Assignment 3: Logistic Regression with Spark

Vanessa Fotso

UMGC DATA 650

Spring 2021

giselef1@umbc.edu

Professor Ozan Ozcan

February 23, 2021

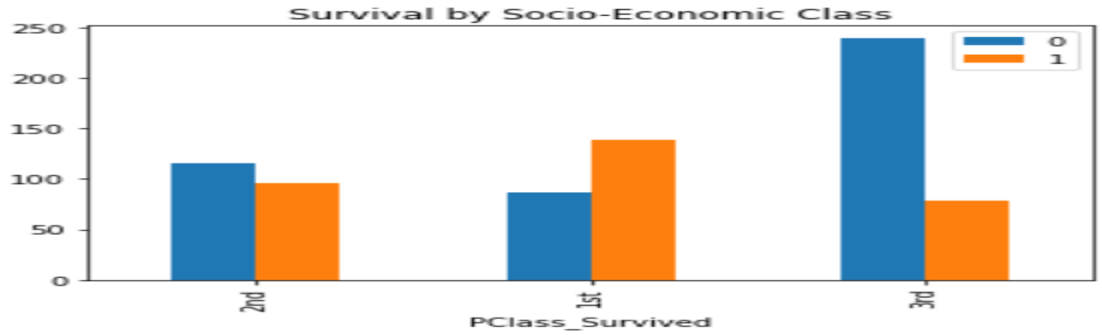
Part 1: Logistic Regression on Titanic Data

Discuss the importance of data exploration and visualization prior to running the logistic regression method.

Logistic Regression is a Machine Learning Classification technique used to estimate the likelihood of a class-dependent variable. The algorithm requires the output variable to be binary and to be determined only based on the meaningful input variables that are independent of each other. It is essential to explore and visualize the data prior building a logistic regression model to assert that the dataset meet the logistic regression assumptions. Data exploration and visualization help us to get familiarized with the dataset at hand by describing and summarizing the data, providing the size and data type, and uncovering hidden patterns. The exploration and visualizations also help analyze the relationship between the variables and derive hypothesis that set the direction of the analysis.

What do the Titanic data exploration results reveal about the relationships between the likelihood of survival and passenger data?

To determine the relationships between the likelihood of survival and the passenger data, the dataset was first queried for the number of passengers who survived and the number of passengers who died based on their Socio-Economic status. The result is shown below:



The above image shows that more passengers have survived (1) in the first class (about 61% of passengers), and that number has decreased from the first class to the third class (about 25%). Thus, the higher the social class, the more likely a person would have survived. Based on the exploration performed, no correlation between the age and the likelihood of survival was found.

Discuss the logistic regression method results, including the classification accuracy for training and test set.

A logistic regression model was built to predict the likelihood of titanic passengers to survive. The measures used to evaluate the model's performance include the accuracy, precision F1 scores, area under the ROC curve, and area under the PR curve. The accuracy represents the percentage of outcome the model can correctly classified; the precision is how often the model is right when predicting a class as true, in this analysis, this translates into the number of predicted survivors that are actual survivors; and recall will be the number of real survivors that the model correctly identified as survivors. All three measurements above (accuracy, precision, and recall) tell how well the model performs in classifying a given passenger's survival. The F score is derived from the precision and the recall. It put more weight on low scores, balancing the precision and recall, so that the algorithm has a high score (strong performance) only if the precision and recall are high. The ROC curve plots the recall against the rate of false positives, while the Precision Recall curve pots the precision against the recall and helps determine the threshold that provides the best precision/recall tradeoff for the analysis, as increasing the recall results in decreasing the precision and vice versa.

```
# This table shows:
# 1. The number of passengers who survived predicted as died
# 2. The number of passengers who survived predicted as survived
# 3. The number of passengers who died predicted as died
# 4. The number of passengers who died predicted as survived

prediction.crosstab('label', 'prediction').show()
```

```
+----------------+---+---+
|label_prediction|0.0|1.0|
+----------------+---+---+
|             1.0| 28| 40|
|             0.0| 67|  9|
+----------------+---+---+
```

Figure 1. Confusion Matrix for Test Dataset of Titanic Logistic Regression Model

The test dataset has a total of 144 passengers. The second column of the above confusion matrix (figure 1) represents the prediction survived passengers: 40 passengers were correctly classified as survived (true positives) and 9 were wrongly classified as survived (false positives). The first column represents the prediction of passengers likely to die: 28 passengers were wrongly classified as not survived (false negatives) and 67 where correctly classified as not-survived (true negatives). Thus, the model here accurately predicts whether a passenger will survive with 74.3% accuracy. This accuracy has significantly decreased by about 9% from the accuracy on the training set (82.2%). This observation suggests that the model may be overfitted. Plus, the model predicted the survival of 74% of actual survivors in the training set and 59% in the test set (recall), while it was able to correctly distinguish only survivors 80% of the time in the training set and 82% of the time in the test set. The big drop in recall between the training and the test set also supports the hypothesis of overfitted model. In addition, the recall shows that the model poorly identified all relevant cases (survivors).

Since F score favors model with similar recall and precision, it might not be a good metric here since the recall is relatively low.

Finally, the area under the PR curve and the ROC curve are approximately equal (0.75 vs. 0.73). These areas are both equidistant from the min area of 0.5 and the max area of 1.0. Based on the results found, several tweaks need to be applied to the derived model to optimize its performance.

Is logistic regression suitable for this problem? Why or Why not?

The results of the analysis did not provide desirable results. There was a significant discrepancy between the metrics of the model on the training and test dataset, leading to think the model might have been overfitted. The issue may be due to the size of the dataset at hand and the way it was partitioned. Logistic regression requires the dataset to be somewhat large to perform well. The Titanic dataset is 756 data points, which is quite small. As it is possible that a larger dataset fixes the problem of overfitting, additional analysis needs to be performed on a larger dataset and correlation between independent variables need to be investigated to see if the analysis meets the assumptions of logistic regression algorithm. Also, the dataset should be further explored and prepared to handle any instance of missing values, outliers or errors that may have impacted the results of the derived model.

What alternative machine learning methods could be suitable for this problem? Consider at least 2 alternative methods.

Other supervised machine learning algorithm such as Random Forest, Linear Support Vector Machine, Naive Bayes, or Decision Trees could have been used for this analysis. An analysis could be run with each of these methods and compared to see which method is the best for the analysis. A decision tree is simple supervised machine learning method that is easy to

implement and interpret. The main advantage here is that it does not require any preprocessing steps and has no assumption regarding the shape of the data. Thus, the algorithm is not affection by the size of the dataset, or the presence of outliers or missing values as observed with the logistic regression technique. Plus, there is no need to first identify the meaningful input variables before building the model as the algorithm searches for the best feature by applying weight while splitting each node. Random forest is composed of multiple decision trees linked together to provide a more stable and accurate classification. The main advantage of random forest is that it can be implemented for both regression and classification analysis. Unlike simple decision tree, random forest's ability to grow and associate trees optimize randomness in a model and look for meaningful input variables among a random subset of features.

Part 2: Logistic Regression on Low-Birth-Weight Data

Objective

Low birth weight is a significant worry among infants since it makes their bodies be more fragile, making it harder for them to eat, inhale, grow, keep up their internal heat level, or battle infections. The objective of this analysis was to assess components which sway the low birth weight in new-born babies and raise the awareness among the pregnant mothers. Low birth weight can be evaluated over the maternal, nourishing, and financial components, taking significant properties on Low birth weight as it can prompt many disorders observed in infants. A logistic regression model was used in this analysis to recognize the compelling factors in anticipating Low birth weight using a medical record of eligible mothers.

Dataset Description

The low-birth-weight dataset is composed of 189 observations representing child births at the Baystate Medical Centre, Springfield, Massachusetts during 1986 (Hosmer, Lemeshow, & Sturdivant, 2013). The dataset has 9 variables, with the variable of interest being the birth weight of newborn children, a binary variable describing the risk of children born with LBW (0 represents a normal birth weight of at least 2.5kg, and 1 indicates a low birth weight under 2.5kg). The predictor variables include the mothers' age (numeric), the race (categorical with 3 levels), the smoking status during pregnancy (categorical with 2 levels), history of premature labor (categorical), history of hypertension (categorical with 2 levels), the presence of uterine irritability in moms (categorical with 2 possible outcomes), and the history of physician visits during the first trimester. The ID variable serves as a unique identifier and is irrelevant to the study.

Data Exploration

The data exploration reveals that the dataset is heavily imbalanced, with about 31% of observations being infants born with low weight and 69% with normal weight.

LOW	count
1	59
0	130

Table1: Low Birth Weight Frequency Table

Additionally, while analyzing the mothers' race against the occurrence of low birth weight, it was found that 42% of black women had children born with low birth weight, while the observation was 24% for white and 37% for other races. This lead to think that black women are more at risk of having children with low birth weight.

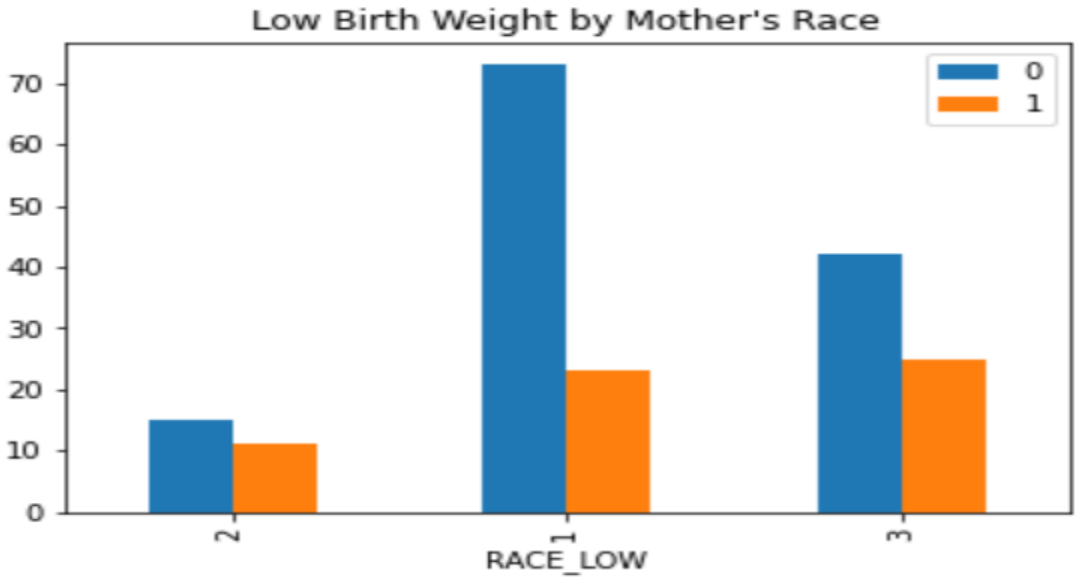


Figure 1: Low Birth Weight by Mothers' Race

Similarly, women who smoke during pregnancy tend to have higher risk compared to those who do not. The data shows 40% of women who smoked gave birth to children with low birth weight, while the percentage was considerably lower in non-smoking women (25%).

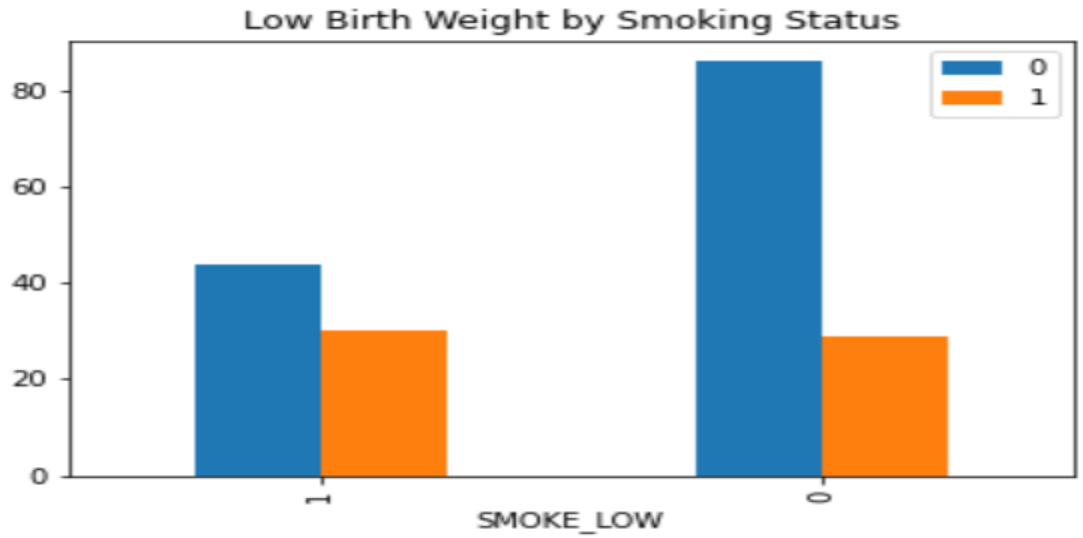


Figure 2: Low Birth Mothers' Weight by Smoking Status

Last, the age variable was found to be well distributed across the dataset, with the average age being about the same in both class (LOW and normal weight); however, the case of low birth weight was mostly observed in women between 20 and 25 years old.

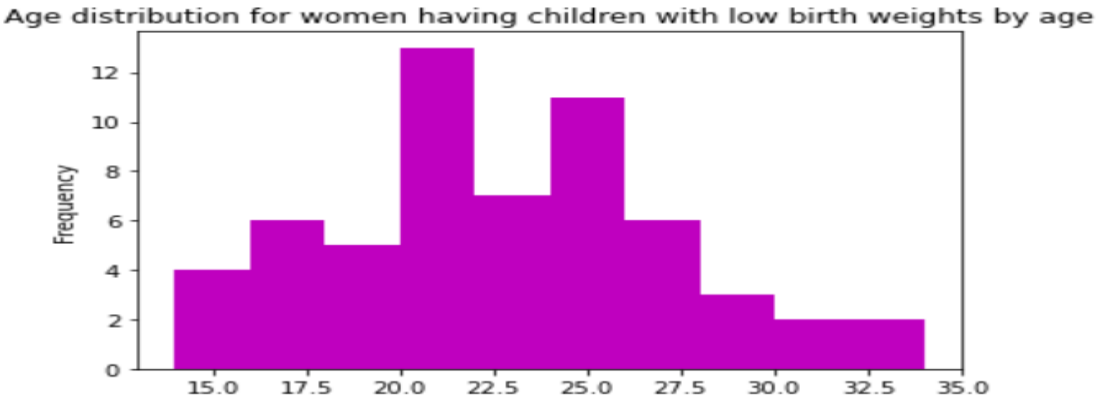


Figure 3: Age Distribution for Women having children with LOW.

Data Modeling

The dataset was not preprocessed prior building the model. Given the small size of the dataset, it was partitioned in 60% (108 cases) training and 40% (81 cases) test set. This was done so that the model would have enough blind data to test its efficiency. Additionally, a random seed

was applied to ensure that the analysis is reproducible. Finally, the model was built with a max iteration of 50 and the regParam parameter set to 0.001to optimize the results.

Results

The model can properly classify children with low birth with 76.1% on the training set and 67.9% on the test set. The accuracy has greatly decreased (by about 8%) between the two sets. This result suggest that the model has been overfitted. This may be since there is not enough data to train and test the model and the disproportion between the low-birth-weight cases and the normal weight cases. The precision as well has drastically dropped from 69.2% on the training set to 44.4% on the test set, and the recall going from 50% to 33.3%. This shows that the derived model poorly performed at identifying actual cases of low birth weight (< 2.5kg) in newborns. Likewise, the F score sharply decreased from 58% in the training set to 38% on the test set, as it is dependent on the recall and precision values. Furthermore, the model showed an area of 0.4 under the PR curve, which is due to weak recall and precisions values observed, and an area of 0.58 under the ROC curve, which is not far from the minimum value of 0.5, but distant from the maximum value of 1.0. This again confirmed the model failing to identify relevant cases in the data.

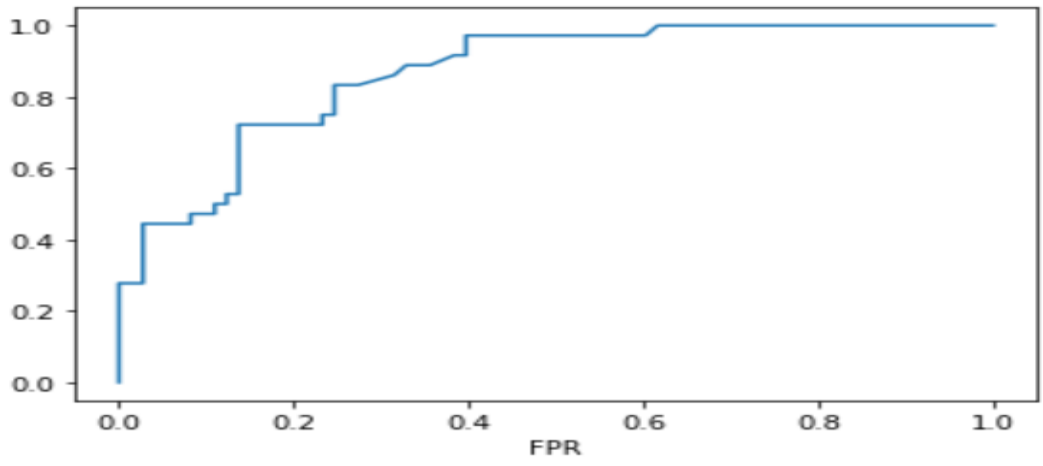


Figure 4: ROC Curve of Low Birth Dataset

Conclusion

A logistic regression model was built to predict the risk factors linked to low childbirth weight. Mother's race and smoking status were found to be strong predictors of the prevalence of low-birth-weight infants. However, the model built fell short with an accuracy of 67.9% and a F score of 38% on the test set. The results demonstrates that the model is not only unable to classify unknown data, but it can barely recognize true instances of low birth weight in infants. Logistic regression fell short here due to the dataset being considerably small and having a heavily unbalanced target variable (130 normal cases vs 39 LOW cases). Due to the small size of the dataset, increasing the number of cases in the test set might not solve the problem of overfitting. One can attempt to scale the target variable to balance its distribution.

Preprocessing the dataset before running the model may improve the model's accuracy. Logistic regression is affected by missing values and non-normal distribution. Plus, the input variables must be meaningful to the target variable to produce acceptable results. One way to evaluate the relationship of all variables to the target variable could be to run a correlation matrix. Cleaning the dataset and using only significant inputs could greatly improve the accuracy of this model. another limitation can be the broadness of some categorical variables (like simply used yes or no for smoking variable is not enough granular). Several other machine learning algorithms including random forest, decision tree, support vector machine or Naïve Bayes could be a good candidate to analyze this dataset. Decision trees can be suitable for this analysis as it can better handle small datasets by splitting data using boundaries parallel to the input axes in a recursive fashion until convergence is reached. Additionally, decision trees can handle unprocessed dataset by assigning weight at each node. Alternatively, Naïve Bayes can be used here as it requires less training time, which is appropriate for a small test set with relatively small input variables. The

Naïve Bayes algorithm converges quicker than other discriminative algorithm like logistic regression. Given the simplicity of the Naïve Bayes algorithm, the problem of overfitting can be avoided here.

Part 3 – Research questions

What is overfitting? What is the impact of overfitting on model performance? Discuss at least 2 approaches to avoid overfitting the model.

Overfitting alludes to the situation where a model cannot generalize or correctly classify unknown data. An obvious indicator of overfitting is when the classification error on the validation or test dataset is significantly greater than the classification error observed on the training dataset. This happens when the model is unable to recognize the underlying patterns in the data, but instead captures the background noise or random fluctuations in the dataset to the point that the accuracy is negatively affected when encountering unseen data.

Overfitting can be prevented by training the model on a considerable large and clean dataset as it can help the algorithm to better recognize the patterns and reduce the noise; however, this technique can be costly as it may require a continuous collection of data. Another approach in preventing overfitting could be using cross validation, a technique that assess the performance of a model on unseen dataset. This allows to tweak the model's hyperparameters for a better optimization.

Discuss 5 (five) key differences between HDFS and Object Storage.

HDFS and object storage are used for different functions when it comes to managing big data. HDFS primary function is to handle data analysis activities such as computing and editing data. It is intended to deal with the high transmission capacity designs that are needed for working on large information (Woodie, 2015). This component causes HDFS to adequately perform information science undertakings on extremely enormous datasets. Object storage, then again, is not equipped for big data computation tasks. Manipulating record and folders is a vital segment of

data science, yet object storage fundamental used is to store large unstructured data, including pictures and recordings. Erasure coding is used here to distribute the data across various disks, assembling spare drives to deal with instances of system failure to reconstruct the record (Woodie, 2015).

Object storage is significantly less exorbitant for long haul information stockpiling when contrasted with HDFS. This is on the grounds that HDFS stores three duplicates of each dataset to forestall the deficiency of information on account of a framework disappointment. For huge datasets, this can go through a ton of superfluous extra room, particularly since most of this information may not be utilized (Richardson, 2017). Another issue with utilizing HDFS for document storage is that there is a more prominent possibility of losing a portion of the information while saving petabytes of records (Woodie, 2015). The information is not as a rule continually reestablished—which implies that on the off chance that one of the three dataset duplicates loses a portion of its information, that duplicate will not revamp the lost information and the client will undoubtedly be ignorant that the information was even lost (Woodie, 2017). Another issue with the use of HDFS for long haul stockpiling identifies with the master node, which stores all metadata for a solitary cluster. As indicated by Richardson (2017), on the off chance that the master node bombs in HDFS, that whole cluster and the entirety of its metadata will no longer be available. Be that as it may, in object storage, the metadata can be moved anyplace in the framework and does not need to rely upon the master node (Richardson, 2017). Moreover, Richardson asserts that on the off chance that the master node falls flat in object storage, one of the slave hubs can without much of a stretch supplant it.

Another reason HDFS is not best for long haul document storage is that computation and storage activity are run simultaneously on the same node—implying that one cannot execute both

tasks independently (Richardson, 2017). Extending the measure of extra room will go through a ton of pointless processing power, while optimizing computations will in like manner require extra room. However, in object storage, extra processing power or storage capacity can be added onto separate hubs (Richardson, 2017). This is helpful since it forestalls squandering additional processing power or extra room when they are not required.

Concerning network latency, object storage utilizes quick organizations for moving information to the framework (Woodie, 2017). This makes it compelling for moving and storing huge volumes of data. In any case, dissecting large information for the most part requires applying the computations to the data as opposed to moving the data to the framework. Hadoop is enhanced for moving computations to the information source, which makes HDFS much quicker than object storage when performing analytics tasks. Hence, picking which of these technologies to utilize rely upon whether the objective is to store data or process it.

We may use R, Python, Scala, and Java programming languages with Spark. Discuss the pros and cons of each language.

Apache Spark provides four option of programming languages: R, Python, Java, and Scala. Each of these languages provides their own advantages and weaknesses that need to be assessed before choosing a preferred language to work on Spark. With respect to difficulty and effort needed to write codes, Scala is by a wide margin the hardest to learn. This is since its structure is more complexed compared to the other three programming languages. Additionally, Scala has more advanced features including macros and tuples that give improved execution yet are more difficult to understand. Scala was found to be 10 times quicker than his opponent python as far as processing and analyzing data. Additionally, since Scala is a static language, it has the advantage of being easy to maintain compare to dynamic languages like python. Scala also provide the Play

framework which is a very intuitive framework with the power to integrate diverse tools which is necessary to enhance functionalities when working with big data, thus, supporting real-time data streaming and processing.

Python is a simple dynamic and easy to understand language. It has simple syntax and standard libraries makes it a better choice for machine learning over Scala as the codes here are cleaner and easier to understand for new programmers. It also provides intuitive tools well suited for natural language processing compared to Scala. The challenge with dynamic languages like python is that the code can be hard to follow as it does not associate type to data. Not knowing the type of the data makes it difficult to know which functions to apply on the data.

Java is another statically type language which means it is also easy to maintain as it is an object-oriented language. This characteristic allows for code abstraction and polymorphism, allowing the code to be reusable. While Java requires more exertion to code due to its verbosity when contrasted with Python, it is yet simpler to use than Scala. As per Hicks (n.d.), one of the difficulties of dynamic languages is that it is more diligently to test for bugs or heartiness without running the code across a wide range of situations. Accordingly, while static languages are harder to learn, they might be quicker and less expensive to actualize. Java also provides a virtual machine which make it portable, and it is the foundation of the Hadoop ecosystem. Finally, another deal breaker is that Java does not support Read-Evaluate-Print-Loop (REPL) which is essential for big data processing.

R is the preferred language for statisticians as it is mainly used to build data models. R is dynamic and has a moderate learning curve and slow performance in spark as it needs conversion into Scala/Python before productizing. This is since after processing the data on worker nodes, the

data will need to be sent back to the driver where additional serialization will be required. Consequently, the processing time is increased, negatively impacting the performance.

In sum, since Spark is written in native JVM compatible Scala language, applications written in JVM compatible languages like Java and Scala do not require any explicit conversion to be executed on the virtual machine, making Scala and Java faster on Spark. On the other hand, since Python and R are not JVM based languages, applications written in these languages requires a middleman to talk to the JVM driver, thus increasing the performance time.

References

Hicks, M. (n.d.). Scala vs. Java: Why Should I Learn Scala? Retrieved April 8, 2018, from https://www.toptal.com/scala/why-should-i-learn-scala

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied Logistic Regression (3rd ed.). New York, NY: Wiley.

Richardson, M. A. (2017). 3 reasons to replace HDFS with Object Storage and 1 reason not to. Retrieved April 6, 2018, from https://it.toolbox.com/blogs/maryannrichardson/3-reasonsto-replace-hdfs-with-object-storage-and-1-reason-not-to-103117

Woodie, A. (2015, June 23). Data Lake Showdown: Object Store or HDFS? Retrieved April 6, 2018, from https://www.datanami.com/2015/06/23/data-lake-showdown-object-store-orhdfs/