CSYE7374 52001 ST: Big-Data Sys & Int Anltcs
Group 5
Midterm Report

Part1: Regression and Classification

Yarn and hadoop setup is being used to run all the applications using spark which is built using yarn, so the text file provided was moved to hdfs

Data Exploration:

1. Data was in text file format, data was loaded using spark into LabeledPoint
2. Used random split to get training and test data

Feature Engineering

1. Scaled the features using the standard scalar function in ml
2. Used PCA to perform dimension reduction and used 25, 50 features

Model:
Used both Classification and Regression models on the data

Regression models applied:

1.LinearRegressionWithSGDL1
2. LinearRegressionWithSGDL1(PCA-10)
3.LinearRegressionWithSGDL1(PCA-50)
4.RidgeRegression
5.RidgeRegressionPCA25
6.RidgeRegressionPCA50
7.LassoRegression
8.LassoRegressionPCA25
9.LassoRegressionPCA50

Classification models applied:

1.LogisticRegressionWithLBFGSL1
2.LogisticRegressionWithLBFGSL1PCA25
3.LogisticRegressionWithLBFGSL1PCA50
4.SVMWithSGD(L1)
5.SVMWithSGDPCA25(L1)
6.SVMWithSGDPCA50(L1)
7.SVMWithSGD(L2)
8.SVMWithSGDPCA25(L2)
9.SVMWithSGDPCA50(L2)

ML classification algorithms:

LogisticRegression

Algorithms	Metrics			
Regression algos	**MSE**	**Variance**	**RME**	**Explained Variance**
LinearRegressionWithSGDL1	3860028.08	26.34	1964.69	-3.97
LinearRegressionWithSGDL1(PCA-25)	3958729.86	21.63	1989.65	-5.43
LinearRegressionWithSGDL1(PCA-50)	3921124.32	24.83	1980.18	-4.54
RidgeRegression	2843878.47	2242.55	1686.38	0.0061
RidgeRegressionPCA25	3505487.25	2190.60	1872.29	-0.031
RidgeRegressionPCA50	3327905.98	2261.59	1824.25	-0.026
LassoRegression	2843871.60	2242.57	1686.37	0.0061
LassoRegressionPCA25	3505484.54	2190.63	1872.29	-0.031
LassoRegressionPCA50	3327902.72	2261.62	1824.25	-0.026
Classification algos	Confusion Matrix	Recall	Precision	Accuracy
LogisticRegressionWithLBFGSL1	0.0 1664.0 0.0 204878.0	Recall(0):0.0 Recall(1):1.0	Precision(0): 0.0 Precision(1): 0.9919	0.99194354

Algorithms	**Metrics**			
LogisticRegressionWithLBFGSL1PCA25	0.0 1664.0 196.0 204682.0	Recall(0):0.0 Recall(1):0.9990	Precision(0): 0.0 Precision(1): 0.9919	0.9909946
LogisticRegressionWithLBFGSL1PCA50	0.0 1664.0 62.0 204816.0	Recall(0):0.0 Recall(1):0.9996	Precision(0): 0.0 Precision(1): 0.9919	0.99164337
Classification algos	Accuracy	Area under ROC		
SVMWithSGD(L1)	0	0.7003		
SVMWithSGDPCA25(L1)	0	0.3591		
SVMWithSGDPCA50(L1)	0	0.3542		
SVMWithSGD(L2)	0	0.636		
SVMWithSGDPCA25(L2)	0	0.569		
SVMWithSGDPCA50(L2)	0	0.606		

ML Algorithm: Logistic Regression:

Training error: 00816

Best set of parameters:
{
    hashing -numFeatures: 25,
    logreg_regParam: 0.1

}

Part 2: Classification Algorithm

Data Exploration:

1. The data presented had categorical values and missing fields.
2. The data was cleaned using physical inspection.
3. The missing fields was filled by calculationg mode value.
4. The categorical data was converted to numerical data using the the python pandas api, with the function .getDummy()
5. The converted was loaded into the RDD using the load text file function
6. Randomsplit was used to split the data into training and test

Feature Engineering:

1. Scaled the features using the standard scalar function in ml
2. Normalized the data using the normalizer

Model

1. Following model were created using ML library:
    a. Logistic regression
2. The Following algorithms were used in Mlib library:
    a. SVMwithSGD
    b. Logistic Regression with SGD
    c. Logistic Regression with LBFGS

Model evaluation:

1. Used cross validator with binaryclassificationevaluator

Model Selection:

1. used the results from CrossValidator – training error to select a model
    a. The best possible model was for **Logistic Regression** (ML) algorithm with an error 0.24089

SVMwithSGD

- Area Under ROC -Area under ROC = 0.8990239900396924

LogisticRegression With SGD

- Recall:0.8437906994560638
- Precision:0.8437906994560638
- accuracy:0.8437907

LogisticRegression With LBFGS

- Precision(0):0.8804617439419958
- Precision(1):0.7479579929988331
- Accuracy:0.85436296

Part 3: Clustering Algorithm

Data Exploration:

1. Initially, the data was in the format of libsvm. This file was loaded to a Labelled point RRD using the MUtils function loadLibSVMFile function
2. The Labelled point RRD's features are used to create a vector of RDD
3. Used randomSplit to split into training and test data.

Data Summarization

1. Summarized the data by using MultivariateStatisticalSummary
2. Visualized the data by printing mean,variance and non-zeros

Feature Engineering

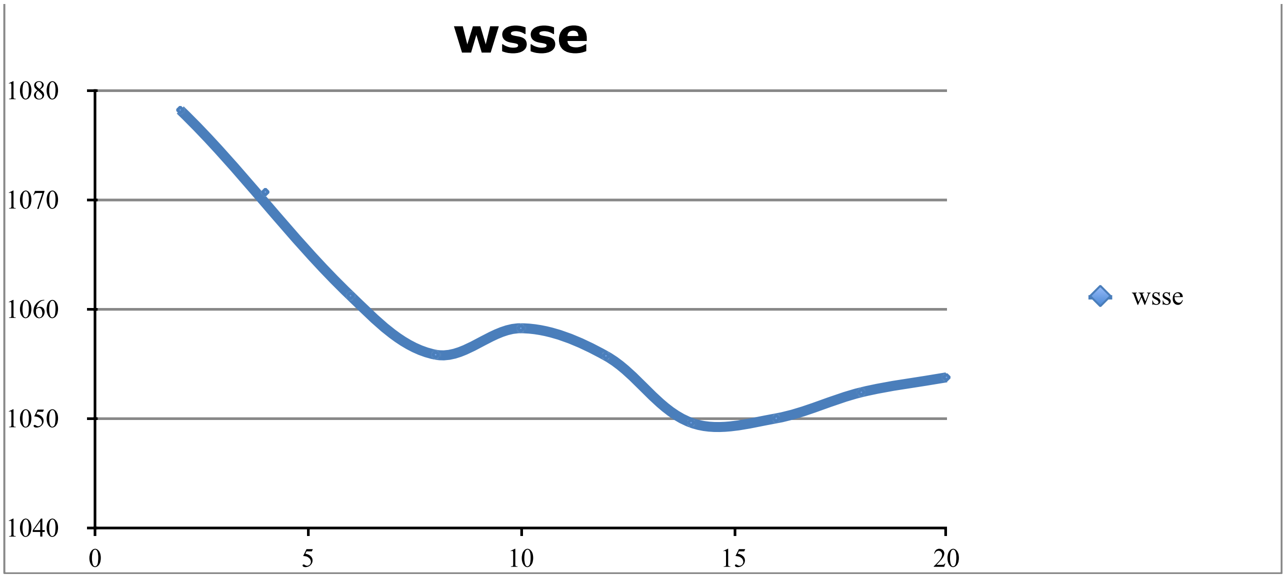
1. Scaled the features using the standard scalar function in ml
2. Normalized the data using the normalizer

Model

1. Used a K-Means clustering algorithm to create a kMeansModel
2. The iteration are set to a standard of 10

Model evaluation:

1. Constructed the elbow graph for cluster selection.



Model Selection:

1. With the inputs from the graph we can see that for the value of k = 5 the values the model is most ideal.
2. The evaluation for K = 5 for this model is:
    a. WSSE: 1068.0635002245176