Ridge Regression Algorithm

Michael Ye

CS 9223 – Big Data

Difficulties/Challenges

I would say it was a pretty tough but really useful project. Spark/Java was difficult while R was relatively easy. I don't have extensive CS experience coming from MOT but am glad I'm taking this course and learning these technologies and challenging myself.

General Definition/Description

Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. Ridge Regression performs 'L2 Regularization' (adds L2 Penalty), by adding a factor of sum of squares of coefficients in the optimization objective. Hence the algorithm is used to prevent overfitting and underfitting by shrinking the coefficients by the same factor.

Mathematical formula

Ridge regression seeks to minimize the following equation:

$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j}^{m} \beta_j^2$$

The summation term on the left above is the typical RSS error measure in a linear regression context. The expression on the right is a penalty term which increases as the coefficients become larger. The value of lambda determines the importance of this penalty term.

Generative or discriminative Algorithm

Ridge Regression is a discriminative algorithm as it models the dependence of unobserved (target) variables y on observed variables x. Therefore, it learns the conditional probability distribution P(y|x) where the probability of y given x.

Any assumptions the algorithm relies on

Ridge Regression doesn't actually have a definite rule for selecting the values of the regularization parameters. It is up to the user through cross validation methods and such.

Restrictions on the independent variable (can they be categorical and numerical, scale, variance)

Suitable for both continuous and categorical.

Restrictions on the dependent variable (categorical or numerical, scale, variance)

When using linear regression with L2 (ridge Regularization), the dependent variable would have to be continuous while in a Ridge Logistic Regression, it is categorical.

Train and Test Datasets Preparation

Split: 80/20

Missing values for Age and Fare replaced by Median.

Features Used: Pclass, Sex, Age, Fare

Label: Survived

Took out header columns

R Approach

Categorical variables transformed into factors.

Glmnet logistic regression was used alongside Ridge penalties (alpha = 0)

Accuracy: 0.826815642458101

AUC: 0.799388586956522

Spark Approach MLLIB classification used: LogisticRegressionWithLBFGS

Used to train a binary Logistic Regression using Limited-memory BFGS. Standard feature scaling and L2 regularization (Ridge regression penalization) are used by default.

Accuracy: 0.826815642458101

AUC: 0.799388586956522

Screenshots: Spark

Accuracy:

```
17/12/06 22:29:43 INFO ContextCleaner: Cleaned accumulator 531
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_42_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO ShuffleBlockFetcherIterator: Getting 1 non-empty blocks out of 1 blocks
17/12/06 22:29:43 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_24_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_40_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO ContextCleaner: Cleaned shuffle 0
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_36_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO Executor: Finished task 0.0 in stage 24.0 (TID 24). 1367 bytes result sent to driver
17/12/06 22:29:43 INFO TaskSetManager: Finished task 0.0 in stage 24.0 (TID 24) in 10 ms on localhost (exec
17/12/06 22:29:43 INFO TaskSchedulerImpl: Removed TaskSet 24.0, whose tasks have all completed, from pool
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_28_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO DAGScheduler: ResultStage 24 (countByValue at MulticlassMetrics.scala:42) finished i
17/12/06 22:29:43 INFO DAGScheduler: Job 22 finished: countByValue at MulticlassMetrics.scala:42, took 0.06
Accuracy = 0.8268156424581006
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_43_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_30_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_34_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_38_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO BlockManagerInfo: Removed broadcast_44_piece0 on 12.42.205.8:52868 in memory (size:
17/12/06 22:29:43 INFO ContextCleaner: Cleaned accumulator 48
17/12/06 22:29:43 INFO SparkContext: Starting job: collect at BinaryClassificationMetrics.scala:192
17/12/06 22:29:43 INFO DAGScheduler: Registering RDD 38 (map at RidgeRegression.java:72)
17/12/06 22:29:43 INFO DAGScheduler: Registering RDD 45 (combineByKey at BinaryClassificationMetrics.scala:
17/12/06 22:29:43 INFO DAGScheduler: Got job 23 (collect at BinaryClassificationMetrics.scala:192) with 1 o
17/12/06 22:29:43 INFO DAGScheduler: Final stage: ResultStage 27 (collect at BinaryClassificationMetrics.sc
17/12/06 22:29:43 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 26)
```

Area Under ROC:

```
17/12/06 22:30:33 INFO TaskSetManager: Starting task 0.0 in stage 57.0 (TID 45, localhost, execu
17/12/06 22:30:33 INFO Executor: Running task 1.0 in stage 57.0 (TID 44)
17/12/06 22:30:33 INFO Executor: Running task 0.0 in stage 57.0 (TID 45)
17/12/06 22:30:33 INFO Executor: Finished task 0.0 in stage 57.0 (TID 45). 748 bytes result sent
17/12/06 22:30:33 INFO BlockManager: Found block rdd_49_0 locally
17/12/06 22:30:33 INFO TaskSetManager: Finished task 0.0 in stage 57.0 (TID 45) in 5 ms on local
17/12/06 22:30:33 INFO Executor: Finished task 1.0 in stage 57.0 (TID 44). 834 bytes result sent
17/12/06 22:30:33 INFO TaskSetManager: Finished task 1.0 in stage 57.0 (TID 44) in 5 ms on local
17/12/06 22:30:33 INFO TaskSchedulerImpl: Removed TaskSet 57.0, whose tasks have all completed,
17/12/06 22:30:33 INFO DAGScheduler: ResultStage 57 (aggregate at AreaUnderCurve.scala:45) finis
17/12/06 22:30:33 INFO DAGScheduler: Job 33 finished: aggregate at AreaUnderCurve.scala:45, took
Area under ROC = 0.7993885869565218
17/12/06 22:30:33 INFO SparkUI: Stopped Spark web UI at http://12.42.205.8:4040
17/12/06 22:30:33 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
17/12/06 22:30:33 INFO MemoryStore: MemoryStore cleared
17/12/06 22:30:33 INFO BlockManager: BlockManager stopped
17/12/06 22:30:33 INFO BlockManagerMaster: BlockManagerMaster stopped
17/12/06 22:30:33 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoor
17/12/06 22:30:33 INFO SparkContext: Successfully stopped SparkContext
```

Screenshots: R

my779@F4Linux1:~

```
Loading required package: foreach
foreach: simple, scalable parallel programming from Revolution Analytics
Use Revolution R for scalability, fault tolerance and more.
http://www.revolutionanalytics.com
Loaded glmnet 2.0-10

> library(ROCR)
Loading required package: gplots

Attaching package: 'gplots'

The following object is masked from 'package:stats':

    lowess

>
> trainData<-read.csv("train.csv", header=FALSE,stringsAsFactors=FALSE)
> testData<-read.csv("test.csv", header=FALSE,stringsAsFactors=FALSE)
>
> testData$V1 <- factor(testData$V1) #survived
> testData$V2 <- factor(testData$V2) #pclass
> testData$V3 <- factor(testData$V3) #sex
>
> trainData$V1 <- factor(trainData$V1) #survived
> trainData$V2 <- factor(trainData$V2) #pclass
> trainData$V3 <- factor(trainData$V3) #sex
>
>
> x <- data.matrix(trainData[,c(2,3,4,5)])
> newx <- data.matrix(testData[,c(2,3,4,5)])
> y <- trainData$V1
>
> cvfit.m.ridge <- cv.glmnet(x, y,
+                            family = "binomial",
+                            alpha = 0,
+                            type.measure = "class")
>
> pred_ridge <- predict(cvfit.m.ridge, newx = newx, s = 'lambda.min', type='resp
> pred_ridge <- round(pred_ridge)
> confusion.matrix <- prop.table(table(pred_ridge, testData$V1))
> accuracy <- confusion.matrix[1,1] + confusion.matrix[2,2]
> paste("Accuracy: ", accuracy)
[1] "Accuracy:  0.826815642458101"
>
>
> ridge_pred_class <- prediction(pred_ridge,testData$V1)
> ridge_perf_rocr <- performance(ridge_pred_class, measure='tpr',x.measure='fpr'
> ridge_slot_fp <- slot(ridge_pred_class, "fp")
> ridge_slot_tp <- slot(ridge_pred_class, "tp")
>
>
> ridge_perf_auc <- performance(ridge_pred_class, measure="auc")
> ridge_AUC <- ridge_perf_auc@y.values[[1]]
> paste("AUC: ", ridge_AUC)
[1] "AUC:  0.799388586956522"
>
```

References:

http://scikit-learn.org/stable/modules/linear_model.html#ridge-regression

https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Ridge_Regression.pdf

https://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf

https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-ridge-lasso-regression-python/

https://spark.apache.org/docs/2.0.1/api/java/org/apache/spark/mllib/classification/LogisticRegressionWithLBFGS.html

https://spark.apache.org/docs/2.1.0/ml-classification-regression.html