(Big Data Analytics Midterm)

Question1 –

1. Pre-processing steps considered are:-
2. Read the YearPerdictionFile.txt file and each sample in data has been normalized using L^2 normalizer and L Infinity Normalizer. L2 normalization is a vector norm.
3. Use the ChiSelector to select the different features in the Algorithms. As the doesn’t allow more than 10000 column values, this Selector cannot be applied to Year PerdictionData.

Following are the Error:

// Chi test cannot be performed , because there are more than 10000 differnt values in each column

// Discretize data in 16 equal bins since ChiSqSelector requires categorical features

/\*val discretizedData = data.map { line =>

val parts = line.split(',')

val myList = Array(parts(12), parts(78))

LabeledPoint(parts(91).toDouble, Vectors.dense(myList.map(x => x.toDouble).toArray))

}

// Create ChiSqSelector that will select 50 features

val selector = new ChiSqSelector(50)

// Create ChiSqSelector model (selecting features)

val transformer = selector.fit(discretizedData)

// Filter the top 50 features from each feature vector

val filteredData = discretizedData.map { line =>

LabeledPoint(line.label, transformer.transform(line.features))

}\*/

1. Result comparison – PCA(Principal Component Analysis) and SVD(Singular Value Decomposition)

(Classification)

PCA is used to feature reduction.

1. Used MultivariateStatisticalSummary in order to get the mean median , mode, sum values of each columns.

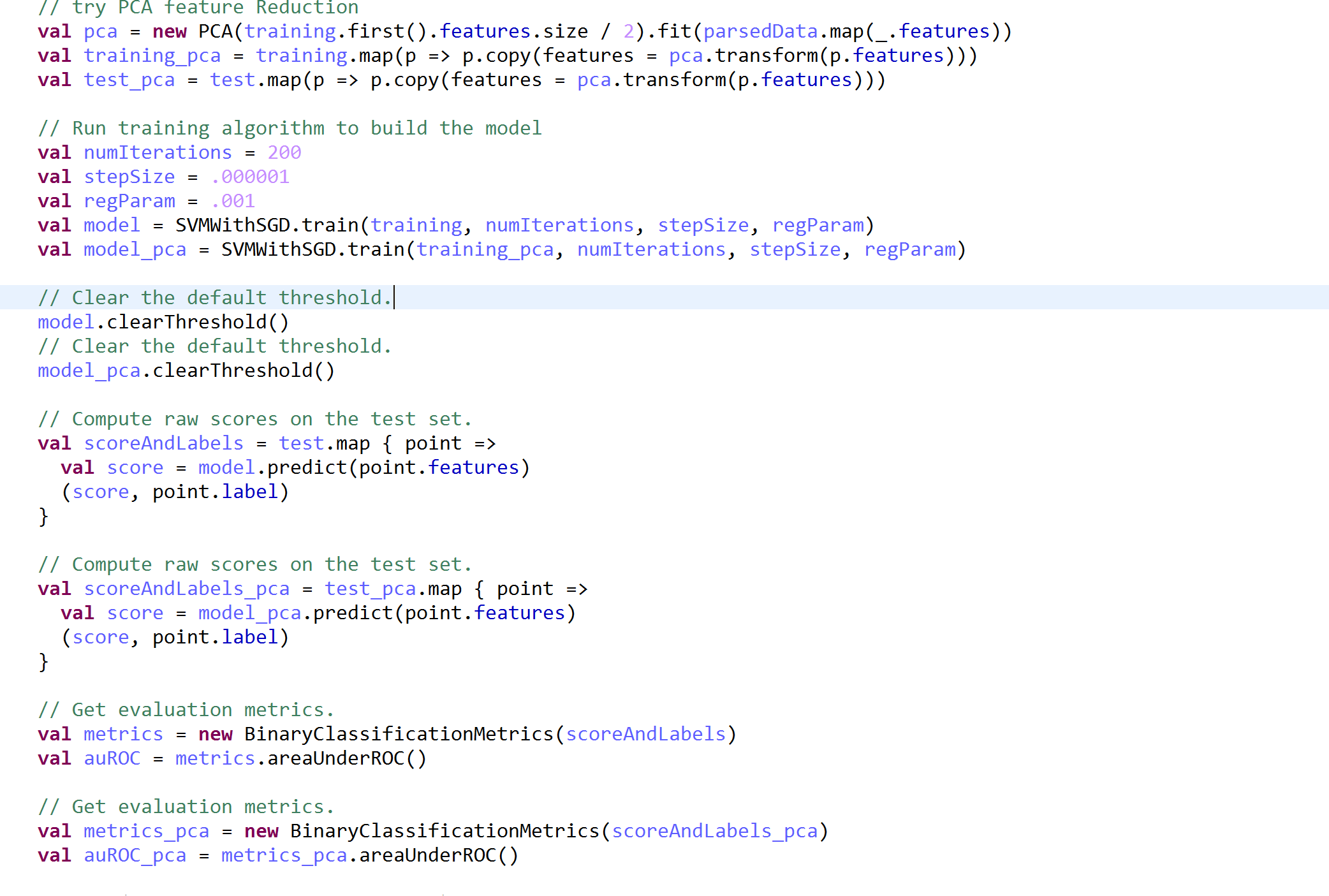
Below is the code:

val summary: MultivariateStatisticalSummary = Statistics.colStats(vectorData)

SVMwithSGD Code Implementation



SVMwithSGD (PCA) code Implementation



SVMwithSGD SVMwithSGD (PCA)

Area under ROC - 0.451490631847 0.4858363297

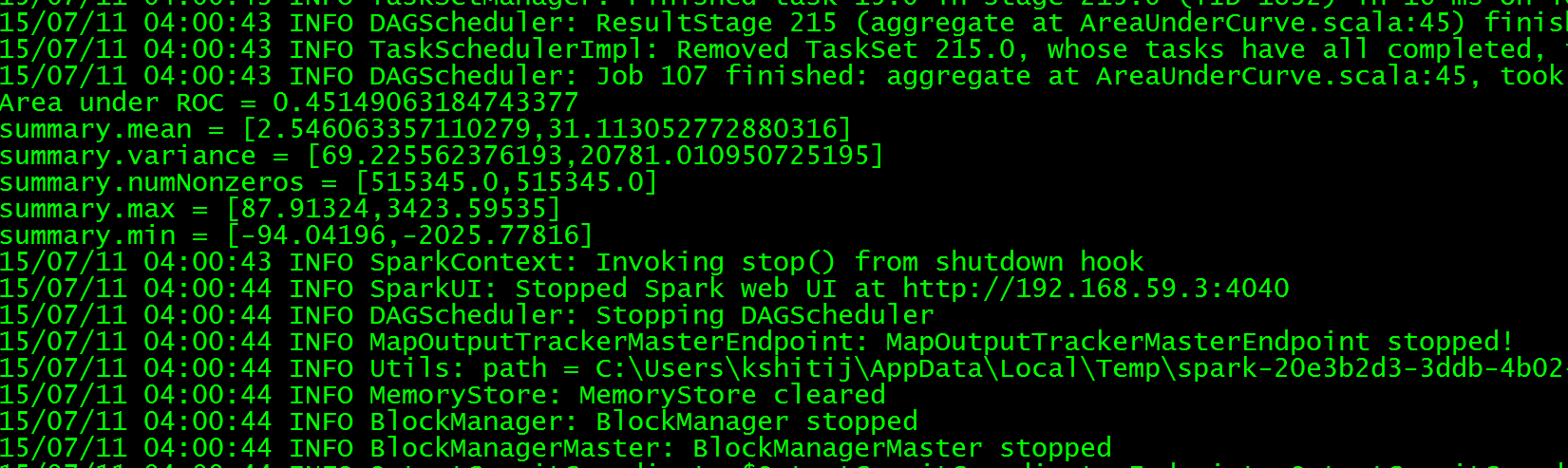
Summary mean – [2.546063337, 31.11305277] [2.546063337, 31.11305277]

Summary variance – [69.225562376, 20781.010950] [69.225562376, 20781.010950]

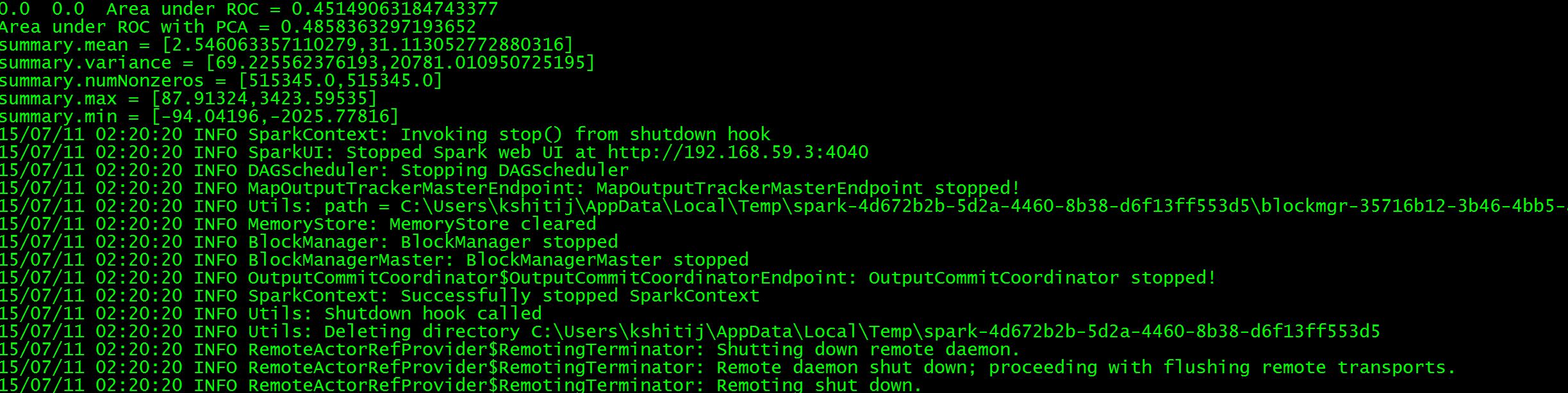
Summary numnonzeroes – [515345.0, 515345.0] [515345.0, 515345.0]

Summary max - [87.913, 3423.59] [87.913, 3423.59]

Summary min – [-94.04196, -2025.77816] [-94.04196, -2025.77816]

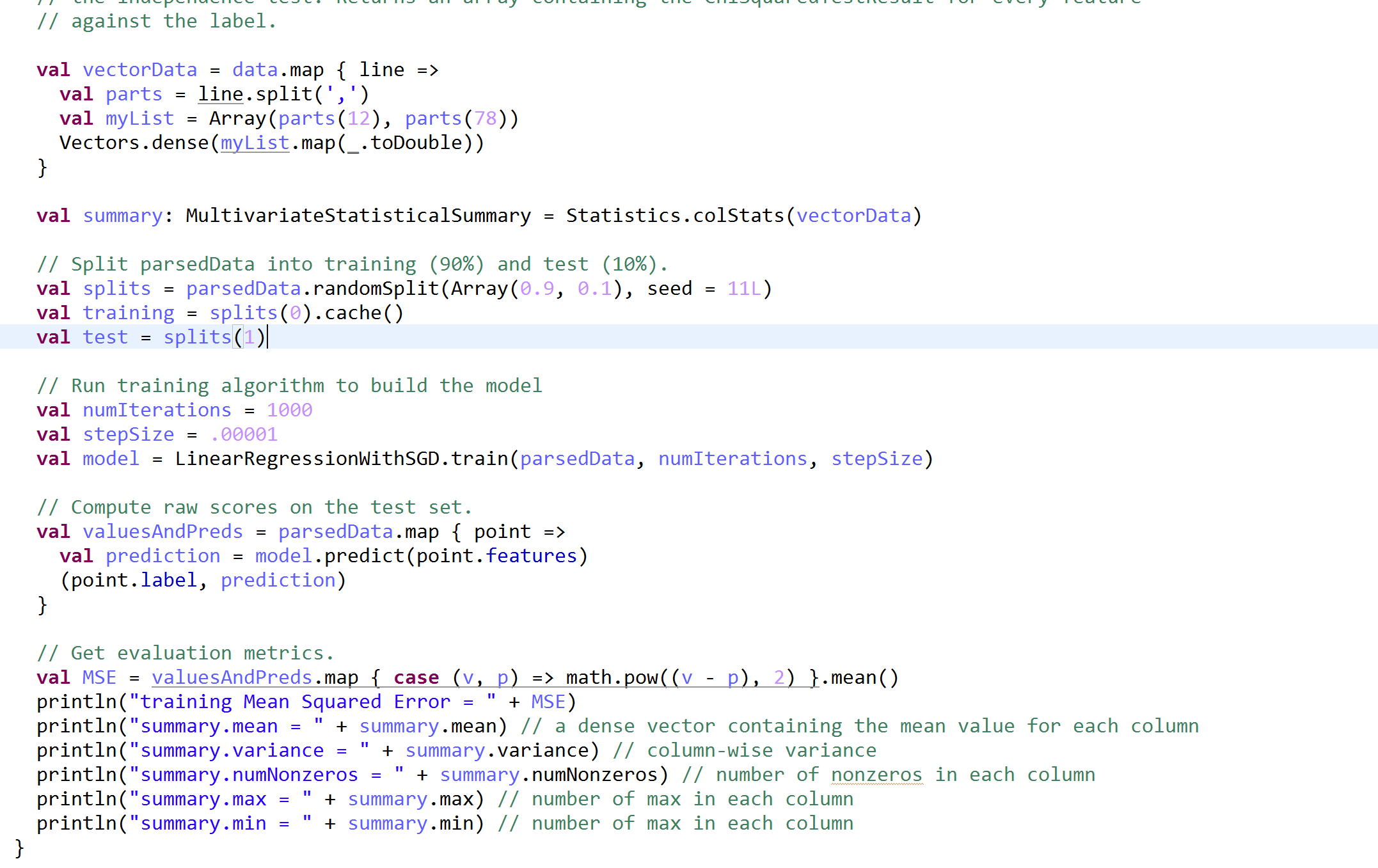


With PCA

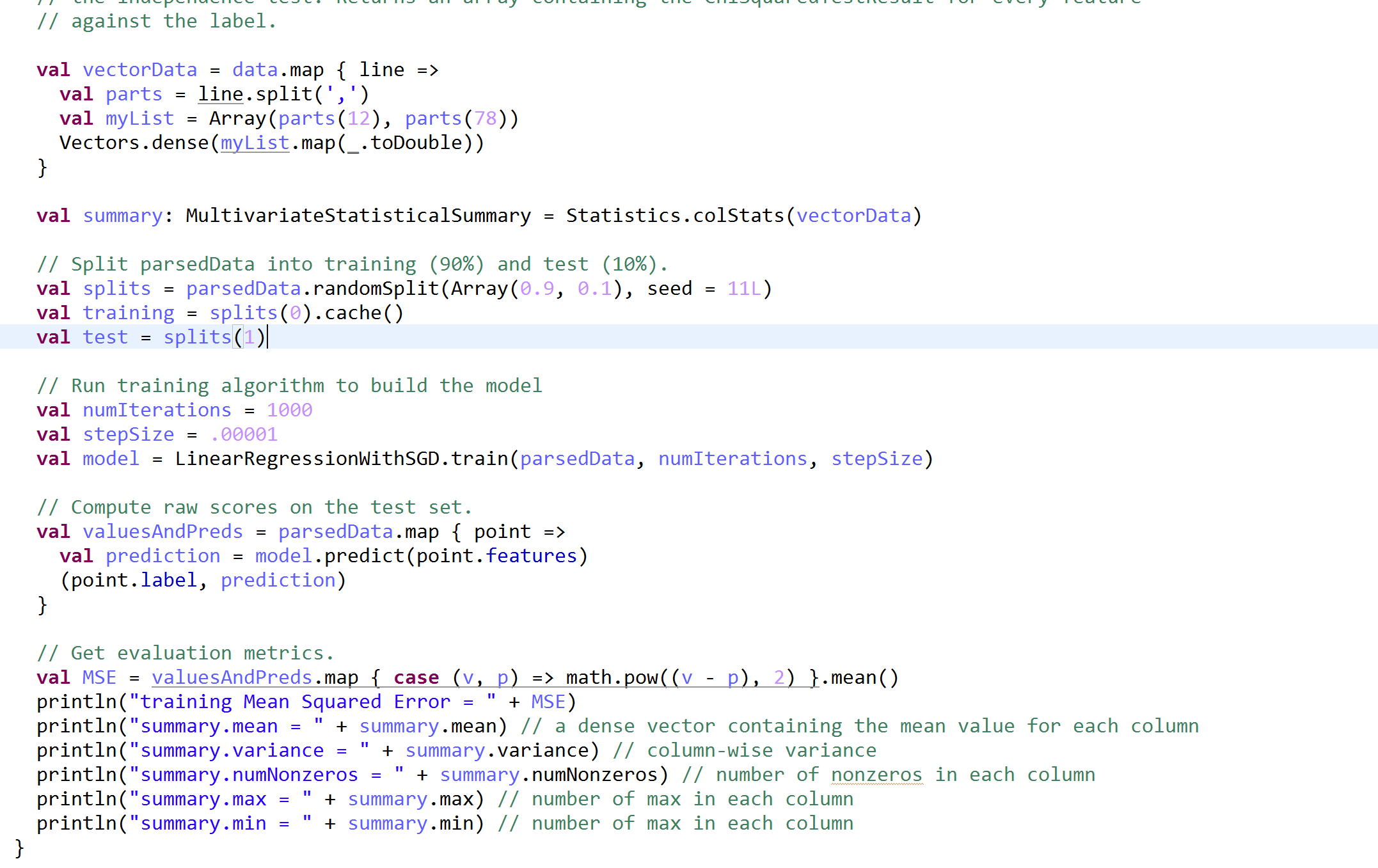


(Regression)

LinearRegressionwithSGD Code Implementation



LinearRegressionwithSGD (PCA) Code Implementation



LinearRegressionwithSGD LinearRegressionwithSGD (PCA)

Training mean squared error - .75798737993133 .75798737993133

Summary mean – [2.546063337, 31.11305277] [2.546063337, 31.11305277]

Summary variance – [69.225562376, 20781.010950] [69.225562376, 20781.010950]

Summary numnonzeroes – [515345.0, 515345.0] [515345.0, 515345.0]

Summary max - [87.913, 3423.59] [87.913, 3423.59]

Summary min – [-94.04196, -2025.77816] [-94.04196, -2025.77816]

1. SVMwithSGD LinearRegressionSGD

Area under ROC - 0.451490631847 Training mean squared error - .75798737993133

Command for Question 1:

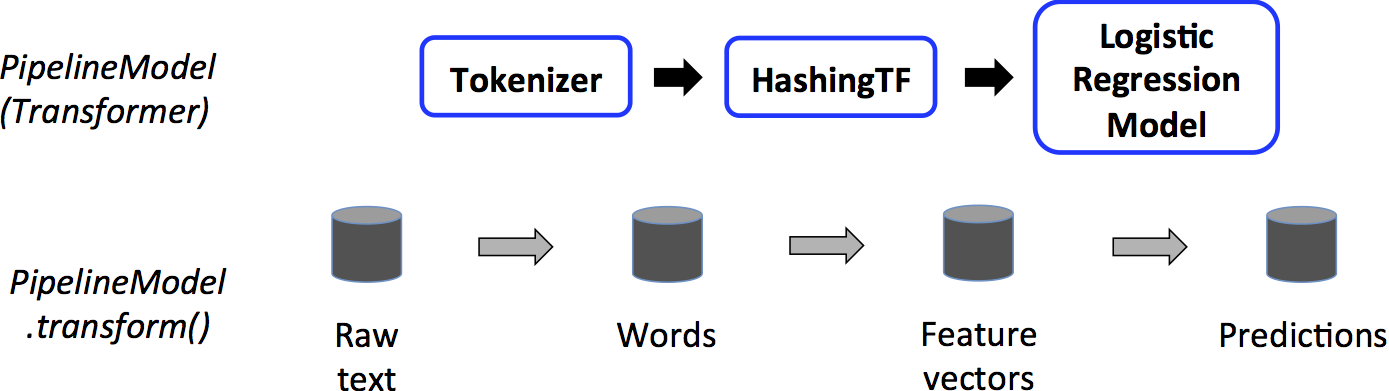
C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.reg.MusicYearPerdictREG" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar

C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.reg.MusicYearPerdictREGLASSO" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar

C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.reg.MusicYearPerdictREGPCA" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar

Question 2—

1. Pre-Processing steps –
2. The data present in the file needed to be cleaned as there were some invalid values present like “ ?”. For that we first created a dataframe using a csv file and then converted the dataframe to dataframeNAFunctions to use the functions to deal with the Na values to clean the data.
3. As the data had categorical values so changing the categorical values to the numerical values was important and for that we used spar
4. OneHotEncoder to convert the categorical dataframes values into numerical ones.
5. Covert the dataframe use the Estimator, Transformer, and ParamMap in order to covert the Pipeline model using Traing dataset frames in Logistic regression Model.
6. Use the Test Data in to predict the result based on the features and labels from the traing data model.



1. Two Different classification methods used were SVMwithSGD and LogisticRegressionwithSGD

to build the training set model for the perdiction.

SVMwithSGD LogisticRegressionwithSGD

Error .01135 .01135

Logistic Regression Code



Command for Question 2:

C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.df.AdultDataSet" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar

Question 3—

1. Pre-Processing steps used

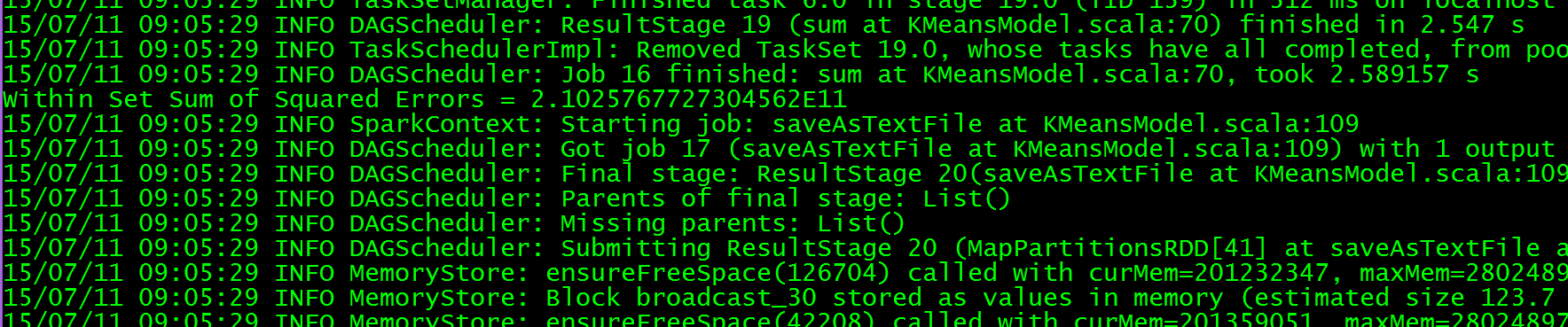
For preprocessing in clustering, we saved all the file in one location on the drive from where we can pull all the files at a time using the “\*” function so that clustering can be done on those files using the below clustering algorithms

Clustering methods used

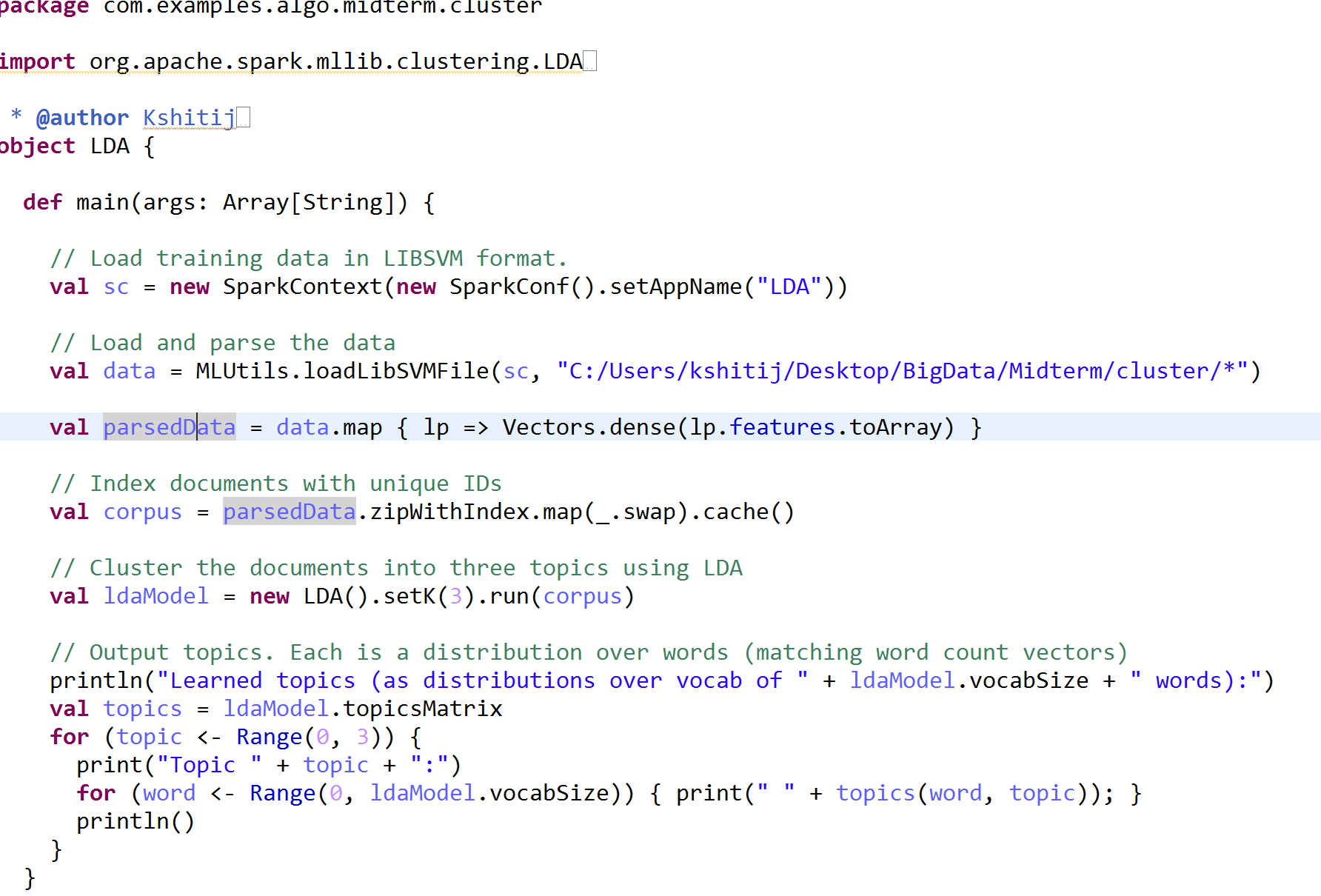
1. KMeansCluster Code



Within Set Sum of Squared Errors = 2605.254084081905



1. LDA – Latent Dirichlet allocation Code Implementation



Command for Question 3:

C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.cluster.KMeansCluster" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar

C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.cluster.GausianMixCluster" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar

C:/spark-1.4.0-bin-hadoop2.6/bin/spark-submit --class "com.examples.algo.midterm.cluster.LDA" --master local[1] target/spark-scala-maven-project-0.0.1-SNAPSHOT.jar