Asia Pacific Journal of Information Systems
Vol. 31 No. 2 (June 2021), 185-196

ISSN 2288-5404 (Print) / ISSN 2288-6818 (Online)
https://doi.org/10.14329/apjis.2021.31.2.185

Iowa Liquor Sales Data Predictive Analysis Using Spark

Ankita Paul[a], Shuvadeep Kundu[b], Jongwook Woo[c,*]

[a] *Graduate student, Computer Information Systems, California State University, Los Angeles, USA*
[b] *Graduate student, Computer Information Systems, California State University, Los Angeles, USA*
[c] *Professor, CIS Department, California State University, Los Angeles, USA*

ABSTRACT

The paper aims to analyze and predict sales of liquor in the state of Iowa by applying machine learning algorithms to models built for prediction. We have taken recourse of Azure ML and Spark ML for our predictive analysis, which is legacy machine learning (ML) systems and Big Data ML, respectively. We have worked on the Iowa liquor sales dataset comprising of records from 2012 to 2019 in 24 columns and approximately 1.8 million rows. We have concluded by comparing the models with different algorithms applied and their accuracy in predicting the sales using both Azure ML and Spark ML. We find that the Linear Regression model has the highest precision and Decision Forest Regression has the fastest computing time with the sample data set using the legacy Azure ML systems. Decision Tree Regression model in Spark ML has the highest accuracy with the quickest computing time for the entire data set using the Big Data Spark systems.

Keywords: Machine learning, Big Data, Predictive analysis, PySpark, Regression

Ⅰ. Introduction

Iowa is a state in the Midwestern United States. It is often viewed as a farming state, where agriculture is a small portion of the state's diversified economy. We wanted to look into a different part of Iowa's economy, which is the alcoholic beverage industry. The Iowa Alcoholic Beverages Division is the alcoholic beverage control authority for the U.S. state of Iowa. Since March 8, 1934, it has regulated the traffic in and maintained a monopoly on the wholesaling of alcoholic beverages in the state, thus making Iowa an alcoholic beverage control state. Therefore, the private retailer need to purchase the alcohol from the state before selling it to the consumers. If the private retailer can predict the sales per the state's volume, retail, and price of alcohol, it should help them to manage the business. We wanted to analyze the sales picture of this booming industry and predict the revenue that this industry can generate in the future based on its current records.

Our dataset is of size 4.13 GB, has 24 columns

[*] Corresponding Author. E-mail: jwoo5@calstatela.edu

and is in CSV format (“Iowa Liquor Sales Sales & Distribution”, n.d.). Since our aim is to predict the sales amount of liquor in Iowa, we have selected the ‘Sales (Dollar)’ column as our label column. Leveraging our knowledge of machine learning algorithms, we have built and run models to conduct the predictive analysis of Iowa liquor sales.

Furthermore, the dataset becomes too large to store and process using the legacy systems, which initiates adopting Big Data. It takes more than a day to develop predictive models using the legacy systems (Gupta et al., 2019; Purushu et al., 2018; Purushu and Woo, 2020) We can define Big Data as non- expensive frameworks, mostly on distributed parallel computing systems, storing large-scale data and processing it in parallel. A large-scale data means data of giga-bytes or more, which cannot be processed or expensive using traditional computing systems (Woo and Xu, 2011). Hadoop and Spark are popular Big Data platforms, and Spark is a popular computing engine for Big Data predictive analysis. Therefore, we have developed the Big Data models using Spark ML (Machine Learning) library for the entire data set and the legacy models with the sample data set using Azure ML systems.

Ⅱ. Related Work

Michael Salmon and Evan Lutins worked individually on this dataset to build predictive models and conduct predictive analysis with different goals and techniques.

Michael Salmon built a predictive model using the dataset of liquor sales in Iowa. However, he used a subset of the Iowa liquor sales data, which comprised sales records of the year 2015 and the first quarter (Jan – March) of 2016. The dataset comprised over 2.7 million rows of data using Linear Regression algorithm with the legacy machine learning method. Our paper is to use sales data from 2015 to build a model that could predict total 2016 sales based on Q1 2016 data. The data was analyzed using the records from 2012 to 2019 to forecast sales and have worked on Azure ML and Spark ML (Salmon, 2017). Our approach is to handle large scale data with Linear Regression, Decision Tree, Graident Boosted Tree Regression models using scalable parallel computing systems, Spark ML.

Evan Lutins conducted the predictive analysis by building a legacy linear regression model with 2.7 million rows of data. It was only for the remaining part of 2016 while our paper has utilized the records till 2019 for prediction. Also, he used Scikit-Learn to run the linear regression model and Pandas library to explore the data while we have built and run Linear Regression, Decision Tree, Graident Boosted Tree Regression models in Azure ML and scalable parallel computing model in Spark ML (Lutins, 2017).

Therefore, in the paper, we implement Predictive Models using distributed parallel computing systems in Spark ML to afford and compute large scale data set, which is called Big Data.

Ⅲ. Machine Learning Algorithms

The models that we have built in the paper are mostly based on Machine Learning algorithms given by both Azure ML and Spark ML.

The Linear Regression, Decision Forest Regression, and the Boosted Decision Tree Regression models are used in Azure ML Studio. Cross-validate module and Tune Model Hyperparameters module are used for training the model. Permutation Feature Importance module is used to check the scores of the importance of the features based on which features were pruned to improve the performance.

In Spark ML, we used Linear Regression model which is a fundamental algorithm to predict numeric value, where TrainValidationSplit was used for training purpose. We have also used Decision Tree Regression and Gradient Boosted Tree Regression in Spark ML which have been popular to predict more accurate numeric value comparing to linear regression algorithm. RMSE and Coefficient of Determination are used as an evaluation parameter.

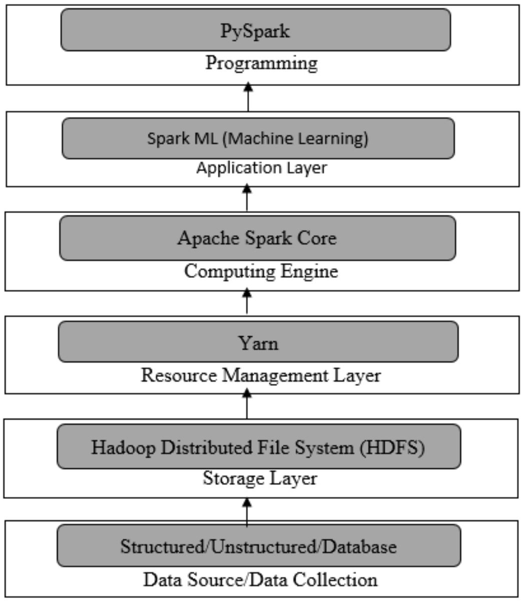
Ⅳ. Our Work

We tried to compare the results and run times by using three different algorithms, both in Azure ML and Spark ML. We have used the independent variables from our dataset to predict the values of the outputs or dependent variable, which is a form of Supervised learning. Since our output variable is of quantitative nature, we have taken resort to Regression models for the prediction task.

We built models in both Azure ML and Spark ML to predict the “Sales (Dollars)” in Azure ML and “SaleInDollars” in Spark ML.

4.1. Azure ML

Due to storage issue in Azure ML, we sampled the original dataset of 4.13 GB to 6.45 MB on Azure ML Studio using Spark ML code. The data is stored at Hadoop Distributed File Systems in Hadoop cluster with 3 nodes because it can sample the data in several seconds using parallel computing and allows us to build prediction models in parallel computing with Spark ML. The models use Partition and Sample module with stratified sampling set to True by selecting the “Sale (Dollars)” column to ensure that the sampled dataset is a true representative of the original dataset. However, the experiments were failing due to memory exhaustion, so we further sampled the 6.45 MB dataset at the rate of 0.095 and used it for our experiments.


PySpark
Programming
Spark ML (Machine Learning)
Application Layer
Apache Spark Core
Computing Engine
Yarn
Resource Management Layer
Hadoop Distributed File System (HDFS)
Storage Layer
Structured/Unstructured/Database
Data Source/Data Collection

<Figure 1> Work Flowchart of Spark Architecture

<Table 1> Settings of Sampling of Original Dataset at 0.01

Settings features	Settings values
Partition or Sample mode	Sampling
Rate of Sampling	0.01
Random seed for sampling	1234
Stratified split for sampling	True
Stratification key column • Selected columns: Column names:	Sale (Dollars)

4.1.1. Linear Regression (Azure ML)

We selected all columns, all features (the sampled dataset contained 17 columns) and selected Sale

(Dollars) as label column. After splitting the dataset at 70:30 train – test ratio using Split Data module, we used Cross Validate Model and Tune Model Hyperparameters to train the model. We selected RMSE as metric for measuring performance and had a run time of 37.24 minutes.

We also used Permutation Features Importance module to check the important features affecting the performance of the model.

<Table 2> Feature ImportancE Table (Linear Regression)

Features	Scores
Bottles Sold	380.17
Volume Sold (Liter)	77.257
Pack	66.775
Vendor Number	62.78
Category	51.802
Bottle Volume (ml)	44.811
Item Number	19.236
State Bottle Cost	12.448
State Bottle Retail	9.823
City	1.0707
County Number	0.924
Zip Code	0.777
Date	0.243
Invoice/Item Number	0
Store Number	-1.846
Store Location	-1.9402

Both the Cross Validate Model and Tune Model Hyperparameters performed equally well.

<Table 3> Evaluation Results of Linear Regression (Azure ML)

Metrics	Cross Validation	Tune Model Hyperparameters
RMSE	137.145669	137.145669
Coefficient Of Determination	0.925955	0.925955

Pruning features Invoice/Item number, Store Number and Store Location resulted in a higher RMSE value of 240.403611 and lower Coefficient of Determination value of 0.772484.

Hence the model before pruning the features gave better accuracy for our prediction.

4.1.2. Boosted Decision Tree Regression (Azure ML)

We selected all columns, all features from the sampled dataset and selected “Sale (Dollars)” as label column. We split the data at 70:30 ratio for training and testing. We chose Single Parameter as Create trainer mode, Maximum number of leaves per tree as 20, Total number of trees constructed as 100.

We used Cross Validate Model, Tune Model Hyperparameters and Permutation Features Importance modules. We selected RMSE as metric for measuring performance and it had a run time of 2.17 minutes.

The Tune Model Hyperparameter performed better than Cross validation, with lower RMSE value and higher Coefficient of Determination value.

<Table 4> Evaluation Results of Boosted Decision Tree Regression (Azure ML)

Metrics	Cross Validation	Tune Model Hyperparameters
RMSE	267.974043	173.140216
Coefficient Of Determination	0.717307	0.881988

Having a look at the Features Importance table, we could find out the features that affected the performance of the model the most and the features that had least importance.

Pruning the less important features Invoice/Item Number, Date, Store Location, County Number, and

Zip Code led to decrease in RMSE value by very negligible amount.

<Table 5> Feature Importance Table (Boosted Decision Tree Regression)

Features	Scores
Bottles Sold	454.818329
Pack	81.371853
Vendor Number	75.608935
Category	54.132026
Bottle Volume (ml)	39.98075
Volume Sold (Liters)	6.885084
State Bottle Cost	2.324521
Store Number	0.948541
City	0.295601
Item Number	0.219255
Zip Code	0.167606
State Bottle Retail	0.015421
Invoice/Item Number	0
Date	0
Store Location	0
County Number	-0.122389

4.1.3. Decision Forest Regression (Azure ML)

We selected all columns, all features from the sampled dataset, selected “Sale (Dollars)” as label column and split the data in 70:30 ratio for training and testing. In the Decision Forest Regression module, we chose Bagging as Resampling Method, Single Parameter as Create trainer mode, Number of decision trees as 8, Maximum depth of the decision trees as 32, Number of random splits per node as 128 and Minimum number of samples per leaf node as 4. We used Cross Validation, Tune Model Hyperparameters, Permutation Feature Importance modules and the experiment had a run time of 36.79 seconds.

Evaluation results revealed that Tune Model Hyperparameter performed better with lower RMSE value and higher Coefficient of Determination than Cross Validate Model.

<Table 6> Evaluation Results of Decision Forest Regression (Azure ML)

Metrics	Cross Validation	Tune Model Hyperparameters
RMSE	338.365264	314.118924
Coefficient Of Determination	0.549286	0.611566

The scores of feature importance are as follows.

<Table 7> Feature Importance Table (Decision Forest Regression)

Features	Scores
Bottles Sold	168.902577
Volume Sold (Liters)	45.997167
Bottle Volume (ml)	27.05422
Pack	15.378501
State Bottle Cost	2.951751
Vendor Number	2.912818
County Number	1.690925
Category	1.339969
Item Number	1.106864
State Bottle Retail	0.719819
City	0.201144
Invoice/Item Number	0
Date	0
Zip Code	-0.030835
Store Location	-0.033807
Store Number	-0.26712

Excluding the columns Store Number, Store Location, Zip Code, Date, Invoice/Item Number resulted in considerably decreased RMSE value and increased Coefficient of Determination value. However, Cross Validation Model performed better this time.

<Table 8> Evaluation results of Decision Forest Regression after pruning features

Metrics	Cross Validation	Tune Model Hyperparameters
RMSE	246.037976	271.932532
Coefficient Of Determination	0.761695	0.708894

4.2. SparkML

We worked on both Databricks Community Edition and Oracle BDCE to run our experiments using Spark ML algorithms.

In Databricks CE, we used a 41.1 MB sized sample dataset, imported it as a table after creating a Test Cluster in Databricks and ran the experiments on the records of that table.

In Oracle BDCE, due to memory exhaustion issue, we could not run the experiment on the original big dataset but used a sample dataset of size 238 MB.

In these models using Scalable Parallel computing systems of Spark ML, the following combinations of parameters are used to find out the optimal Linear Regression model: Regularization Parameters: [0.3, 0.1, 0.01], maxIter: [10, 5]. To implement optimal Decision Tree, the combination of max depth and max bins parameters are calculated: maxDepth: [2, 5, 10, 20, 30], maxBins: [10, 20, 40, 80, 100]. For Gradient Boosted Tree Regression, parameters were tuned in the following way: maxDepth [2, 5, 10], maxBins, [10, 20, 40], maxIter, [5, 10, 20] and minInfoGain, [0.0, 0.1, 0.2].

4.2.1. Linear Regression (Databricks CE - Spark ML)

To predict the sales amount of Iowa liquor sales with Linear Regression algorithm, we used the columns with Integer and Double data types and cast them all into Double data type. The features used were Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters and the label column selected was “SaleInDollars” since we are going to predict the sales amount. We split the data into 70:30 ratio of train and test datasets.

Vector Assembler was used to assemble the features in a vector. Linear Regression was used and pipeline was defined. In the Parameter Grid, the parameters regParam and maxIter were defined. We used Train ValidationSplit with train ratio of 0.8 to train the model. Pipeline was used as an estimator and run with fit() method on training dataset. Regresion Evaluator was used to retrieve the RMSE value that resulted in 129.02 in 4.07 minutes.

4.2.2. Gradient Boosted Tree Regression (Databricks CE - Spark ML)

In Gradient Boosted Tree Regression algorithm, we used the features Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters and the label column as “SaleInDollars”, all cast into Double data types. We split the data into 70:30 ratio of train and test datasets. Vector Assembler was used to assemble the features in a vector. Since Gradient Boosted Tree Regression algorithm does not require any normalization or scaling of the features separately, we did not use Standard Scaler to scale the features. GBTRegressor was used and Pipeline was defined. In the Parameter Grid, the parameters maxDepth, maxBins, maxIter and minInfoGain were defined. TrainValidation Split method with train ratio of 0.8 was used. The pipeline was used as an estimator and was run with fit() method on training dataset to train the model. RegressionEvaluator was used as an evaluator to retrieve the RMSE value which was 107.52. The whole experiment took 4.14 minutes to run.

4.2.3. Decision Tree Regression (Databricks CE - Spark ML)

In Decision Tree Regression algorithm, we used the features Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters and the label column as "SaleInDollars", all cast into Double data types. We split the data into 70:30 ratio of train and test datasets. Vector Assembler was used to assemble the features in a vector. The features were scaled by using Standard Scaler. DecisionTreeRegressor was used and Pipeline was defined. In the ParamGridBuilder, maxDepth and maxBins parameters were defined. TrainValidation Split method with train ratio of 0.8 was used. The pipeline was used as an estimator and was run with fit() method on training dataset to train the model. RegressionEvaluator was used as an evaluator to retrieve the RMSE value which was 84.34024. The experiment took 1.86 minutes to run.

4.2.4. Linear Regression (Oracle BDCE - Spark ML)

We used Linear Regression to predict sale amounts. The features used were Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters and the label column was "SaleInDollars", all cast into Double data types. We split the data into 70:30 ratio of train and test datasets. Vector Assembler, LinearRegression was used and pipeline was defined. Both Cross Validation and TrainValidationSplit was used and RegressionEvaluator was used to retrieve RMSE. The model with CrossValidator took 4.15 minutes to run and gave a RMSE value of 182.1768. The model with TrainValidationSplit performed a bit better with lower runtime of 45 seconds and bit lower RMSE value of 181.1703.

4.2.5. Gradient Boosted Tree Regression (Oracle BDCE - Spark ML)

In Gradient Boosted Tree Regression algorithm, we used the features Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters and the label column as "SaleInDollars", all cast into Double data types. We split the data into 70:30 ratio of train and test datasets. Vector Assembler, GBTRegressor was used and pipeline was defined. The parameters maxDepth, maxBins, maxIter and minInfoGain were defined in the Parameter Grid. TrainValidation Split method with train ratio of 0.8 was used. RegressionEvaluator retrieved RMSE value of 90.576 and the whole model took 3.23 minutes to run.

4.2.6. Decision Tree Regression (Oracle BDCE - Spark ML)

In Gradient Boosted Tree Regression algorithm, we used the features, [Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters] and the label column as "SaleInDollars", all cast into Double data types. We split the data into 70:30 ratio of train and test datasets. Vector Assembler, DecisionTreeRegressor was used and pipeline was defined. In the ParamGridBuilder, maxDepth and maxBins parameters were defined. We used both CrossValidator with 5 folds and

TrainValidation Split method with train ratio of 0.8 in separate experiments. RegressionEvaluator retrieved RMSE value of 103.955849 using TrainValidationSplit in 20 seconds, while CrossValidation method gave a value of 63.72514.

Ⅴ. Experimental Results

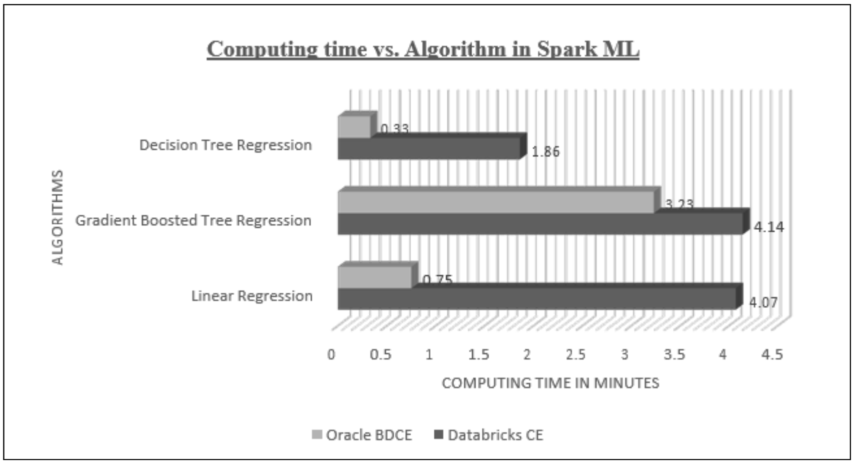
<Table 9> and <Table 10> show the experimental result, which compares the computing time in the traditional and Big Data systems. It also presents the models' accuracy in Root Mean Square Error

<Table 9> Comparison of Models with the data size 6.45 MB in Azure ML

Azure ML			
Metrics	Linear Regression	Boosted Decision Tree Regression	Decision Forest Regression
RMSE	137.14566	171.97299	246.03797
Coefficient of Determination	0.925955	0.883574	0.761695
Run time	37:24 minutes	2.17 minutes	36.798 seconds

<Table 10> Comparison of Models with the Data Size 41.1 MB on Databricks CE and 238 MB on Oracle BDCE in Spark ML

Metrics	Linear Regression	Gradient Boosted Tree Regression	Decision Tree Regression
Databricks CE			
RMSE	129.02	107.52	84.34024
Run time	4.07 mins	4.14 mins	1.86 mins
Oracle BDCE			
RMSE	181.17	90.576	63.72514
Run time	45 sec	3.23 mins	20 sec



<Figure 2> Bar Chart Showing the Comparison between Computing Times of the Different Algorithms in Databricks CE and Oracle BDCE

(RMSE) and the Coefficient of Determination.

<Figure 2> presents a bar chart showing the differences in computing time on the two different Spark ML platforms: Databricks CE and Oracle BDCE. The cluster in Databricks CE is composed of one node, while the cluster in Oracle BDCE consists of three nodes. Based on <Table 9> and <Figure 2>, we can observe that Decision Tree and Linear Regression models running in Oracle BDCE are about six times faster than in Databricks CE. And, Oracle BDCE is 1.3 times faster in Gradient Boosted Tree Regression model.

We can summarize the experimental result as:

1. Linear Regression performed better in Spark ML than in Azure ML. While in Azure ML, after pruning features, the RMSE value for Linear Regression had increased; in Spark ML, pruning some more features led to lower RMSE value than in Azure ML.
2. In Azure ML, the model with Linear Regression performed best with lowest RMSE value and highest Coefficient of Determination.
3. In Spark ML using PySpark CLI, the model with Decision Tree Regression performed best with lowest RMSE value as well as lowest run time.
4. Spark Big Data models are linearly scalable with a number of nodes.

Ⅵ. Conclusion

We have implemented prediction models of liquor sale data in Iowa using the legacy and Big Data systems, which are Azure ML and Spark ML, respectively. Furthermore, we present the Big Data architecture, which can store massive data and allow the iterative machine learning possible in distributed parallel computing using Spark ML. The data size is 4.13 GB, which is sampled to 6.45MB, 41.1MB, and 238MB for Azure ML, and Spark ML in Databricks CE and Oracle BDCE. The legacy Azure ML can afford small size data set such as 6.45 MB, and the Linear Regression model has the highest accuracy, RMSE: 137, with a computing time, 37 minutes. For the data size 41.1 MB, which is 6 – 7 times bigger than 6.45 MB, Decision Tree Regression models using Spark ML in Databricks has the highest accuracy with RMSE 84.34024 and the computing time 1.86 minutes. For the data size 238 MB, which is six times bigger than 41.1 MB, Decision Tree Regression models using Spark ML in Oracle BDCE has the highest accuracy with RMSE 63.725 and the computing time 20 seconds. We observed that the Spark Big Data platform is linearly scalable because it is six times faster than Databrick CE's while Oracle BDCE has more memory and cores.

In the future, we plan to build Random Forest Regression and Factorization Machine Regression models. Besides, we leverage Big Data systems to implement Deep Learning regression model. These models will show the interesting comparison result with the models in the present paper.

<References>

[1] Gupta, N., Le, H. A., Boldina, M., and Woo, J. (2019). Predicting fraud of AD click using traditional and spark ML. *KSII The 14th Asia Pacific International Conference on Information Science and Technology* (APIC-IST), pp.24-28.

[2] Iowa Liquor Sales Sales & Distribution (n.d.). Retrieved 2019 from https://data.iowa.gov/Sales-Distribution/Iowa-Liquor-Sales/m3tr-qhgy

[3] Lutins, E. (2017). Predicting-Iowa-Liquor-Sales. *GitHub*, 2017 [Online]. Retrieved from https://github.com/elu

tins/Predicting-Iowa-Liquor-Sales

[4] Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D. B., Amde, M., Owen, S., and Xin, D. (2015). *MLlib: Machine learning in apache spark*. arXiv preprint arXiv:1505.06807.

[5] Purushu, P., Melcher, N., Bhagwat B., and Woo, J. (2018). Predictive analysis of financial fraud detection using azure and spark ML. *Asia Pacific Journal of Information Systems (APJIS), 28*(4), 308-319.

[6] Purushu, P., and Woo, J. (2020). Financial fraud detection adopting distributed deep learning in big data. *KSII The 15th Asia Pacific International Conference on Information Science and Technology* (APIC-IST) 2020, July 5-7 2020, Seoul, Korea, pp.271-273.

[7] Salmon, M. (2017). Predictive modeling with iowa state liquor sales data. *Towards data science*, 2017[Online]. Retrieved from https://towardsdatascience.com/predictive-modeling-with-iowa-state-liquor-sales-data-e45342081b83

[8] Woo, J., and Xu, Y. (2011). Market basket analysis algorithm with map/reduce of cloud computing. *The 2011 International Conference on Parallel and Distributed Processing Techniques and Applications* (PDPTA 2011), Las Vegas.

[9] Woo, J. (2013). Market basket analysis algorithms with mapreduce. *DMKD-00150, Wiley Interdisciplinary Reviews Data Mining and Knowledge Discovery, 3*(6), 445-452.

<Appendix> Hardware Specifications

In the paper, we have used cloud computing systems: Microsoft Azure Machine Learning Studio for Azure ML algorithms and Databricks Community Edition to implement Spark ML algorithms. Furthermore, we have used Hadoop cluster on the Oracle Big Data Cloud platform to run PySpark commands for the predictive analysis. The specifications are as follows:

<Table 11> Hardware Specifications

Azure	Databricks	Oracle BDCE
Memory – 10 GB **Nodes** - 1 **Max no. of modules per experiment** - 100	**Memory** – 15.3 GB **Nodes**- 1 **Driver** - (2 cores, 1 DBU) **Databricks Runtime Version** – 6.5 (Scala 2.11, Spark 2.4.5) Python version - 3	**Memory** – 247.625312 GB **Storage** – 1003.6 GB **Nodes** – 3 **No. of processors** – 32 Cluster version - Hadoop 2.7.1.2.4.2.0-258 **CPU speed** – 2.20 Ghz **Python version** – 2.7.14 **Spark version** – 2.1.0.2.6.0.3-10

◆ About the Authors ◆

Ankita Paul

Ankita Paul is a Graduate student of Computer Information Systems at California State University, Los Angeles in United States of America. She has a Master's & Bachelor's in Economics from Jadavpur University in India. Her research interests include Big Data Analytics, Python Automation, Statistical analysis, Business Intellingence, Artificial intelligence & Machine learning.

Shuvadeep Kundu

Shuvadeep Kundu is a Graduate student of Computer Information Systems at California State University, Los Angeles in United States of America. He has a Master's & Bachelor's in Electronics and Communications Engineering from West Bengal University of Technology in India. He has a 15 years of job experience in the field of IT Consulting, Distributed Computing, Software Defined Networks, Cloud Computing & Computer Networks. He is currently employed as a Technical Consultant with Cisco Systems US for the past 13 years. His research interests include Network Function virtualization, Software Defined Networks, Automating networks in Ansible, Puppet, Chef with containerization via Docker and Kubernetes orchestration.

Jongwook Woo

Dr. Jongwook Woo received his Ph.D. from USC and went to Yonsei University. He is a Professor at CIS Department of California State University Los Angeles and serves as a Technical Advisor of Teradatsa, Council Member of IBM Spark Technology Center and as a president at KSEA-SC. He has consulted companies in Hollywood: CitySearch, ARM, E!, Warner Bros, SBC Interactive. He published more than 40 papers and his research interests include Scalable Deep Learning, Big Data Analysis and Prediction. He has been awarded Teradata TUN faculty Scholarship and received grants from DataBricks, NVidia, Amazon,

Submitted: December 8 2020; 1st Revision: April 1, 2021; Accepted: April 15, 2021