri and H. Tsuda, "Objective measurement for satisfaction values to sightseeing spots and route recommendation system," 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, 2016, pp. 002699-002704.

[11] S. Zhou and Y. Li, "Using Fuzzy Measures to Assess Service Satisfaction Values of Retailers," 2012 IEEE Asia-Pacific Services Computing Conference, Guilin, 2012, pp. 388-392.

[12] Refaat, Mamdouh. "Review of Data Mining Modeling Techniques." Data Preparation for Data Mining Using SAS, 2007, 15-27. doi:10.1016/b978-012373577-5/50005-3.

[13] H. İrem Türkmen, B. Diri, G. Biricik and R. Doğan, "Demographic information classification exploiting spoken language," 2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU), Antalya, 2011, pp. 13-16.

[14] A. R. Alharbi and M. A. Thornton, "Demographic Group Classification of Smart Device Users," 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, 2015, pp. 481-486.

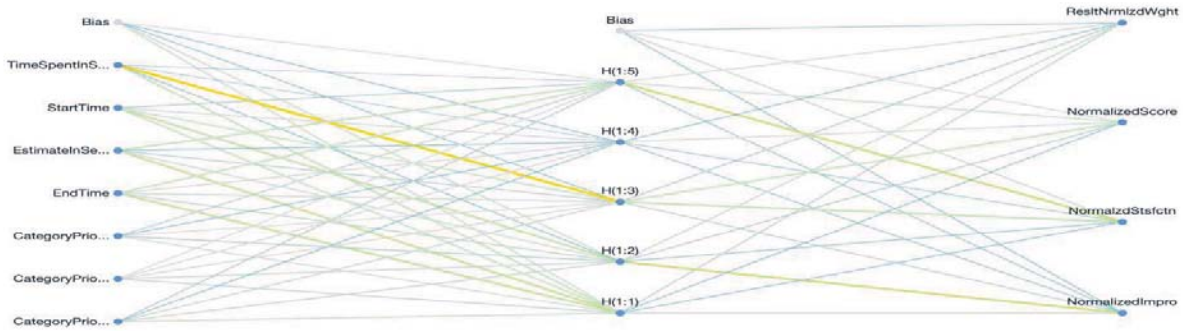


Fig. 1. Neural network structure chosen for predicting the users results



Authorized licensed use limited to: ECOLE POLYTECHNIQUE DE MONTREAL. Downloaded on January 16,2022 at 01:14:31 UTC from IEEE Xplore. Restrictions apply.