**Northeastern University College of Professional Studies**

****

**ALY6110 Data Management & Big Data**

**CRN: 82268**

**Final Project - Predicting Customer Response in Banking**

**Presented By :**

Avinash Sunil Tripathi ([tripathi.av@northeastern.edu](mailto:tripathi.av@northeastern.edu))

Dishant Rajesh Modi ([modi.di@northeastern.edu](mailto:modi.di@northeastern.edu))

**Instructor :** Valeriy Shevchenko

**Date:** 06/21/2020

# Summary

Telemarketing advertising campaigns are a billion-dollar effort and one of the central uses of the machine learning model. The project is related to the direct marketing campaigns of a banking institution. The marketing campaigns were based on phone calls. The dataset contains around 1.029 million rows and is observed in Apache Spark, making it a case of Big Data. Often, more than one contact to the same client was required, in order to assess if the product (bank term deposit) would be ('yes') or not ('no') subscribed. In this project, we predicted Customer Response to Bank Direct Telemarketing Campaign Project in Apache Spark (Machine Learning) using Classification Model and Logistic Regression.

# 

# 

# Analysis

We have used a Banking Marketing campaign dataset for our project. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in-order to access if the product, bank term deposit would subscribe or not. The classification goal is to predict if the client will subscribe (yes/no) a term deposit (variable y). We examined the structure of the dataset and we see that the target variable “y” is a factor variable with 2 levels yes: will subscribe for term deposit; No: will not subscribe for term deposit.

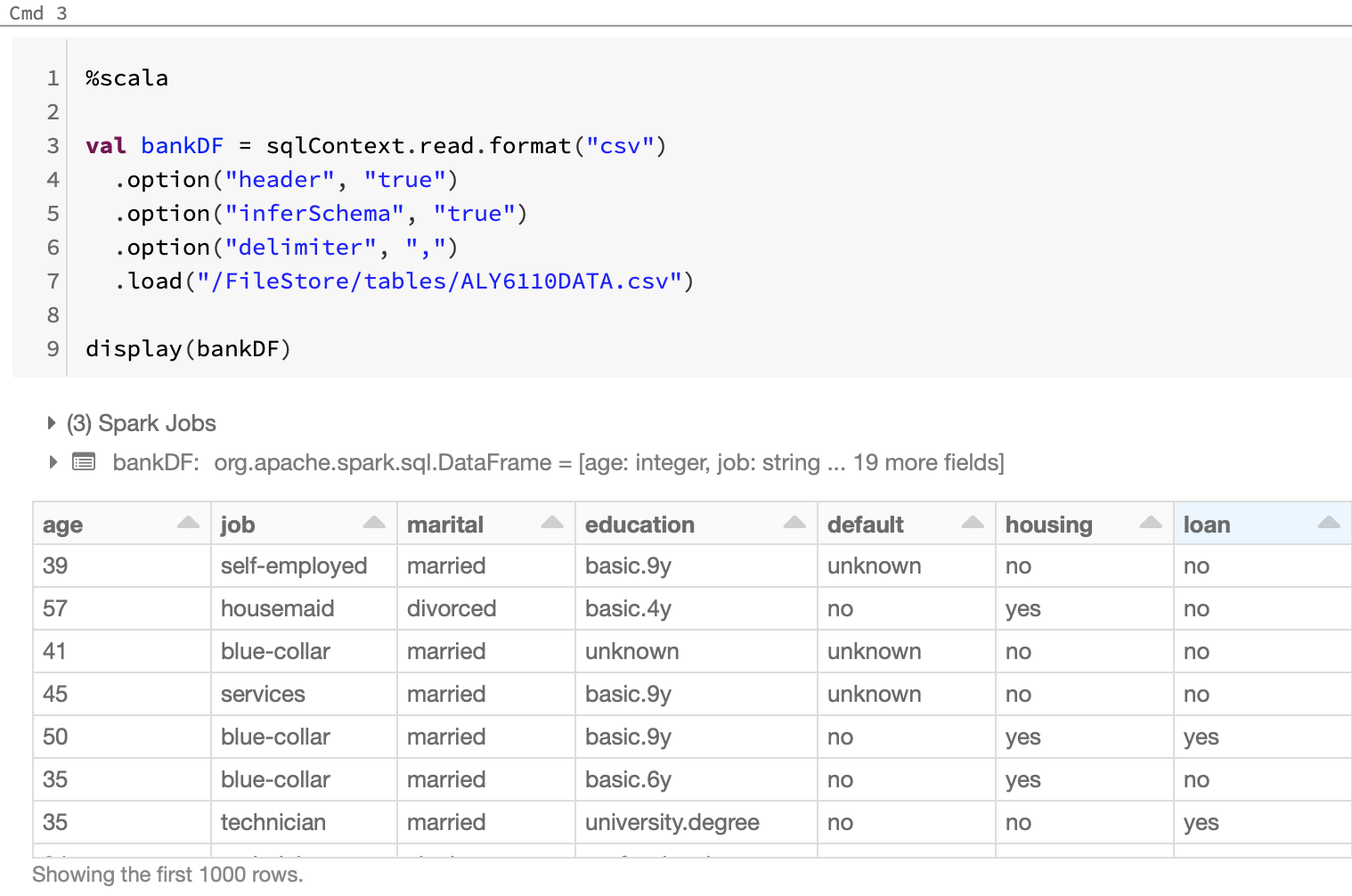
In line with the aforementioned, this study considered the typical case of bank direct marketing campaign dataset with main objectives. To predict customer response to bank direct marketing by applying two classifiers namely Decision Tree and Logistic Regression Model.

We used Community Edition Databricks to Run Apache Spark, and performed Analysis in Scala and SQL.

**Loading Source Data:**

The data for this Project is provided as a CSV file containing Customer details we need to predict if the customer will subscribe to a term deposit.

We will load this data into a DataFrame and display it.



* The data gets loaded into the **bankDf** Dataframe.

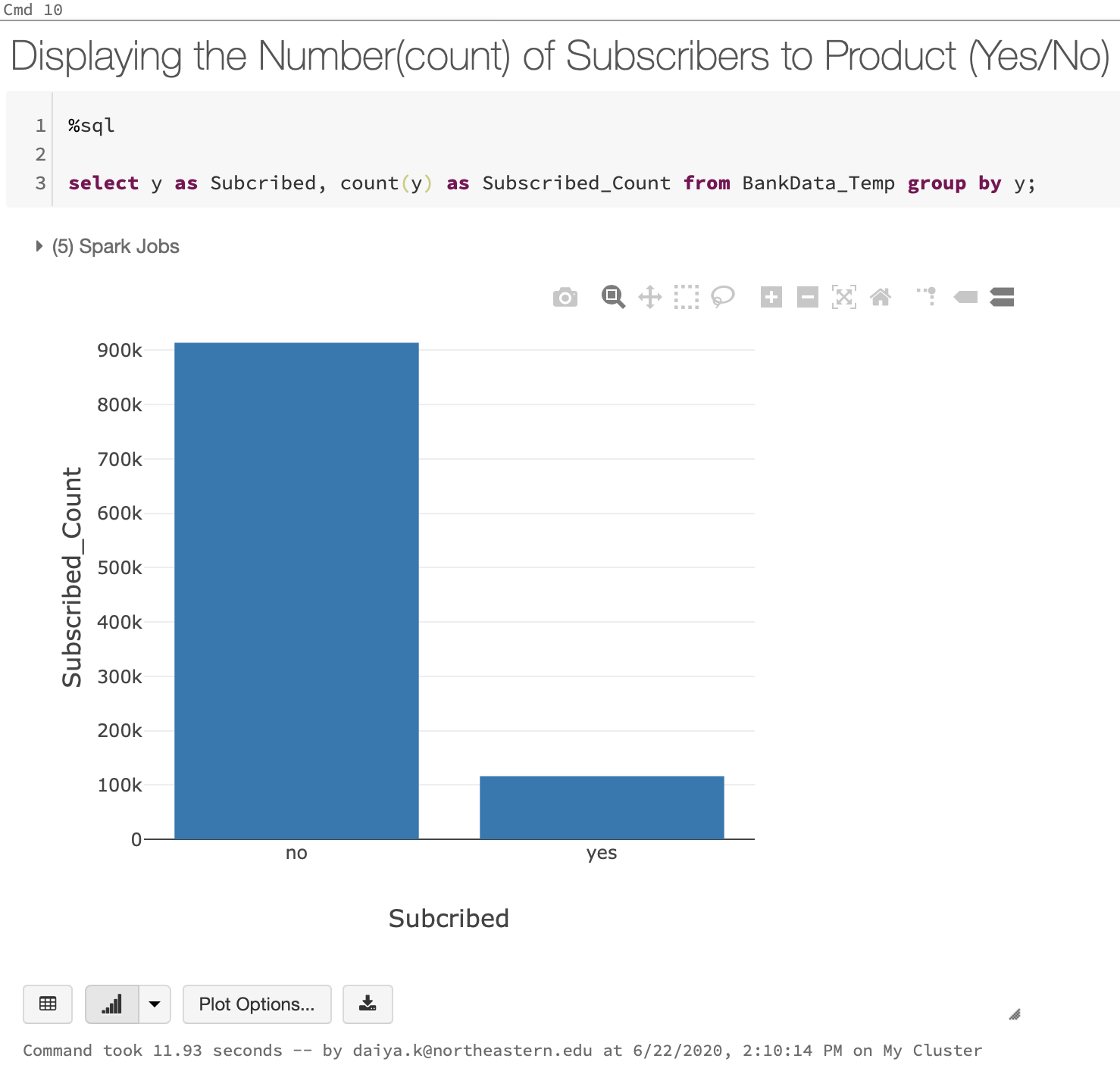
**(contd.)**

**Exploratory Data Analysis**

We performed EDA using SQL queries. Databricks seems to handle the Data very well and allows us to use Scala and Sql side-by-side.

1. **Distribution of Our Labels** : This is an important aspect that will be further discussed is dealing with imbalanced dataset. Knowing that we are dealing with an imbalanced dataset will help us determine what will be the best approach to implement our predictive model.

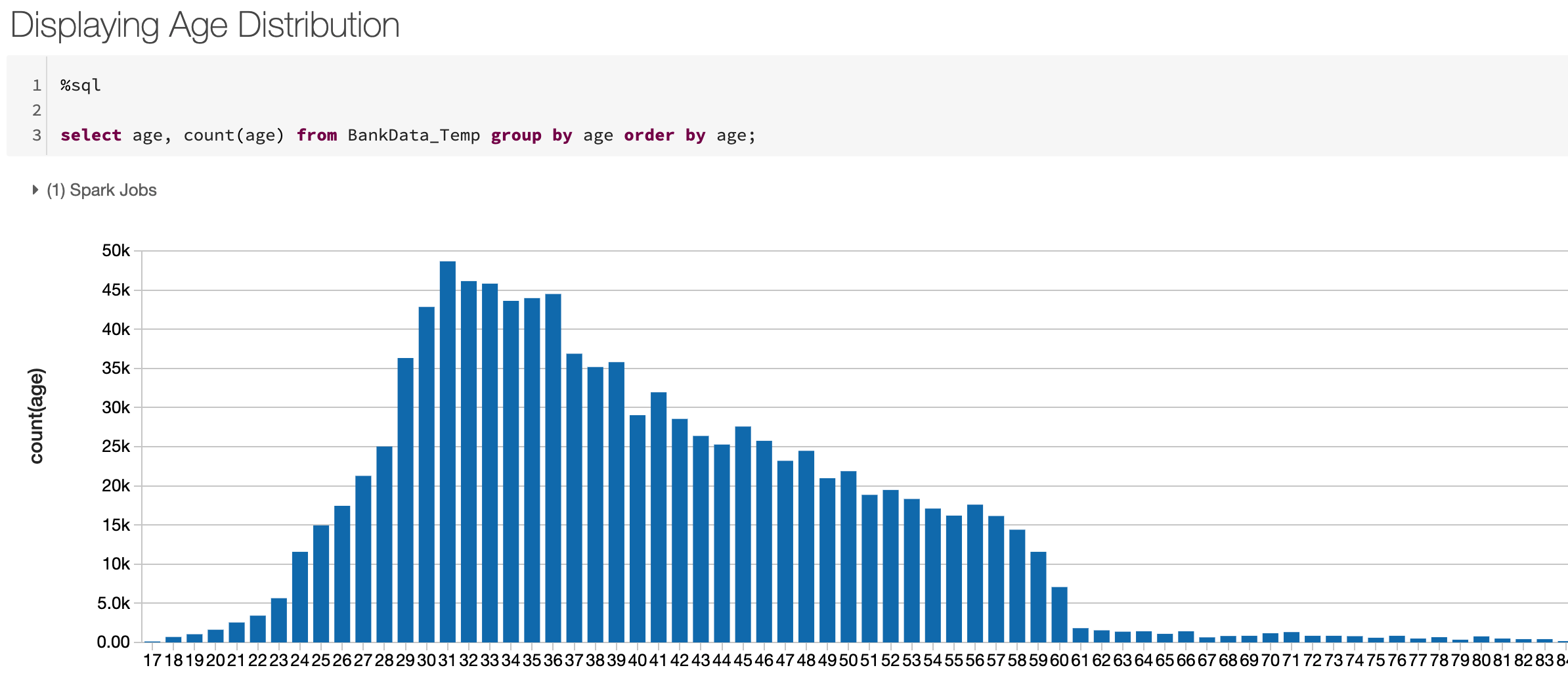
Total Number of Subscribers to the Product :



* Above output shows that our target variable is not equally distributed as there are around **89% ‘No’ and 11% ‘Yes’**

1. **Age distribution**: Questions to be answered :

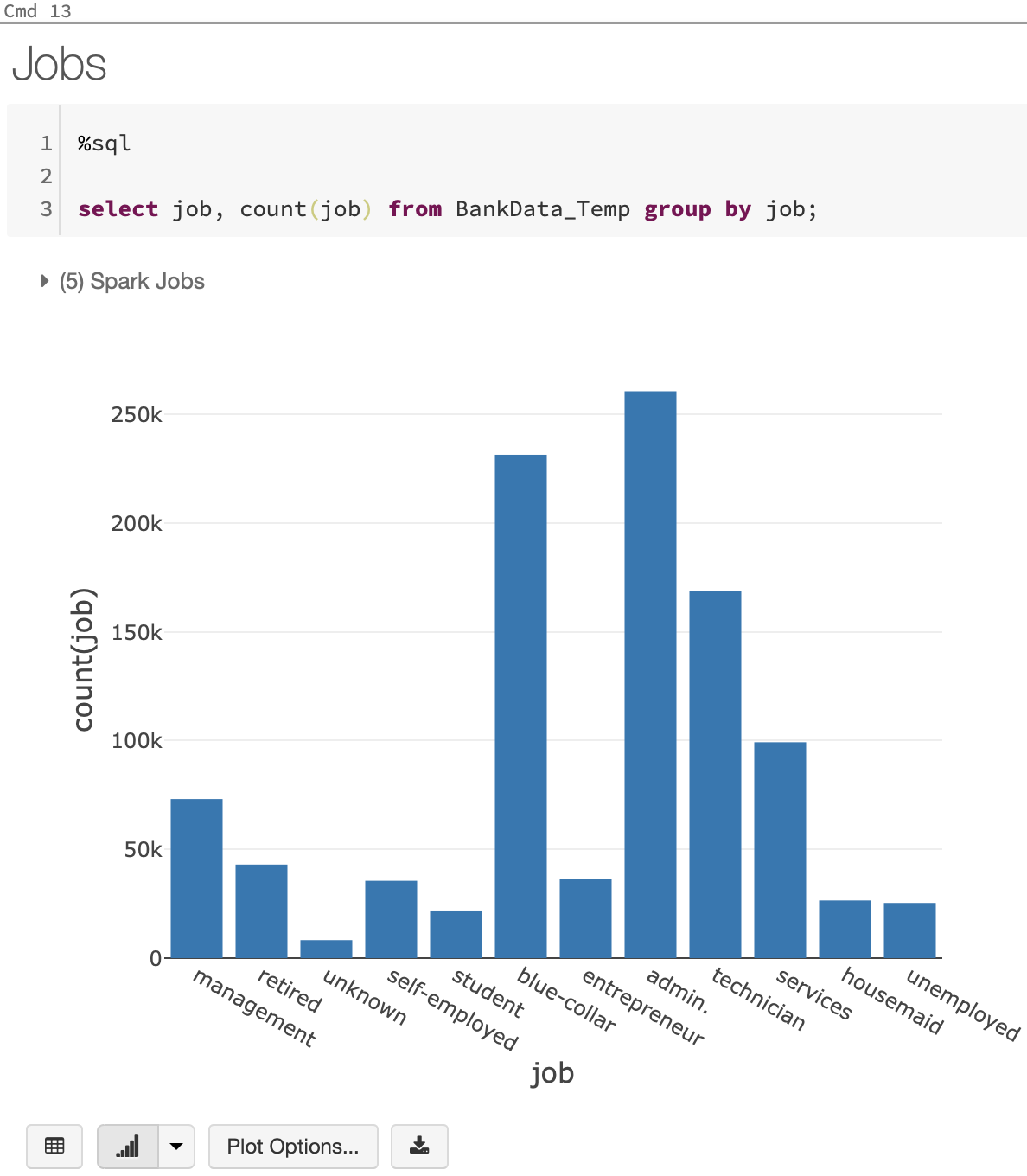
Let’s see how the age is distributed in a dataset?



The SQL query covers most people ranging between 20- 50 year of age.

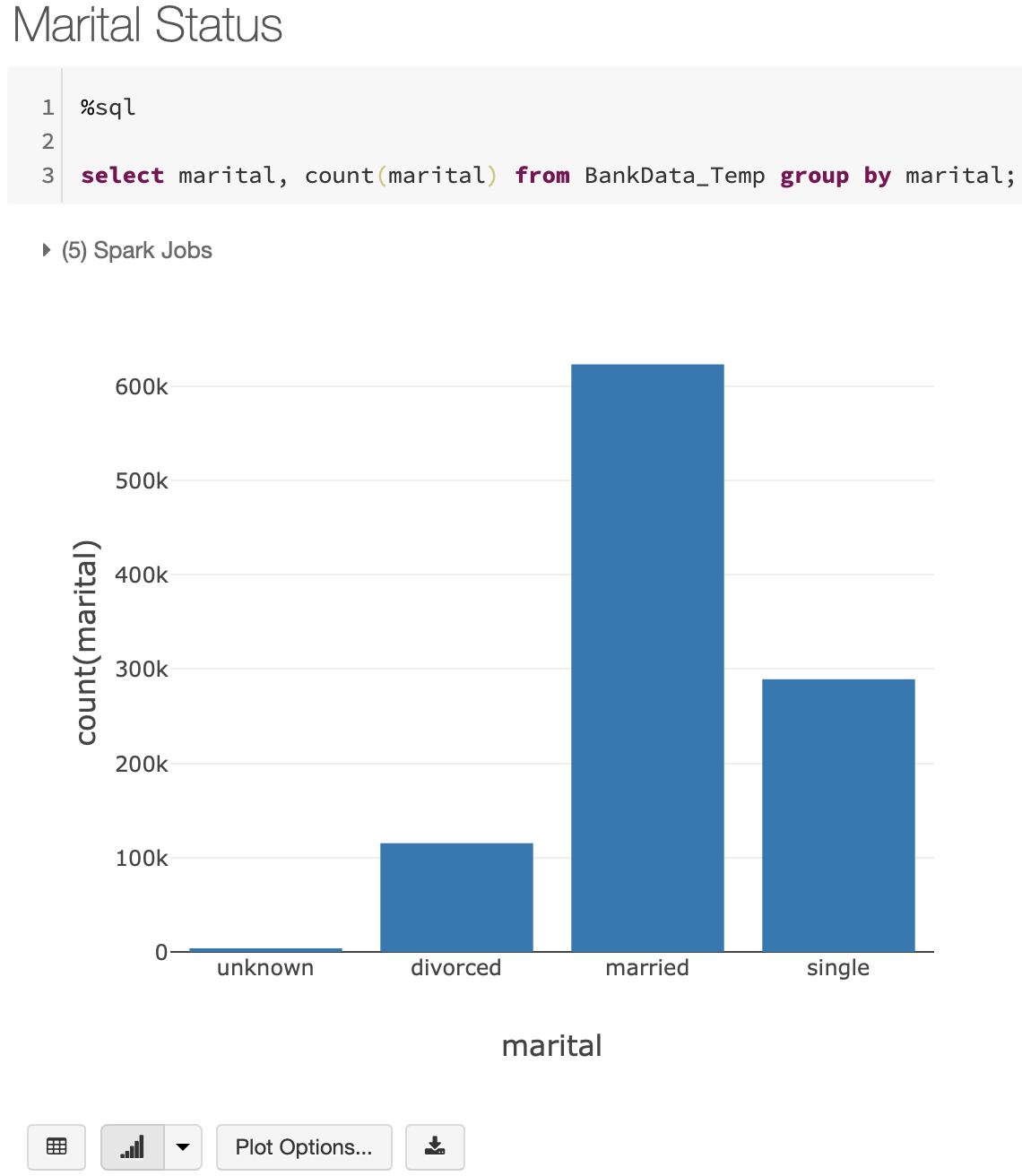
**Are there any significant discrepancies?**

1. **Jobs** : Shows the proportion of job types



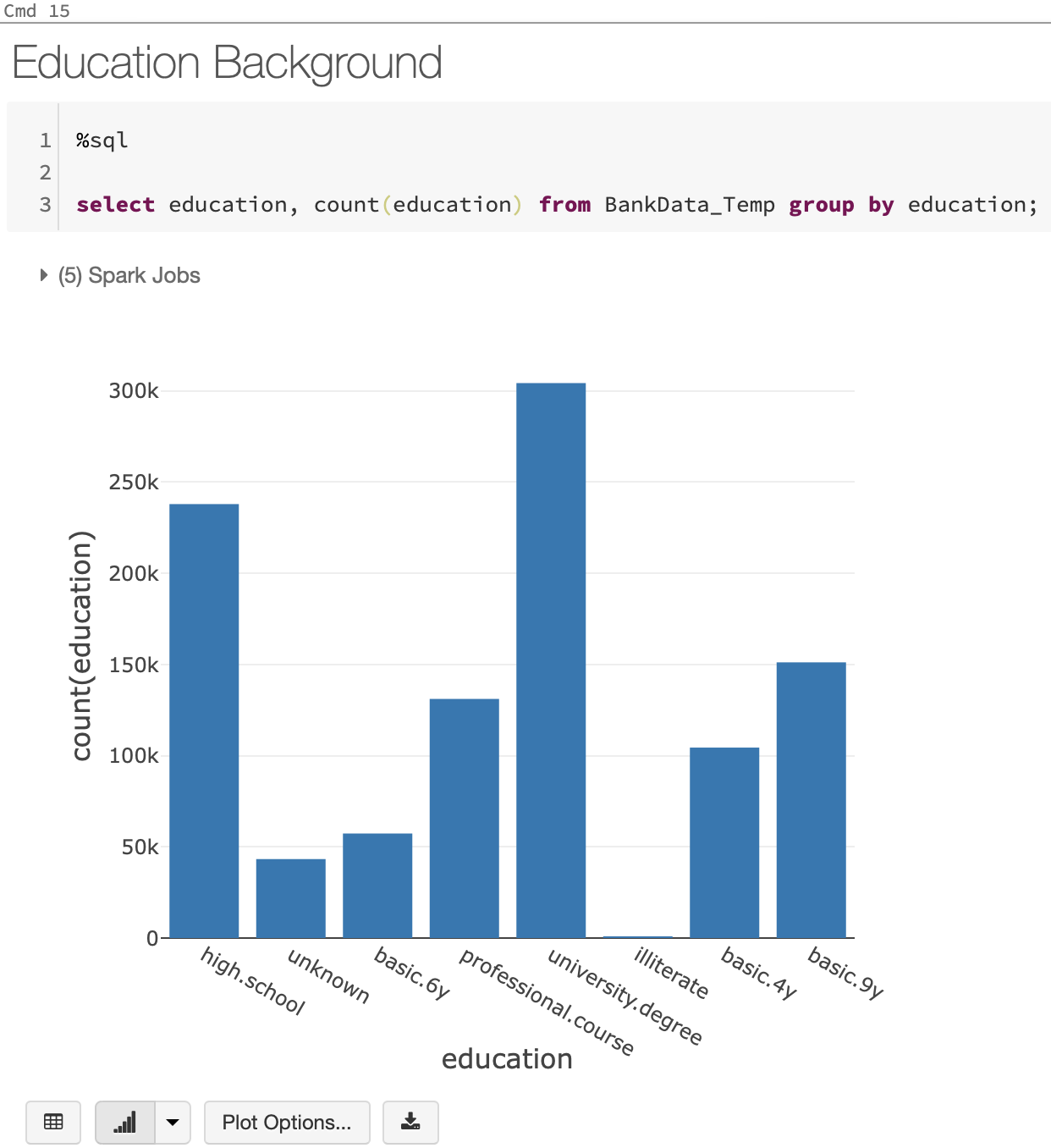
We can see there is proper distribution of all the Jobs.

1. **Marital Status :** If the person is Single, Married or Divorced

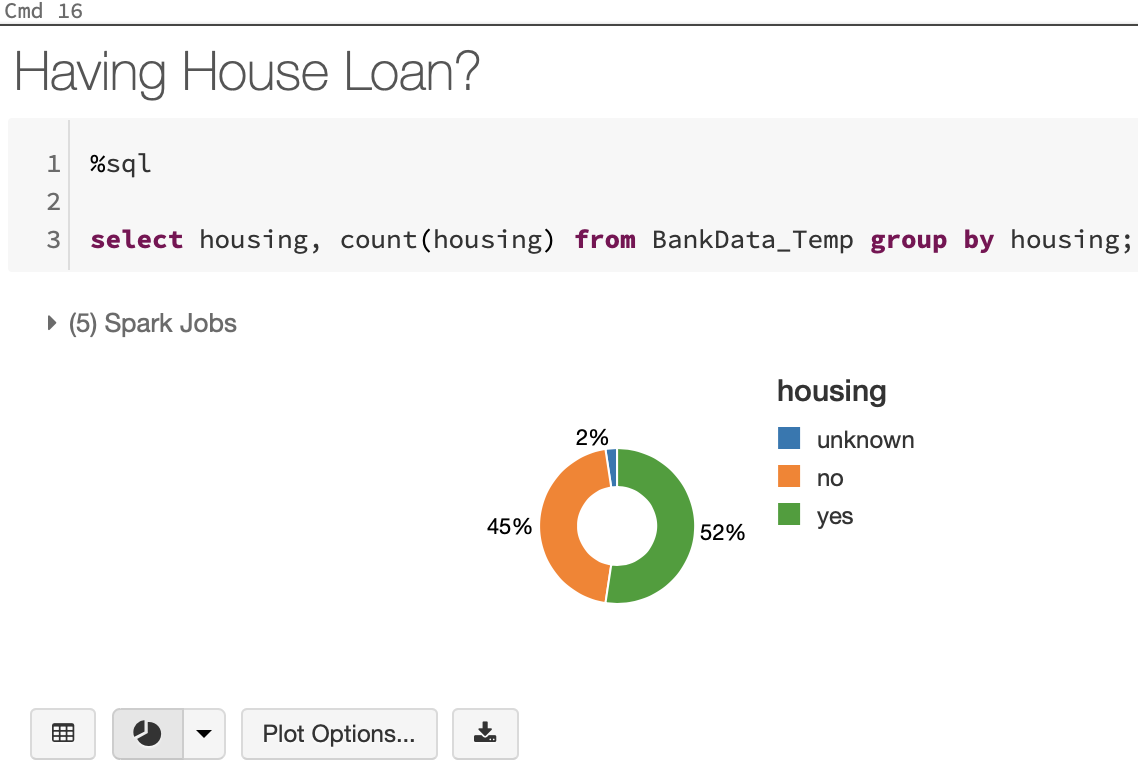


This SQL query will show the number of Single, Married and Divorced people in the dataset.

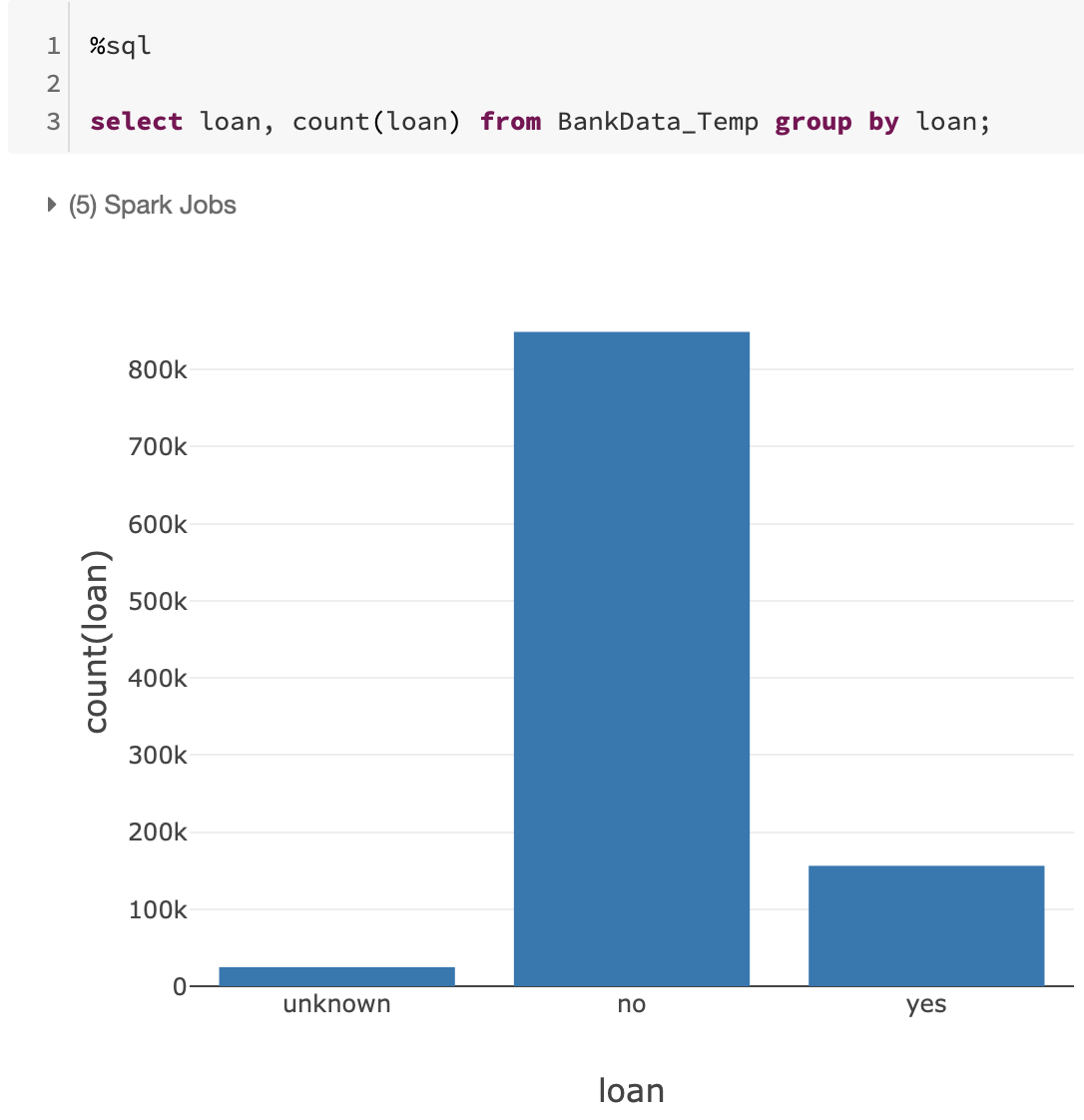
1. **Education Background** : Shows the proportion of Education Background of the people



1. **Having House Loan :** Shows number of people having active House Loan



**Having Personal Loan** : Number of People Having any Personal Loans



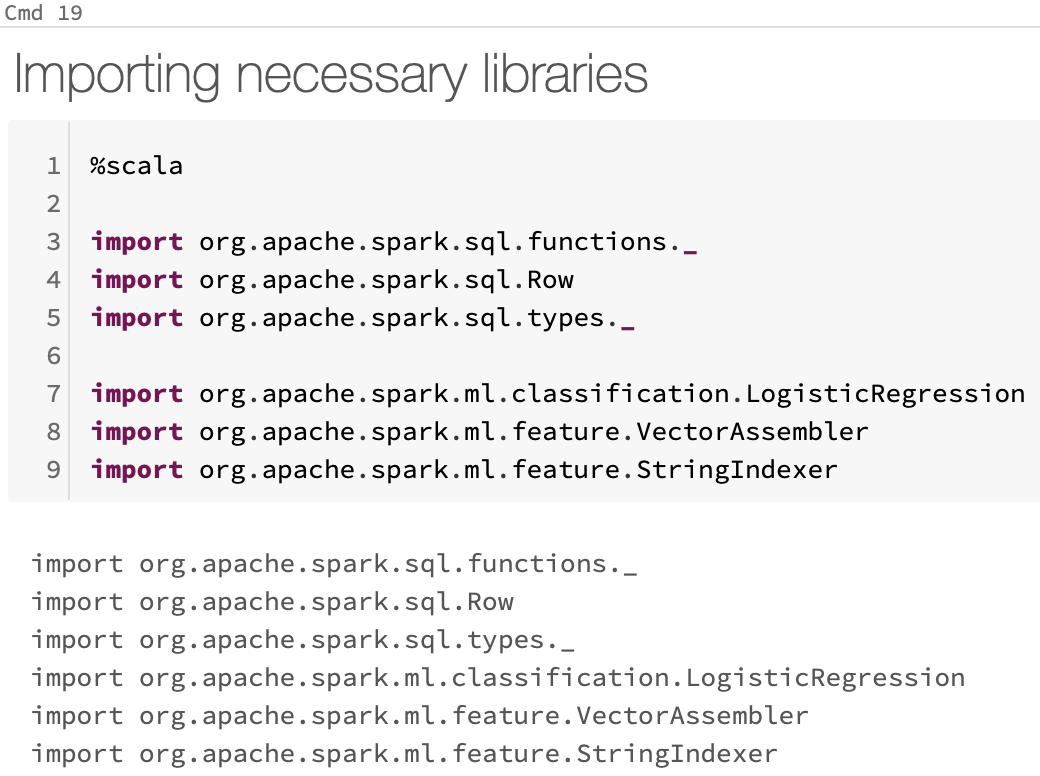
**Model Building and prediction**

We will be using logistic regression and decision trees to evaluate our test scores and finally, decide which model a banking telemarketing department can choose to predict the customer response.

**Creating a Regression Model**

Now we will implement a Regression Model that will predict the Customer Response based on the attributes available in Banking Telemarketing Data.

**Import Spark SQL and Spark ML Libraries**

****

Above lines of code will import necessary libraries for the operation of the Scala code.

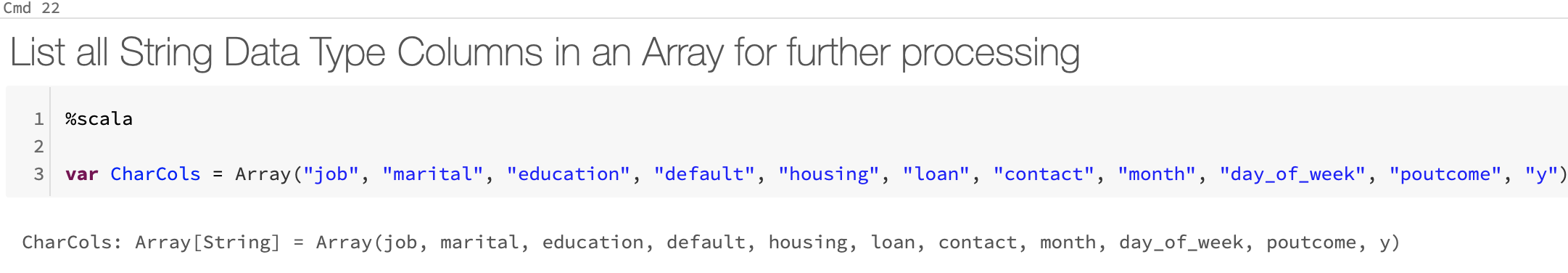
**Prepare the Training Data**

To train the regression model, we need a training data set that includes a vector of numeric features, and a label column. We used the VectorAssembler class to transform the feature columns into a vector, and then renamed the Y column to label.

VectorAssembler(): is a transformer that combines a given list of columns into a single vector column. It is useful for combining raw features and features generated by different feature transformers into a single feature vector, in order to train ML models like logistic regression and decision trees.

VectorAssembler accepts the following input column types: all numeric types, boolean type, and vector type. In each row, the values of the input columns will be concatenated into a vector in the specified order.

In each row, the values of the input columns will be concatenated into a vector in the specified order.



**(contd.)**

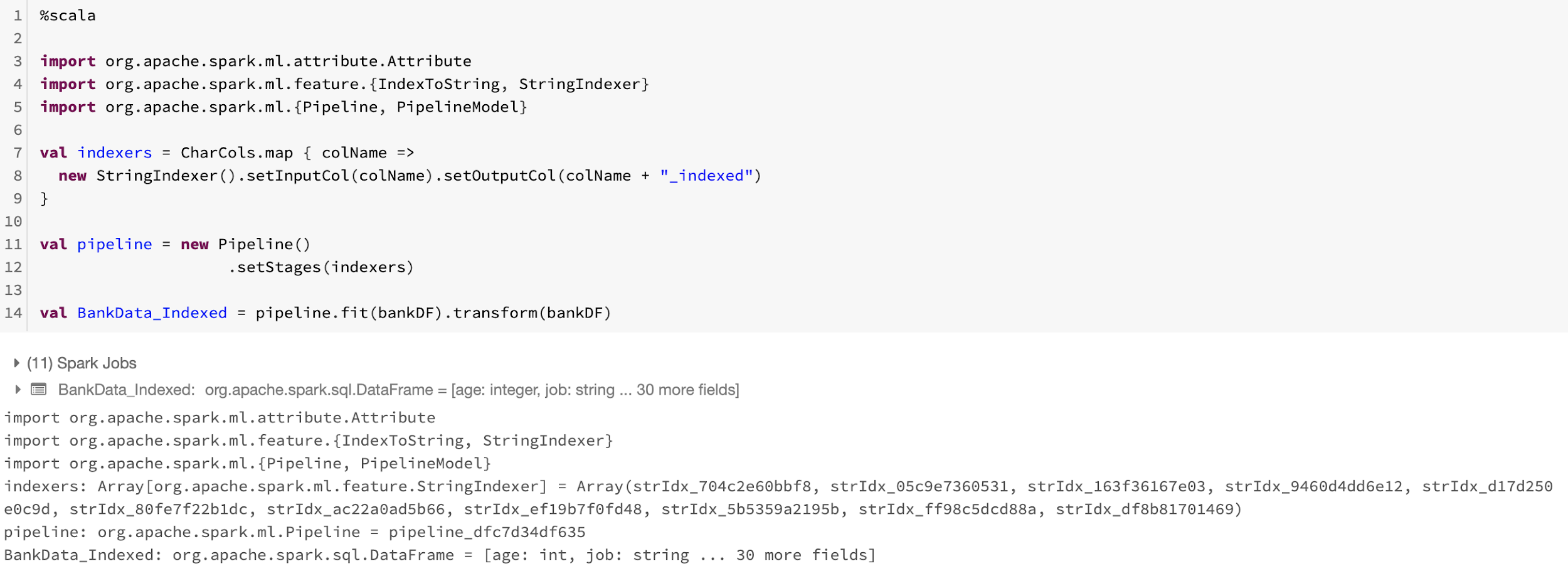
**Defining the Pipeline**

A predictive model often requires multiple stages of feature preparation.

A pipeline consists of a series of transformer and estimator stages that typically prepare a DataFrame for modeling and then train a predictive model.

In this case, we will create a pipeline with stages:

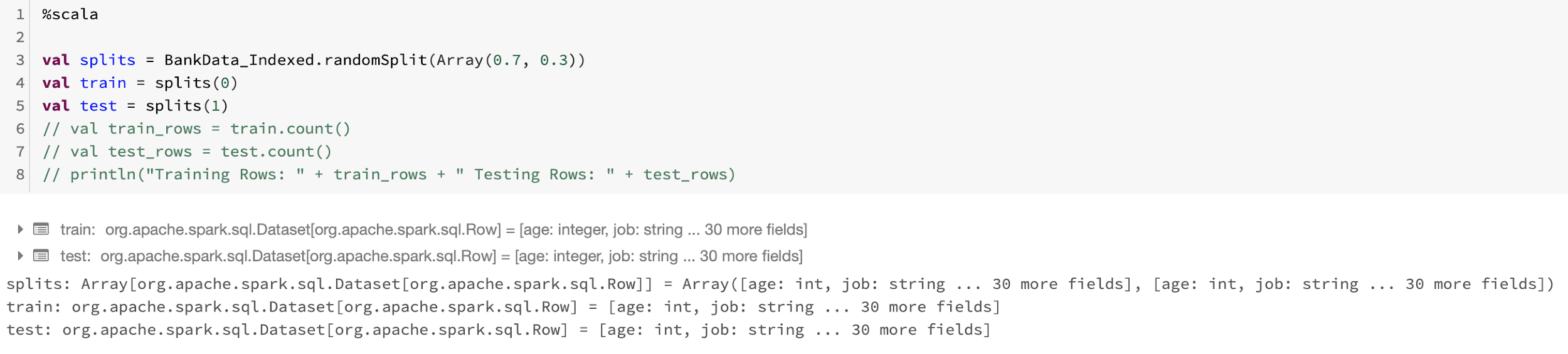
* A StringIndexer estimator that converts string values to indexes for categorical features
* A VectorAssembler that combines categorical features into a single vector



The libraries are now imported and the pipeline is ready to function.

**Split the Data**

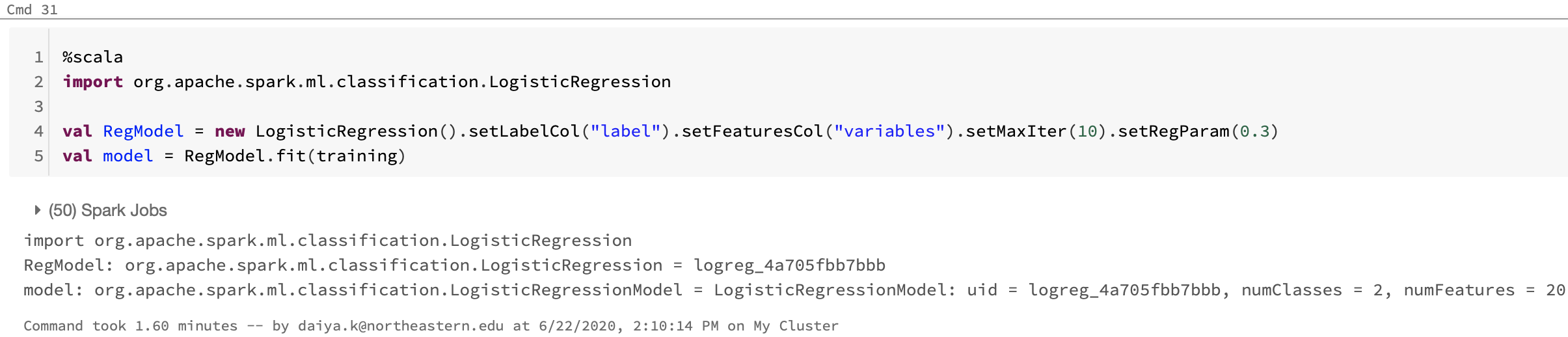
It is common practice when building supervised machine learning models to split the source data, using some of it to train the model and reserving some to test the trained model. In this project, you will use 70% of the data for training, and reserve 30% for testing. In the testing data, the label column is renamed to trueLabel so you can use it later to compare predicted labels with known actual values.



**Training a Regression Model**

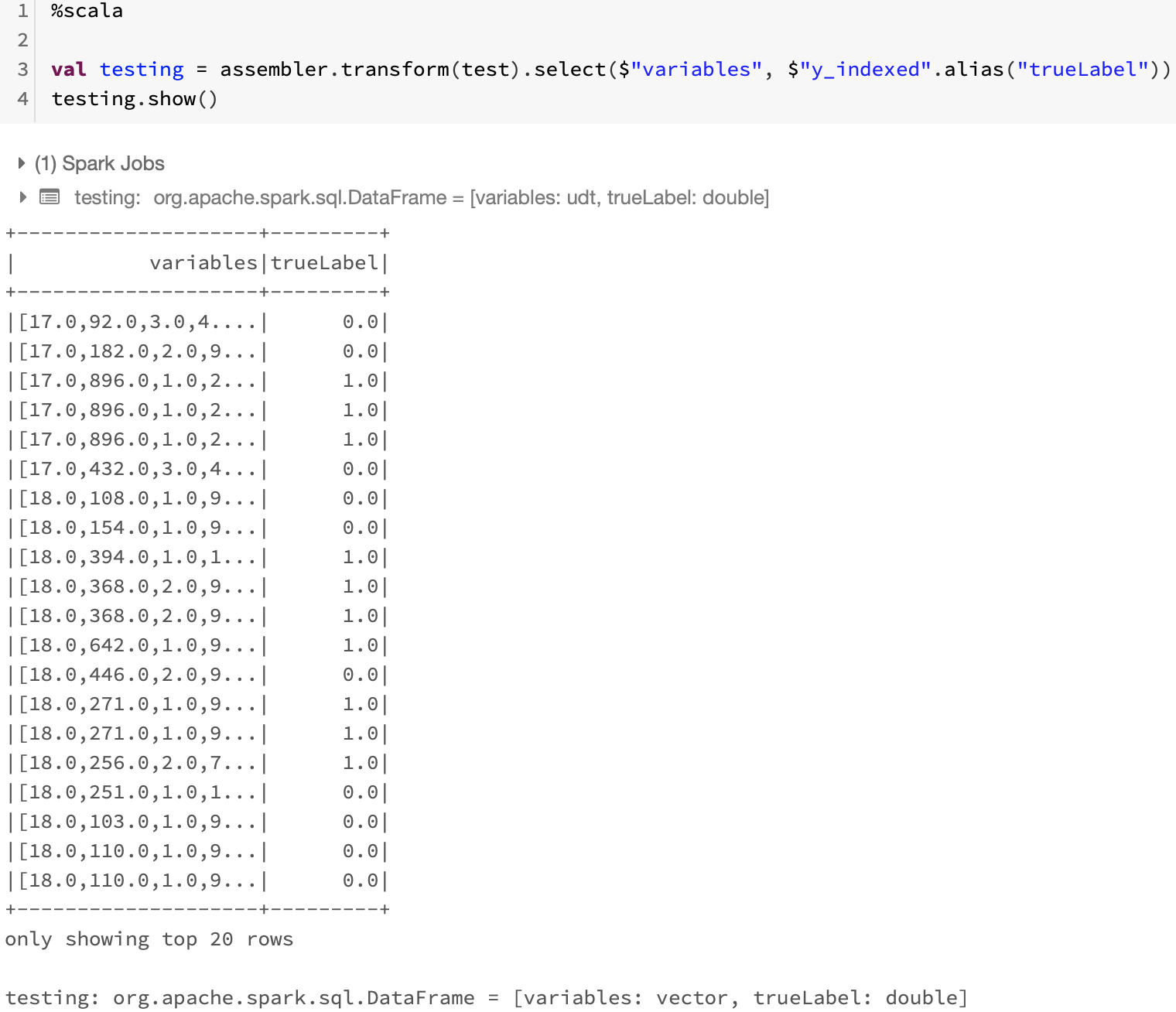
Next, we need to train a regression model using the training data. To do this, we created an instance of the regression algorithm that we wanted to use and used its fit method to train a model based on the training DataFrame. In this Project, we will use a Logistic Regression algorithm.

(contd.)



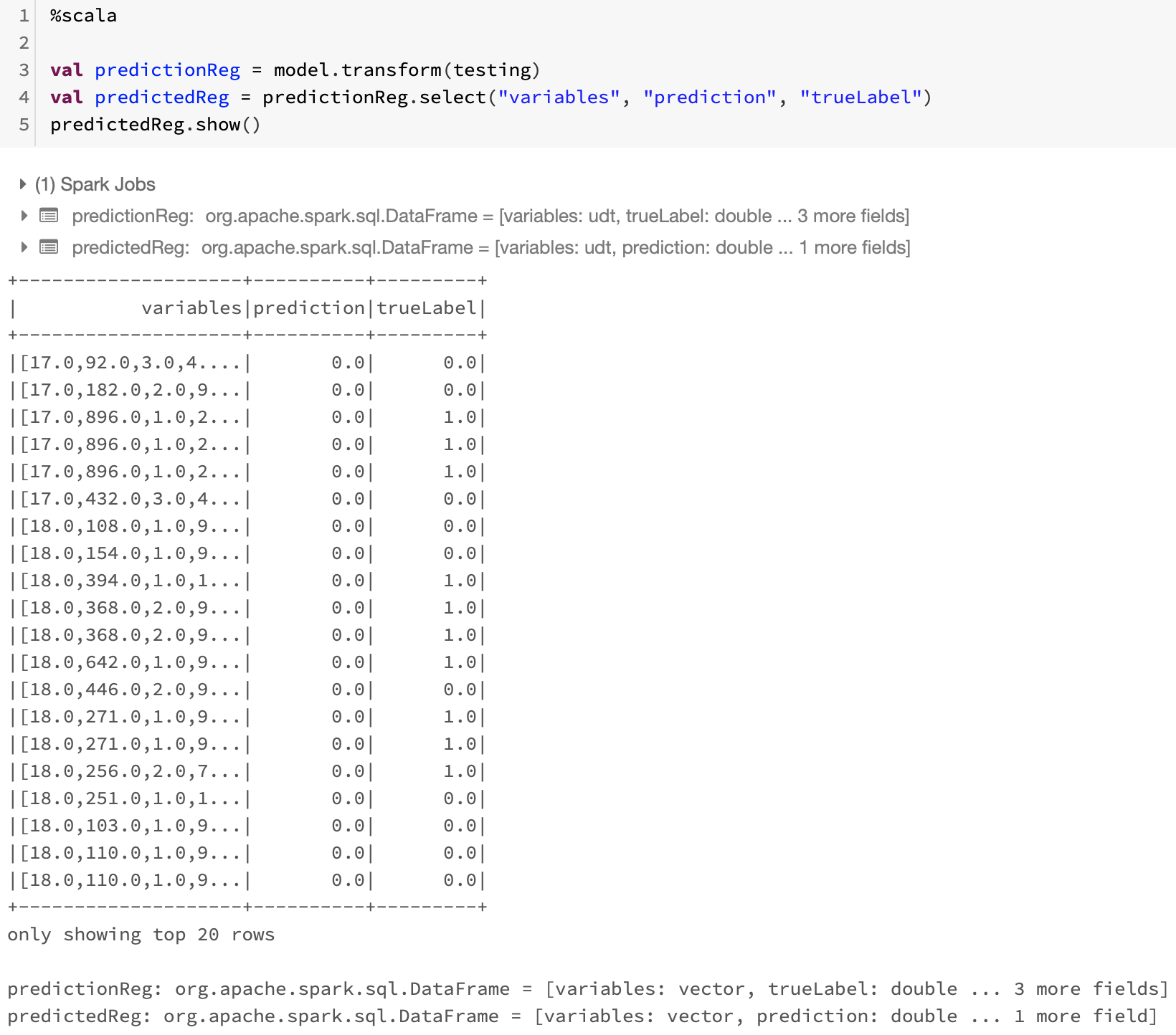
**Preparing the Test Data**

Now that we have a trained model, we can test it using the testing data we reserved previously. First, we need to prepare the testing data in the same way as we did the training data by transforming the feature columns into a vector. This time we'll rename the y\_indexed column to trueLabel.



**Test the Model**

Now you're ready to use the transform method of the model to generate some predictions. But in this case you are using the test data which includes a known true label value, so you can compare the predicted Revenue.



**Model Evaluation**

spark.mllib comes with a number of machine learning algorithms that can be used to learn from and make predictions on data. When these algorithms are applied to build machine learning models, there is a need to evaluate the performance of the model on some criteria, which depends on the application and its requirements. spark.mllib also provides a suite of metrics for the purpose of evaluating the performance of machine learning models.



We got the **AUC Accuracy of ~87%**

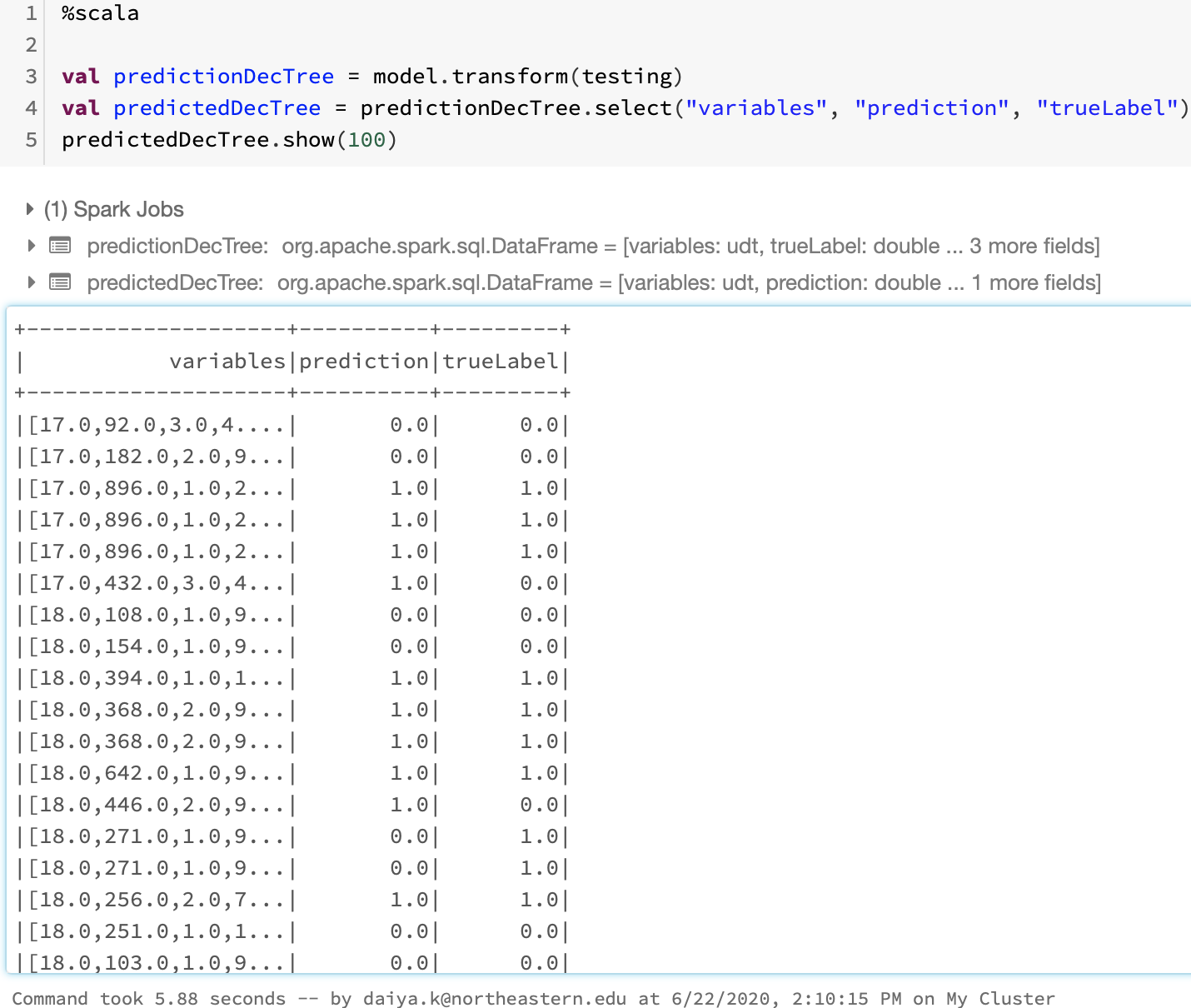
**Train a Classification Model (Decision tree classifier)**

Next, we need to train a Classification Model using the training data. To do this, we created an instance of the Decision tree classifier algorithm that we wanted to use and used its fit method to train a model based on the training DataFrame. In this Project, we used a Decision tree classifier algorithm.



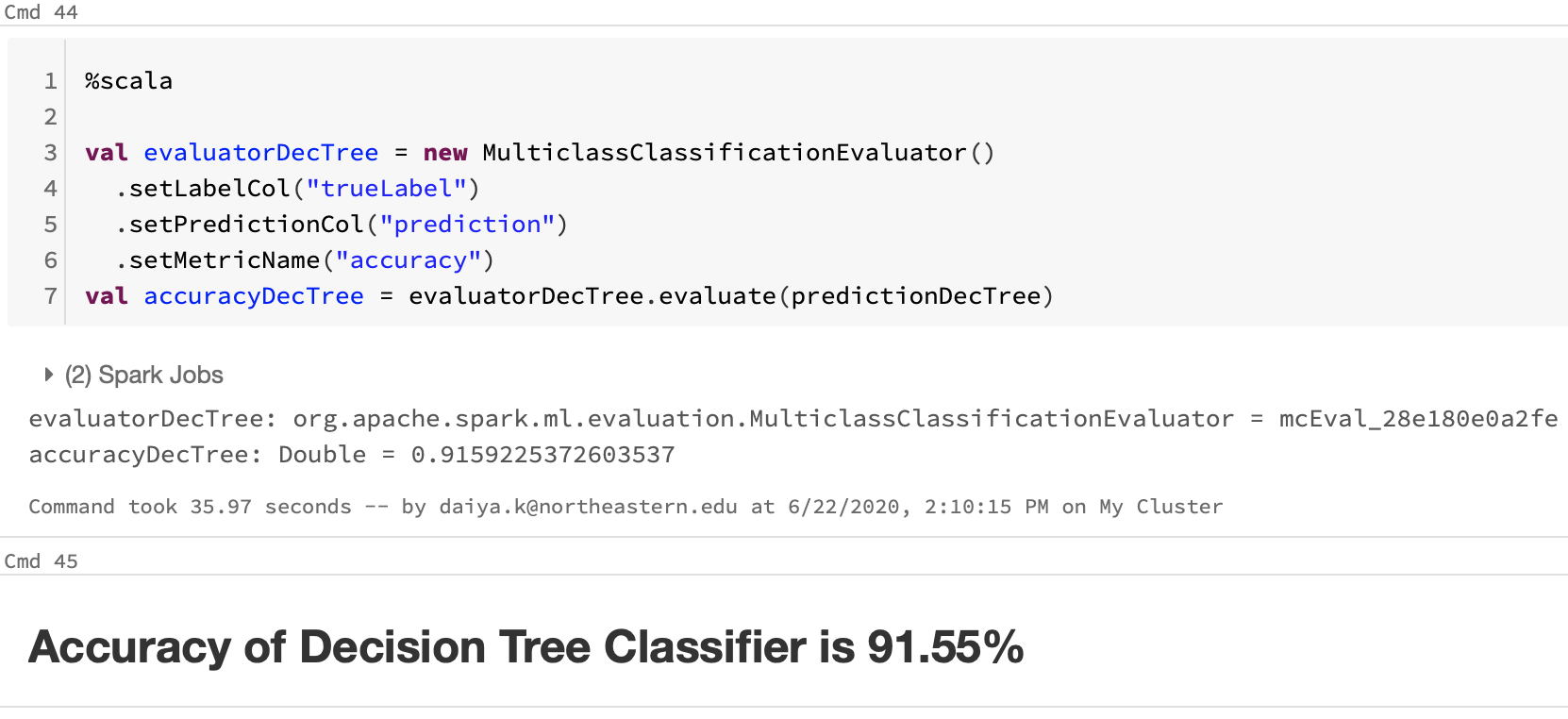
**Test the Model**

Now we're ready to use the transform method of the model to generate some predictions. But in this case we are using the test data which includes a known true label value, so we can compare the predicted Attrition.



**Classification Model Evaluation**

spark.mllib comes with a number of machine learning algorithms that can be used to learn from and make predictions on data. When these algorithms are applied to build machine learning models, there is a need to evaluate the performance of the model on some criteria, which depends on the application and its requirements. spark.mllib also provides a suite of metrics for the purpose of evaluating the performance of machine learning models.



**Additional comments**

1. spark.mllib comes with a number of machine learning algorithms that can be used to learn from and make predictions on data. When these algorithms are applied to build machine learning models, there is a need to evaluate the performance of the model on some criteria, which depends on the application and its requirements. spark.mllib also provides a suite of metrics for the purpose of evaluating the performance of machine learning models.
2. The only difficulty faced was during the creation of a pipeline for the data preprocessing. We have used string indexer and vector assembler for preprocessing:

* A StringIndexer estimator that converts string values to indexes for categorical features
* A VectorAssembler that combines categorical features into a single vector

1. As we were not familiar with scala so it took us some time to learn and complete the model evaluation.

# Conclusion

This model is simple and easy to implement. The bank marketing manager can identify the potential client by using the Decision Tree model if the client’s information like education, duration of the call, number of contacts performed during this campaign, Personal loan, previous outcomes, housing loan, etc is made available. This will assist the Bank while making critical decisions with regards to minimizing the cost to the bank by avoiding to call customers who are unlikely to subscribe to their packages. Here, we used Scala because it has reactive cores and a list of asynchronous libraries, testing is much better in scala because it is a statically typed language, the data types are decided by it during runtime in python and on other side, this is not the case in Scala that is why while dealing with large data process. All in all, Scala should be considered instead of Python.

# 

# 

# 

# 

# 

# 

# 

# 

# References

1. MLlib: Main Guide - Spark 3.0.0 Documentation. Spark.apache.org. (2020). Retrieved 20 June 2020, from <https://spark.apache.org/docs/latest/ml-guide.html>.
2. Introduction to DataFrames - Scala — Databricks Documentation. Docs.databricks.com. (2020). Retrieved 20 June 2020, from <https://docs.databricks.com/spark/latest/dataframes-datasets/introduction-to-dataframes-scala.html>
3. Apache Spark - Quick Guide - Tutorialspoint. Tutorialspoint.com. (2020). Retrieved 20 June 2020, from <https://www.tutorialspoint.com/apache_spark/apache_spark_quick_guide.htm>.
4. Valot, M., & Jorand, N. Scala programming projects (pp. 45-54).