# ITCS 5102 - SPL FINAL PROJECT

# CHURN PREDICTION USING ML MODELS

**Team Members :**

1. Aarti Nimhan - 801098198
2. Dedeepya Chintaparthi - 801056005
3. Ramanjaneyulu Reddy BL - 801101005

## 1. Project Description:

The main motive of our project is “Churn Prediction using Naive Bayes Classification Model and Linear Regression Model”. Churn Prediction is the process of analyzing customers who are more likely to unsubscribe their current services due to dissatisfaction. The Telecom Industry mostly uses this prediction to know their customers better. It finds the set of unsatisfied customers and tries to satisfy them with better offers, in order to stop them from leaving the network. This also helps to learn from satisfied customers so that the company knows its strengths as well.

The following “Telco Customer Churn” dataset from Kaggle is used in our project. https://www.kaggle.com/blastchar/telco-customer-churn/version/1.
This dataset contains 21 columns and 7043 rows. Each row in the data set represents a customer and every column holds the customer’s attributes described on the column metadata. In this project, ‘Churn’ column is chosen as the target/decision variable to predict if any customer churns or not.

### 1.1. Dataset Description:

- **customerID:** This attribute represents a unique ID assigned to each customer.
- **Gender:** This attribute tells if the customer is either Male /Female.
- **SeniorCitizen:** This attribute tells if the customer is a senior citizen or not (1,0). 1 indicates the customer is a senior citizen and 0 indicates that the customer is not a senior citizen.
- **Partner:** This attribute tells if the customer has a partner or not in terms of Yes/No.
- **Dependents:** This attribute tells if the customer has dependents or not in terms of Yes/No.
- **tenure:** This attribute represents the number of months the customer has stayed with the company.

- **PhoneService:** This attribute tells whether a customer has a phone service or not in terms of Yes/No.
- **MultipleLines:** This attribute tells whether the customer has multiple lines or not (Yes, No, No phone service).
- **InternetService:** This attribute represents the Customer’s internet service provider , whether the service is DSL/ Fiber optic/ No.
- **OnlineSecurity:** This attribute tells us whether the customer has online security or not (Yes, No, No internet service).
- **OnlineBackup:** This attribute tells us whether the customer has online backup or not (Yes, No, No internet service)
- **DeviceProtection:** This attribute tells us whether the customer has device protection or not (Yes, No, No internet service)
- **TechSupport:** This attribute tells us whether the customer has tech support or not (Yes, No, No internet service)
- **StreamingTV:** This attribute tells us whether the customer has streaming TV or not (Yes, No, No internet service)
- **StreamingMovies:** This attribute tells us whether the customer has streaming movies or not (Yes, No, No internet service).
- **Contract:** This attribute tells us about the contract term of the customer (Month-to-month, One year, Two year).
- **PaperlessBilling:** This attribute tells us whether the customer has paperless billing or not (Yes, No).
- **PaymentMethod:** This attribute tells us about the customer’s payment method (Electronic check, mailed check, Bank transfer (automatic), Credit card (automatic))
- **MonthlyCharges:** This attribute tells us about the amount charged to the customer on a monthly basis.
- **TotalCharges:** This attribute tells us about the total amount charged to the customer
- **Churn:** This is the decision attribute. It tells us whether the customer churned or not (Yes or No).

## 2. Why have we chosen only this language for our project?

Scala is an object oriented as well as functional programming language. Since Scala has static types it helps avoid bugs in complex applications. Scala interoperates seamlessly with Java. Best used over a wide range from machine learning to web applications. Applications working with streaming data, concurrency and distributed applications, analyzing data with Apache Spark. Netflix, Twitter, LinkedIn, AirBnB, AT&T, eBay, and even Apple. It's also used in the finance domain as well. Robust, Scalable, complete object oriented, and Type safety are the most important features of Scala.

Spark has a lot of higher-level libraries which include SQL query support, machine learning libraries, graph processing and more. These libraries can be smoothly combined to create a complex workflow. Spark also provides a lot of easy-to-use APIs for operating on larger datasets to transform data and also manipulate semi-structured data.

As to satisfy the requirements of the project the candidate languages we could use with Spark are Scala, Python, R, and Java. We shortlisted Python and Scala of which we chose Scala. Here are a few rationales for the choice: Scala is approximately 10 times faster than Python when it comes to Performance measures. Although the learning curve is more for Scala it supports powerful concurrency through primitives. Being a statically typed language, it provides a Type-Safety feature.

## 3. Features of the language :

1. Scala is a Functional and object-oriented programming language.
2. In Scala, many loops can be replaced by single words, making it less wordy than the standard Java, thus making the language concise.
3. Scala programming language provides a wide range of data structures like Arrays, Maps, Lists etc.
4. Apache Spark provides Scala, a Machine Learning library called MLlib with a goal to make practical Machine Learning easy. MLlib contains high-quality classification algorithms like Logistic regression, Naive Bayes etc.., Regression algorithms like Linear Regression. It also contains Clustering algorithms, Topic Modelling, Decision trees etc.
5. RDD is abbreviated to Resilient Distributed Dataset, which is an immutable distributed collection of records. Rdd is one of the fundamental data structures of Spark. Spark Rdd can perform two types of operations namely Transformations and Actions. Transformations include map, flat map, filter which are amongst the most commonly used ones. Transformation basically means modifying the data set to new from an existing one by passing the dataset through a function such that the function returns a new RDD representing the resultant data set.
6. Data frame is data organized in the form of named columns similar to a relational table. In order to select a particular column from the data frame one can use the methods apply method provided by Scala.
7. A vector assembler is used to merge or create an additional feature column containing information that we wish for the machine learning algorithm to consider. This transformer adds a single column to the Data frame called features.

## 4. How have we used these features in our project:

1. We have used the MLlib library from Spark in Scala to import the Naive Bayes Classification model, Linear Regression Model and Vectors.
2. We have used the sql library from Spark to import the Spark Session to create a new session.
3. We have used the concept of data frames to load the data from the csv file to spark data frame, where we can perform required transformations and actions on top of the data frame which contains the data.
4. We have converted the Data Frame to RDD to apply the data to the model as below

```
val rdd = dfFinal1.rdd
```

5. We have transformed the rdd by using map function of rdd, which takes a function as parameter as below

```
          val parsedData = rdd.map (
row =>
  LabeledPoint (
    row. getInt (2),
    Vectors.dense(
      row. getInt (0),
      row. getDouble (1), row. getInt (3), row.getInt(4), row.getInt(5), row.getInt(6)))
```

6. We have used Vector Assembler to merge multiple columns (the features selected) into a single vector column. It accepts boolean, numeric and vector type inputs.

## 5. Implementation of the Project:

### 5.1. Pre-processing the data:

After reading the data set, we have performed basic pre-processing on the data. In our project pre-processing includes the following:

- Finding the duplicate values and removing them.
- Finding the null values and replacing them with blanks

### 5.2 . Feature Selection:

For the feature selection, we have gone through all the features in the dataset and chosen the important features that would affect the predicted value (churn).

The following are the features considered to predict the value of ‘Churn’:

- SeniorCitizen
- Monthly Charges
- gender
- partner
- dependents
- contract

### 5.3. Splitting the Data:

We have split our dataset into training and testing sets with a proportion of 60% to training data and a proportion of 40% to the testing data.

### 5.4. Models Used:

We have used the Naive Bayes and Linear regression models to train our dataset to predict the value of Churn. The data is trained, and the accuracies of the models are printed.

### Naive Bayes Model:

- It is a classification model based on Bayes rule. It relies on a very simple representation called ‘Bag of Words’.
- The assumption of Naive Bayes Classifier is that there is strong independence between every attribute/feature of the data points.
- Simple Bayes or Independence Bayes are the other names for the Naive Bayes model.
- Some of the applications of Naive Bayes classification models are text classification, sentiment analysis etc.

### Linear Regression Model:

- Linear regression is used to determine the degree to which particular independent variables are influencing dependent variables.
- Here the dependent variable is continuous, the independent variable(s) can be continuous or discrete, and the nature of the regression line is linear.
- Linear Regression model establishes a relationship between dependent variable (Y) and one or more independent variables (X) using a best fit straight line. It is represented by an equation Y = a + b * X + e, where a is intercept, b is slope of the line and e is an error term.

- To find the equation of line to best fit the set of ordered pairs (x1,y1), (x2,y2), ...(xn, yn) we must first calculate the mean of all the x values and the mean of all the y values. Then we calculate slope. Using y-intercept and slope find the equation of the line for further predictions.

### 5.5. Output of the Models:

#### Naive Bayes Model:

Accuracy of the Naive Bayes model is 50.28%

#### Linear Regression Model:

Accuracy of the Linear Regression model is 60.35%

r2 value: 0.18754

Root Mean Squared Error: 0.39649

From the accuracies of the models, we can see that the Linear regression model gave the best results with an accuracy of 60.35% for our data set to predict the Churn value.

## 6. Step by step project execution instructions with screenshots.

1. Create HDFS directories by using below command:

#project root directory

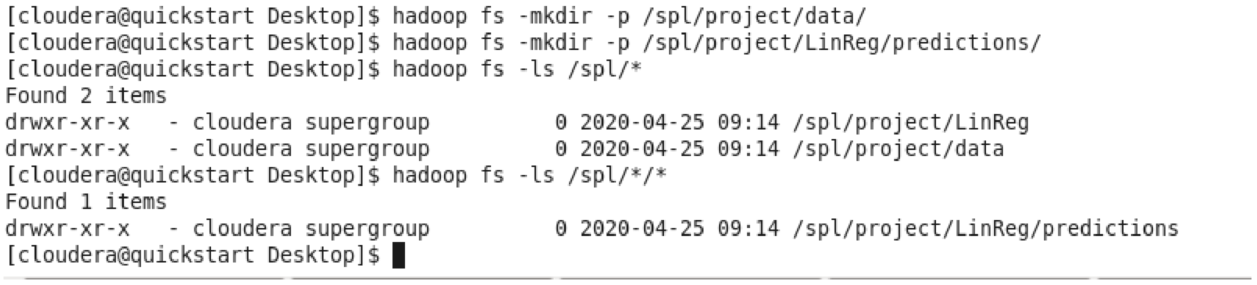
hadoop fs -mkdir -p /spl/project/

#for storing input files

hadoop fs -mkdir -p /spl/project/data

#for storing linear regression predictions

hadoop fs -mkdir -p /spl/project/LinReg/predictions/

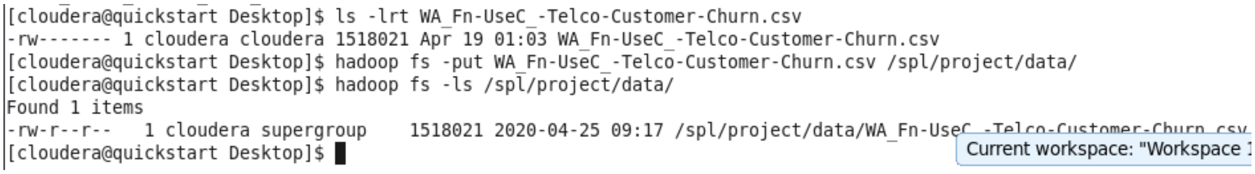


```
[cloudera@quickstart Desktop]$ hadoop fs -mkdir -p /spl/project/data/
[cloudera@quickstart Desktop]$ hadoop fs -mkdir -p /spl/project/LinReg/predictions/
[cloudera@quickstart Desktop]$ hadoop fs -ls /spl/*
Found 2 items
drwxr-xr-x   - cloudera supergroup          0 2020-04-25 09:14 /spl/project/LinReg
drwxr-xr-x   - cloudera supergroup          0 2020-04-25 09:14 /spl/project/data
[cloudera@quickstart Desktop]$ hadoop fs -ls /spl/*/*
Found 1 items
drwxr-xr-x   - cloudera supergroup          0 2020-04-25 09:14 /spl/project/LinReg/predictions
[cloudera@quickstart Desktop]$ 
```

All the folders required are successfully created in HDFS.

2. Copy the input file to HDFS location by using below command

hadoop fs -put WA_Fn-UseC_-Telco-Customer-Churn.csv /spl/project/data/

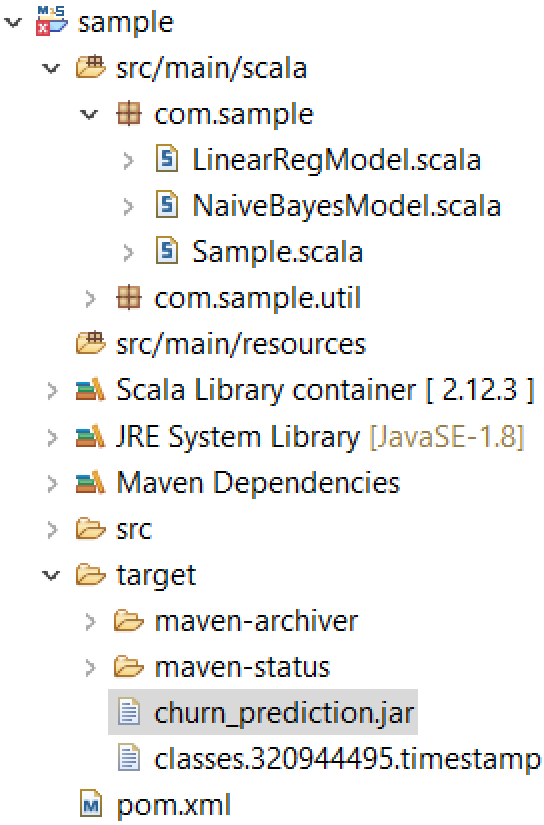


```
[cloudera@quickstart Desktop]$ ls -lrt WA_Fn-UseC_-Telco-Customer-Churn.csv
-rw------- 1 cloudera cloudera 1518021 Apr 19 01:03 WA_Fn-UseC_-Telco-Customer-Churn.csv
[cloudera@quickstart Desktop]$ hadoop fs -put WA_Fn-UseC_-Telco-Customer-Churn.csv /spl/project/data/
[cloudera@quickstart Desktop]$ hadoop fs -ls /spl/project/data/
Found 1 items
-rw-r--r--   1 cloudera supergroup    1518021 2020-04-25 09:17 /spl/project/data/WA_Fn-UseC_-Telco-Customer-Churn.csv
[cloudera@quickstart Desktop]$ 
```

Input file is successfully copied to the desired HDFS location and is listed (above figure).

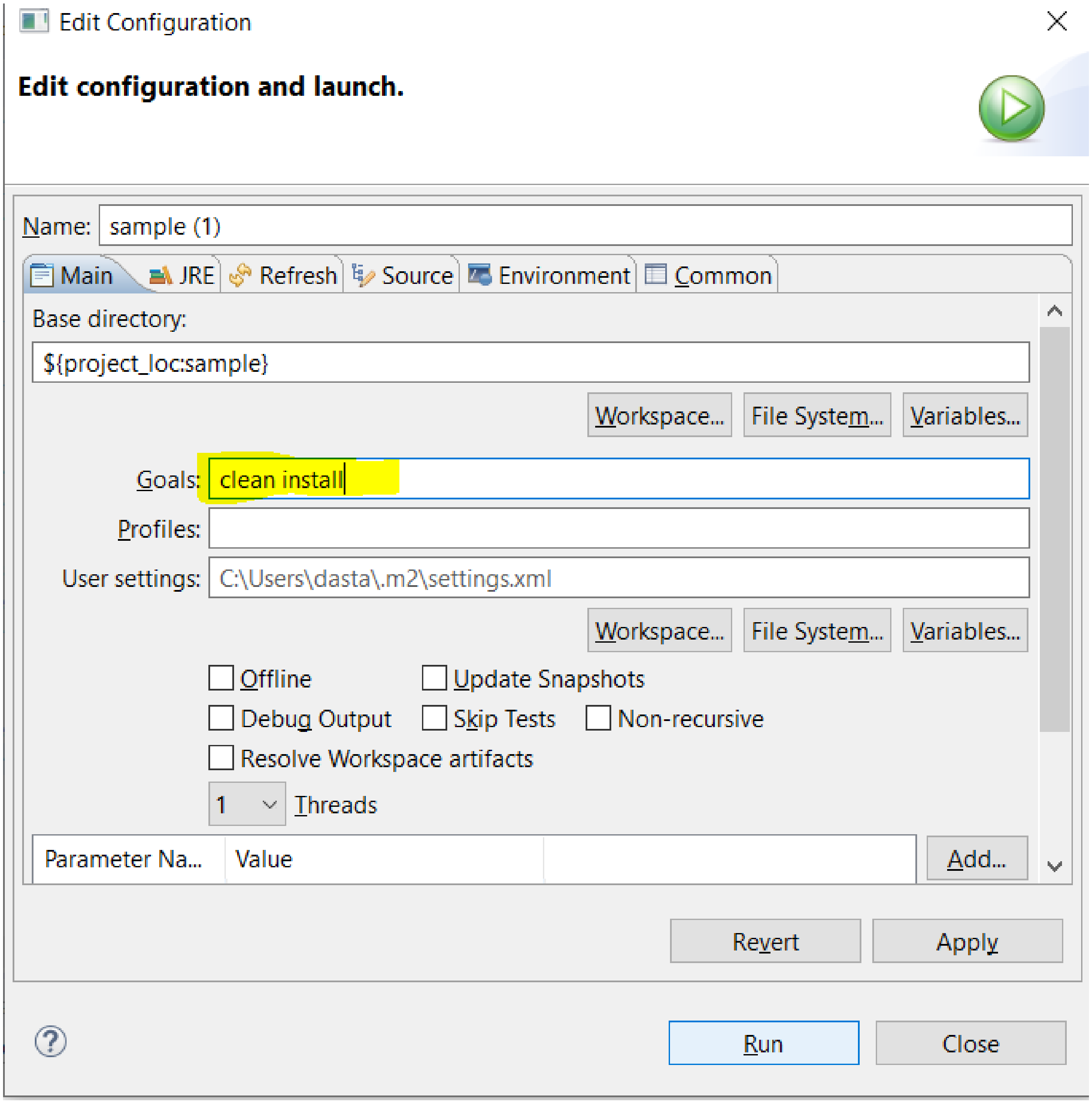
3. Build the jar file out of project code using maven “clean install”

a) project structure



```
sample
  src/main/scala
    com.sample
      LinearRegModel.scala
      NaiveBayesModel.scala
      Sample.scala
    com.sample.util
  src/main/resources
  Scala Library container [ 2.12.3 ]
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  src
  target
    maven-archiver
    maven-status
    churn_prediction.jar
    classes.320944495.timestamp
  pom.xml
```

b) Right click on project root, select run as, then select maven build. Then give clean install as instruction to maven build and click on Run.



Edit Configuration

Edit configuration and launch.

Name: sample (1)

Base directory:
${project_loc:sample}

Goals: clean install

Profiles:

User settings: C:\Users\dasta\.m2\settings.xml

□ Offline □ Update Snapshots

□ Debug Output □ Skip Tests □ Non-recursive

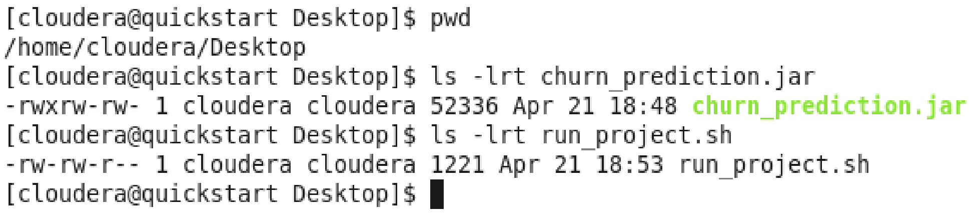
□ Resolve Workspace artifacts

1 Threads

| Parameter Na... | Value |
|---|---|

After this step, a jar file will be generated in the target folder, copy the file to cloudera virtual machine.

4. Copy the churn_prediction.jar file and run_project.sh files to current working directory



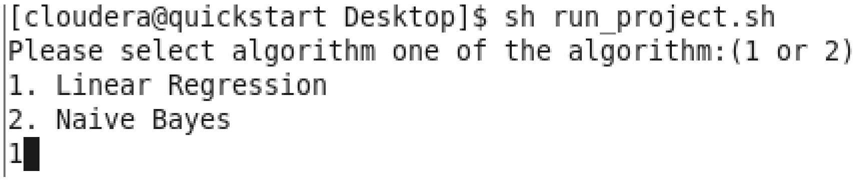
```
[cloudera@quickstart Desktop]$ pwd
/home/cloudera/Desktop
[cloudera@quickstart Desktop]$ ls -lrt churn_prediction.jar
-rwxrw-rw- 1 cloudera cloudera 52336 Apr 21 18:48 churn_prediction.jar
[cloudera@quickstart Desktop]$ ls -lrt run_project.sh
-rw-rw-r-- 1 cloudera cloudera 1221 Apr 21 18:53 run_project.sh
[cloudera@quickstart Desktop]$ 
```

/home/cloudera/Desktop is the current working directory and churn_prediction.jar and run_project.sh file is copied to this location.

5. Run the run_project.sh file by using below command

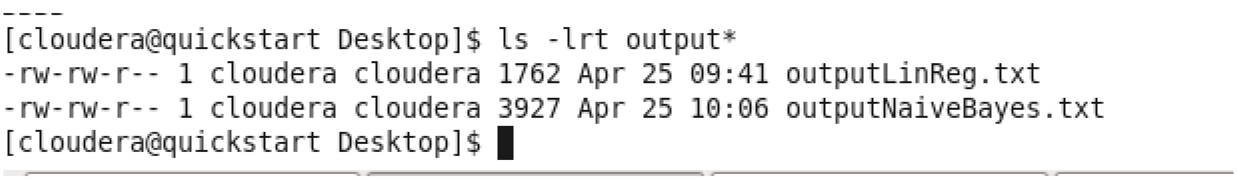
sh run_project.sh

When we execute the script, the console will ask user to select an algorithm to execute on given dataset. On selecting option here, that particular algorithm will be executed on the dataset, will build a prediction model and generate the output files and prediction files in current working directory and /spl/project/LinReg/predictions/ folder



```
[cloudera@quickstart Desktop]$ sh run_project.sh
Please select algorithm one of the algorithm:(1 or 2)
1. Linear Regression
2. Naive Bayes
1
```

Below are the output files generated after running 2 algorithms

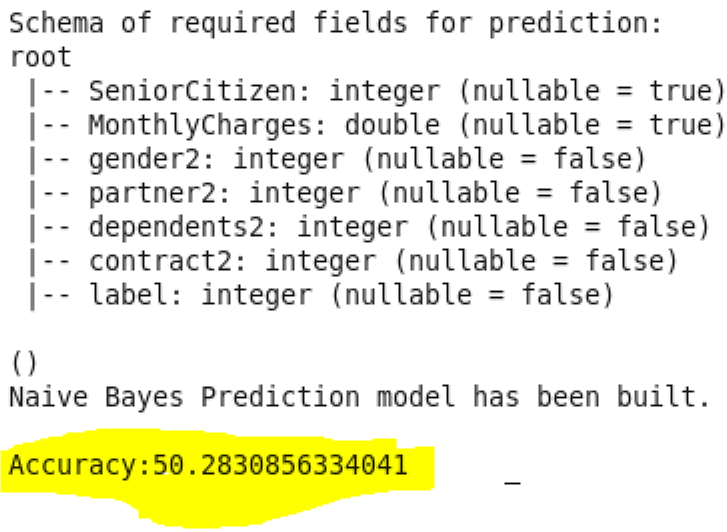


```
----
[cloudera@quickstart Desktop]$ ls -lrt output*
-rw-rw-r-- 1 cloudera cloudera 1762 Apr 25 09:41 outputLinReg.txt
-rw-rw-r-- 1 cloudera cloudera 3927 Apr 25 10:06 outputNaiveBayes.txt
[cloudera@quickstart Desktop]$ 
```

Prediction files are generated in below location

**Output:**

**Screenshot of Naïve Bayes output:**

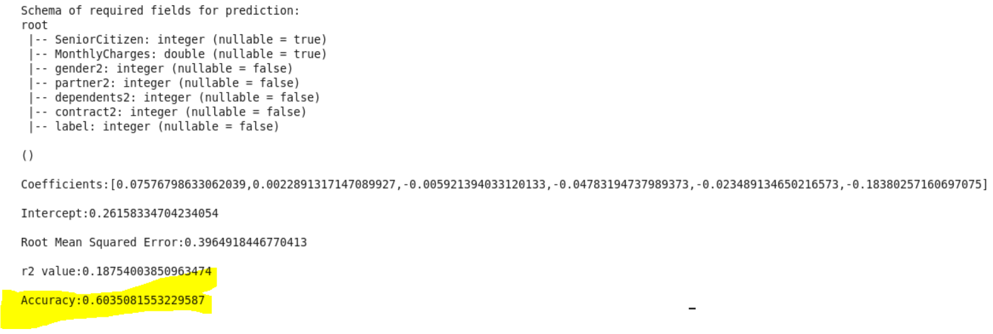


```
Schema of required fields for prediction:
root
 |-- SeniorCitizen: integer (nullable = true)
 |-- MonthlyCharges: double (nullable = true)
 |-- gender2: integer (nullable = false)
 |-- partner2: integer (nullable = false)
 |-- dependents2: integer (nullable = false)
 |-- contract2: integer (nullable = false)
 |-- label: integer (nullable = false)

()
Naive Bayes Prediction model has been built.

Accuracy:50.2830856334041
```

## Screenshot of Linear Regression output:



```
Schema of required fields for prediction:
root
 |-- SeniorCitizen: integer (nullable = true)
 |-- MonthlyCharges: double (nullable = true)
 |-- gender2: integer (nullable = false)
 |-- partner2: integer (nullable = false)
 |-- dependents2: integer (nullable = false)
 |-- contract2: integer (nullable = false)
 |-- label: integer (nullable = false)

()

Coefficients:[0.07576798633062039,0.0022891317147089927,-0.005921394033120133,-0.04783194737989373,-0.023489134650216573,-0.18380257160697075]

Intercept:0.26158334704234054

Root Mean Squared Error:0.3964918446770413

r2 value:0.18754003850963474

Accuracy:0.6035081553229587
```

# 7. Tools and Technologies used:

## 7.1. Technologies:

- **Apache Maven**: Apache Maven is a software project management and comprehension tool which is used to automate the build process.
- **Shell scripting**: A shell script is a computer program which can be executed by a Unix shell. It is basically a file containing a list of commands, which are read by the shell and run as though they were entered on a command line.
- **Scala**: Is a programming language
- **Spark**: Apache Spark is a distributed general-purpose cluster-computing framework.

## 7.2. Tools:

- **Eclipse**: Is an IDE for Scala

- **Cloudera**: Cloudera distribution for Hadoop ( an open source ecosystem for storing and analyzing data)
- **VMware workstation**: To enable setting up virtual machines on a single physical machine.

## 8. Conclusion:

To conclude, we have researched the ‘Apache Spark’ environment with ‘Scala’ programming language and used their suitable features in our project to predict the ‘Churn’ for the Telecom Industry. Basically, we have used one classification model (Naive Bayes) and one Regression model (Linear Regression) for this purpose. After training the data with these models and after analyzing the accuracies of each of the models, we see that the Linear Regression model gave the best results with an accuracy of 60.35%, when compared to the Naive Bayes Model whose accuracy is 50.28%.

## References:

- https://acadgild.com/blog/spark-rdd-scala
- https://spark.apache.org
- https://www.oreilly.com/library/view/mastering-apache-spark/9781786462749/Text/e9821f5d-6ad5-4533-8670-f18aef153d6f.xhtml
- https://spark.apache.org/mllib/
- https://www.kaggle.com/blastchar/telco-customer-churn/version/1.
- https://www.gavstech.com/why-scala-for-big-data-and-machine-learning/