Iowa Liquor Sales Predictive Analysis

Shuvadeep Kundu, Ankita Paul, and Houhou Kevin Xu
Department of Information Systems,
California State University, Los Angeles
Tel: (626)500-8930
email: skundu@calstatela.edu, apaul11@calstatela.edu, hxu9@calstatela.edu

Abstract: The aim of the project is the application of several Machine Learning models to Iowa Liquor Sales data in order to analyze and predict sales. For this purpose we have used algorithms in both Azure ML and Spark ML. We plan to use the dataset on data.iowa.gov for our Iowa Liquor Sales Predictive Analysis. This dataset contains the spirits purchase information of Iowa Class "E" liquor licensees by product and date of purchase from January 1, 2012 to 2019. The dataset is updated in real-time but we decided not to use the dataset including 2020 because it will not have large enough sample size. The size we used is 4.13 GB. The dataset can be used to analyze total spirits sales in Iowa of individual products at the store level. Iowa is an alcohol beverage control state and the state maintains a monopoly on wholesaling of alcohol throughout the State. Private retailers must purchase their alcohol from the state before selling it to individual consumers. We have used machine learning algorithms to build models for predicting the amount of sales of liquor.

1. Introduction

The main motive of our project is to use the Iowa Liquor Sales dataset (splitting into sampling, testing, and training datasets) to perform Predictive Analysis using various machine learning algorithms. By running the different algorithms on Azure ML, Spark ML on Databricks CE and PySpark CLI on Oracle BDCE and comparing their RMSE and run-time, we are able to build the best model for predictive analysis with the lowest RMSE. Iowa is a state in the Midwestern United States. Iowa is often viewed as a farming state, where agriculture is actually a small portion of the state's diversified economy. We wanted to look into a different part of Iowa's economy: the alcoholic beverage industry. Population size: 3.17 Million (2020) and most Populous City: Des Moines, 215k. (2018) 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. This Predictive Analysis is important because it gives the State of Iowa greater knowledge and more accurate prediction on the future demand of various types of liquor to stock for wholesaling.

2. Related Work

One of the related works by Michael Salmon [1] was to use sales data from 2015 to build a model that could predict total 2016 sales based of 1Q 2016 data and he analyzed the data using the pandas library in Python. Another Related work is on Rakam [2] to examine Iowa's drinking pattern in terms of popular liquors, popular stores etc. using the data analysis software Rakam. The third work is by Jim Lung [3], his goal in the project was to determine a geographic location (county) in Iowa that will yield the highest amount of liquor sales. Our focus was mainly on predictive analysis. We used Azure ML and Spark ML platforms to predict sales. In contrast, our research is about predictive analysis. We did predictive analysis by using various machine learning models and compare the model to each other and find out which model has the lowest RMSE with the least run-time and the most accuracy. (Since the model that has the lowest RMSE should give the most accurate prediction on sales in this predictive analysis)

3. Background/Existing work

In our project we have implemented several algorithms in AzureML and SparkML. Most of our models are based on previous existing works.

3.1 Regression

The Decision Forest Regression and the Boosted Decision Tree Regression is based on the study of the prediction of heating load. In the prediction of heating load for the energy efficiency dataset, we used Linear Regression and Decision Forest Regression model. We used cross-validation module and hypertune module for the training purpose. We also used Permutation feature module to check the features importance and accordingly pruned the features. In Spark ML, for our project we have used Decision Tree Regression and Gradient Boosted Tree Regression. In the lab work we didn't perform the above two models but the work is similar to the lab work where we predicted arrival delay of the flight dataset. We used the Linear Regression model in the lab and crossvalidation module was used for the training purpose. RMSE was used as an evaluation parameter.

Azure ML	Databricks CE	Oracle BDCE
• Memory – 10 GB • Nodes - 1 • No. of experiment - • 100	• Memory – 15.3 GB • Python 3 • Driver (2 cores, 1 DBU)	• Memory – 242.821 GB • Storage – 1003.6 GB • Nodes – 3 (1 Master node, 2 Slave nodes) • OCPU – 32 • CPU Speed – 2.20GHz

4. Our work

Data collection ➡ Data processing and transformation ➡ splitting the data ➡ Training and testing ➡ Evaluation

To start with, we uploaded the dataset, cleaned it and sampled it at rate of 0.095 and ran our experiment in Azure ML. Refer to Figure 1. Thereafter we split the data into training set and testing set with Split Data module. As mentioned earlier we have used different algorithm to proceed further. Lastly, we Train, Score, and Evaluate the model to find the best model.

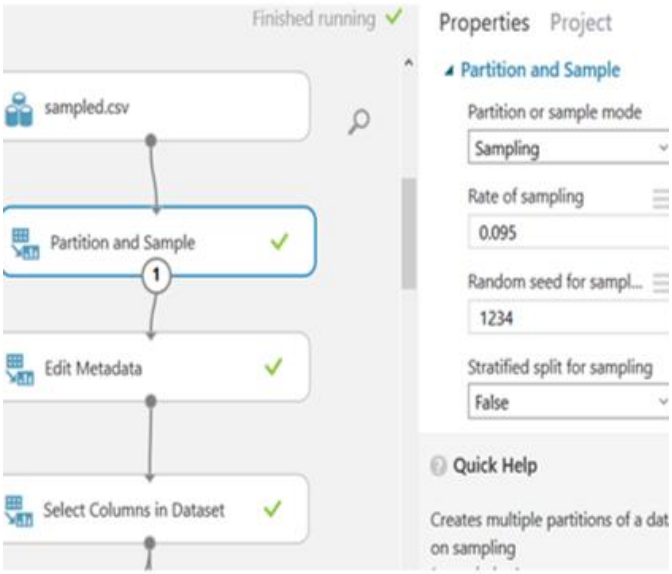


Figure 1. Sampling of 0.095

4.1 Azure ML Linear Regression

In simple linear regression a single independent variable is used to predict the value of a dependent variable. In multiple linear regression two or more independent variables are used to predict the value of a dependent variable. The difference between the two is the number of independent variables. We selected all columns, all features (the sampled dataset contained 17 columns) and selected Sale (Dollars) as label column. We used Cross Validate Model, Tune Model Hyperparameters, and Permutation Features Importance modules. We selected RMSE as metric for measuring performance and had a run time of 37.24 minutes and RMSE of 137.146. Refer to Figure 2.

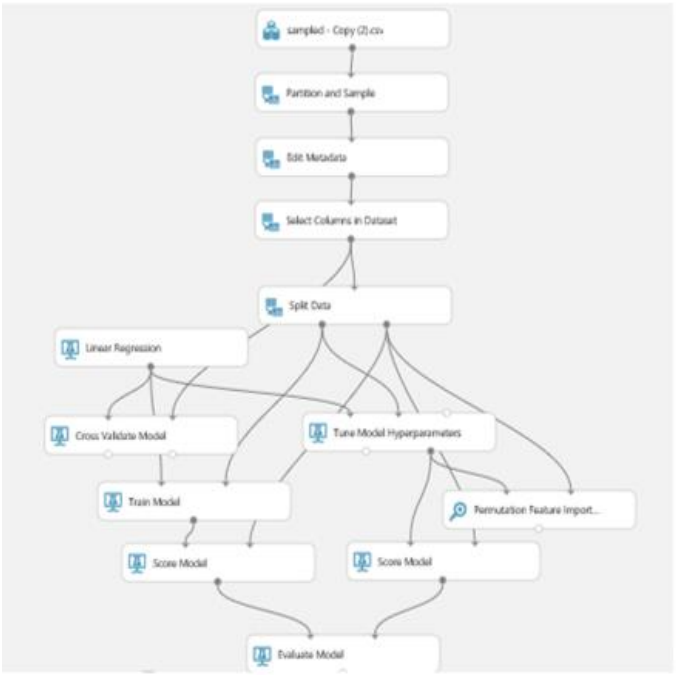


Figure 2. Azure ML Linear Regression

4.2 Azure ML Permutation feature importance

The permutation importance features found Invoice/Item Number, Date, Store Location, County Number, and Zip Code have lesser importance and after eliminating them, RMSE decreased by negligible amount. Refer to Figure 3.

Bottles Sold	380.178966
Volume Sold (Liters)	77.257687
Pack	66.775842
Vendor Number	62.780436
Category	51.802287
Bottle Volume (ml)	44.811355
Item Number	19.236181
State Bottle Cost	12.448437
State Bottle Retail	9.823359
City	1.070746
County Number	0.924101
Zip Code	0.777995
Date	0.243088
Invoice/Item Number	0
Store Number	-1.84673
Store Location	-1.940274

Figure 3. Permutation Feature Importance Columns in Linear Regression

4.3 Azure ML Linear regression features pruned

We used Linear regression pruned feature to prune Invoice/Item number, Store Number and Store Location but that resulted in a higher RMSE value at 240.403. Refer to Figure 4.

Metrics

Mean Absolute Error	176.678729
Root Mean Squared Error	240.403611
Relative Absolute Error	0.413495
Relative Squared Error	0.227516
Coefficient of Determination	0.772484

Metrics

Mean Absolute Error	176.67872
Root Mean Squared Error	240.4036
Relative Absolute Error	0.413495
Relative Squared Error	0.227516
Coefficient of Determination	0.772484

Figure 4. RMSE after pruning

4.4 Azure ML Boosted Decision Tree Regression

We have used boosted decision tree regression model to avoid over-fitting and underfitting of the model as this helps us by fitting the residual of the trees that precedes it. Boosting in a decision tree ensemble tends to improve accuracy with small risk of less coverage. We selected all columns, all features (the sampled dataset contained 17 columns) and selected Sale (Dollars) as label column. We used Cross Validate Model, Tune Model Hyperparameters, and Permutation Features Importance modules. Refer to Figure 5.

We selected RMSE as metric for measuring performance and it had a run time of 2.17 minutes and a RMSE of 173.140 before pruning. The values in the folds are close to the values shown in the Mean row. Finally, the values in the Standard Deviation row are much smaller than the corresponding values in the Mean row. These consistent results across the folds indicate that the model is insensitive to the training and test data chosen and is likely to generalize well. After pruning, RMSE decreased by a small amount to 171.973. Refer to Figure 6.

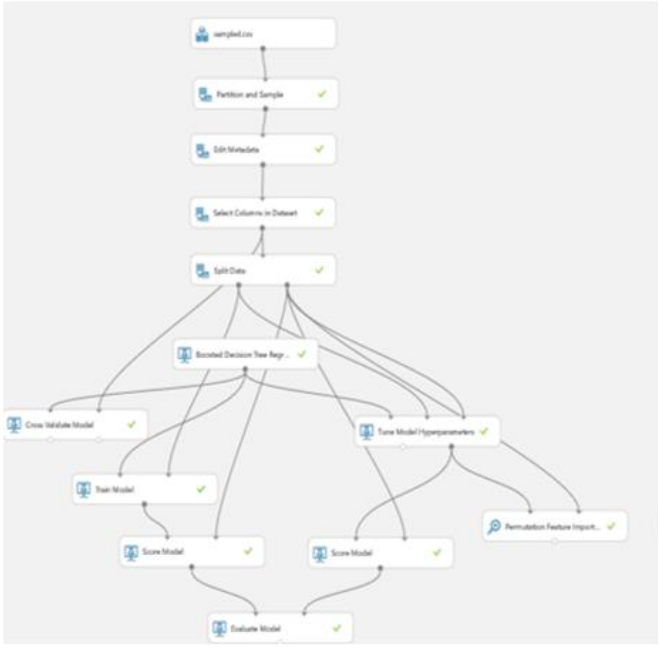


Figure 5. Boosted decision Tree

Metrics

Mean Absolute Error	119.92497
Root Mean Squared Error	173.140216
Relative Absolute Error	0.28067
Relative Squared Error	0.118012
Coefficient of Determination	0.881988

Metrics

Mean Absolute Error	119.084776
Root Mean Squared Error	171.97299
Relative Absolute Error	0.278704
Relative Squared Error	0.116426
Coefficient of Determination	0.883574

Figure 6. RMSE of 173.140 before pruning and 171.973 after pruning

4.5 Azure ML Decision Forest Regression

The we selected all Columns and all features. We used Cross Validation, Tune Model Hyperparameters, Permutation Feature Importance and had a run time of 36.79 seconds and a RMSE of 271.933. Refer to Figure 7.

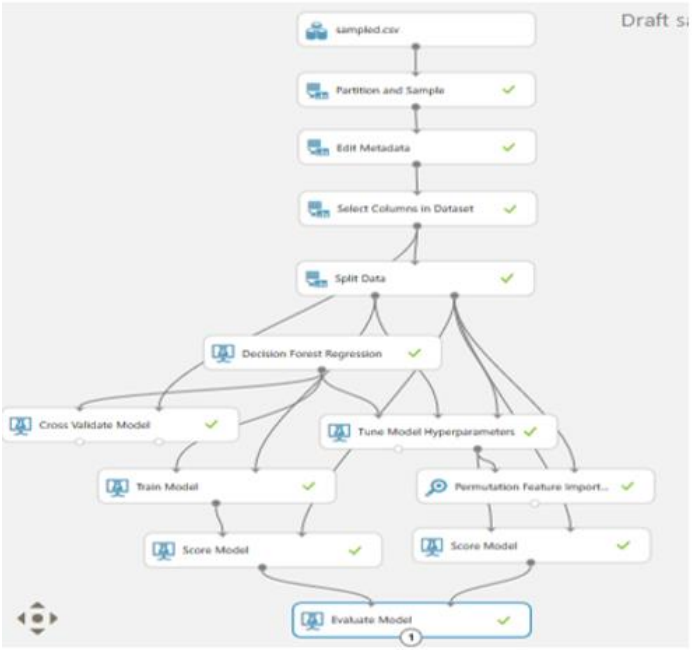
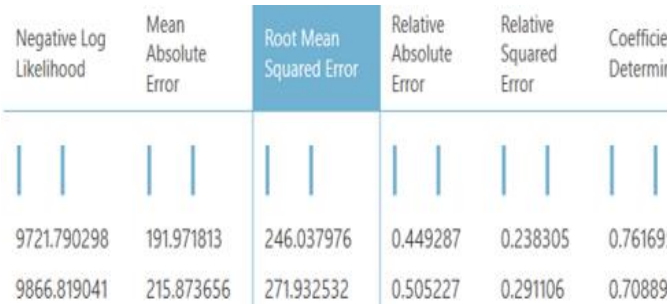


Figure 7. Decision Forest

4.6 Azure ML After pruning features

Then we pruned and eliminated County Number, Store Location, Zip Code, Date, and Invoice/Item Number. After pruning, our RMSE decreased by considerable amount to 246.038 Refer to Figure 8.



Negative Log Likelihood	Mean Absolute Error	Root Mean Squared Error	Relative Absolute Error	Relative Squared Error	Coefficie Determi
9721.790298	191.971813	246.037976	0.449287	0.238305	0.76169
9866.819041	215.873656	271.932532	0.505227	0.291106	0.70889

Figure 8. RMSE after pruning

4.7 Databricks - Spark ML Linear Regression

We used a sample data size of 41.1MB. We split the data 70:30 Train-Test. Relevant Columns Selected are Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters, and SaleInDollars (as label). We used Vector Assembler, LinearRegression, Define Pipeline, CrossValidator, and TrainValidatorSplit With parameters defined are regParam and maxIter.We had a run Time of 4.07 minutes. We used RegressionEvaluator to retrieve a RMSE_CV: 141.968 and RegressionEvaluator to retrieve RMSE_tvs: 141.968

4.8 Databricks - Spark ML Gradient Boosted Tree Regression

We used a **s**ample data size of 41.1MB Split Data 70:30 Train-Test with relevant columns Selected: Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters, and SaleInDollars. We used Vector Assembler, GBTRegressor, and Define Pipeline with Parameters defined: maxDepth, minInfoGain, and stepSize. We were able to use TrainValidationSplit, train the model, test the model, and had a run time: 4.14 minutes. We used RegressionEvaluator to retrieve a RMSE_GBT: 115.382

4.9 Databricks - Spark ML Decision Tree Regression

We had a sample data size of 41.1MB with relevant columns selected: Pack, BottleVolumeInMl, StateBottleCost, StateBottleRetail, BottlesSold, VolumeSoldInLiters, and SaleInDollars. We split the data in 70-30 Train-Test ratio and used Vector Assembler, DecisionTreeRegressor, and Define pipeline. We used ParamGridBuilder, CrossValidator to train the model and test the model with a run time of 1.86 minutes. Last we used RegressionEvaluator to retrieve a RMSE_DTR: 94.737

4.10 PySpark CLI on Oracle BDCE - Spark ML Linear Regression

We ran the same codes in PySpark CLI on Oracle BDCE which we used in Databricks to train the model. We used Linear Regression, Gradient Boosted Tree Regression, and Decision Tree Regression to train the models. The first step in Workflow is to load the data. Then we have imported Spark Libraries, followed by extracting features from raw data and splitting data. Post that we did Cross validation for training and transforming so that testing can be performed on it. Last step includes Evaluation in the workflow of Spark ML. We then used Spark ML to run the model with CrossValidator and it took 4.15 to run. It gave lower RMSE value than in Databricks at RMSE_CV: 182.177**.** We also ran the model with TrainValidationSplit and it took 45 seconds to run. It gave lower RMSE value than in Databricks as well. Next we ran the model with Cross Validation and got a RMSE_tvs; 181.170 which is higher than what we got in Databricks.

4.11 PySpark CLI on Oracle BDCE - Spark ML Gradient Boosted Tree Regression

Then we ran Gradient Boosted Tress Regression and it took 3.23 minutes to run but it gave a lower RMSE value than in Databricks at RMSE_GBT: 170.164

4.12 PySPark CLI on Oracle BDCE - Spark ML Decision Tree Regression

Then we ran Decision Tress Regression, and it took 20 seconds to run but it gave lower RMSE value than in Databricks of RMSE_DTR: 116.423

5. Conclusion

After running all the algorithms in Azura ML, Spark ML on Databricks CE and PySpark CLI on Oracle BDCE, by looking and comparing the RMSE from each algorithm, we conclude that the Linear Regression model performed better in Spark ML than in Azure ML. We had noticed that in Azure ML, after pruning features, the RMSE for Linear Regression had increased but in Spark ML, pruning features led to a lower RMSE value. In Azure ML, the model with Linear Regression performed best with lowest RMSE value and highest Coefficient of Determination. In Spark ML, the model with Decision Tree Regression performed best with lowest RMSE value as well as lowest run-time. Our work is interesting and important because by testing the different algorithms on AzureML, SparkML on Databricks CE and PySpark CLI on Oracle BDCE and comparing their RMSE and run-time, we are able to find out which model is has the best accuracy for predictive analysis with the lowest RMSE and the least run-time. This Predictive Analysis is important because it gives the State of Iowa greater knowledge and more accurate prediction on the future demand of various types of liquor to stock for wholesaling. By using the Permutaion Feature Importance and pruning features, most of the time we are able to increase the accuracy and decrease the RMSE of the model by excluding lesser imporant features from the model. The accuracy of the models are relatively compared to all the models created and tested in Table 1. Refer to Table 1. for RMSE and Runtime comparison on all the algorithms.

Table 1. RMSE and Run-Time Comparison

Azure ML		
Model	**RMSE**	**Time Taken**
Linear Regression	137.146	37.24 minutes
Boosted Decision Tree Regression	171.973	2.17 minutes
Decision Forest Regression	246.038	36.79 minutes
Spark ML – Databricks		
Model	**RMSE**	**Time Taken**
Linear Regression	141.968	4.07 minutes
Gradient Boosted Tree Regression	115.382	4.14 minutes
Decision Tree Regression	94.737	1.86 minutes
Spark ML – PySpark CLI on Oracle BDCE		
Model	**RMSE**	**Time Taken**
Linear Regression CrossValidator	182.177	4.15 minutes
Linear Regression TrainValidatorSplit	181.170	45 seconds
Gradient Boosted Tree Regression	170.164	3.23 minutes
Decision Tree Regression	116.423	20 seconds

References

[1] https://towardsdatascience.com/predictive-modeling-withiowa-state-liquor-sales-data-e45342081b83

[2] https://rakam.io/blog/analyzing-liquor-sales-iniowa/

[3] http://restudio-pubs-static.s3.amazonaws.com/392083_44450dc1463f4a788ed298c2c3b6538d.html#/

Dataset URL: https://data.iowa.gov/Sales Distribution/Iowa-Liquor-Sales/m3t-qhgy

GitHub URL:
https://github.com/bvkbvkbvk/bvk