LOGISTIC REGRESSION WITH CONJUGATE GRADIENT DESCENT FOR DOCUMENT CLASSIFICATION

by

SRUTHI NAMBURI

B.Tech, Jawaharlal Nehru Technology University (JNTU), India, 2013

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Science
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2016

Approved by:

Major Professor
Dr. William Hsu

Copyright

Abstract

Logistic regression is a model for function estimation that measures the relationship between independent variables and a categorical dependent variable, and by approximating a conditional probabilistic density function using a logistic function, also known as a sigmoidal function. Multinomial logistic regression is used to predict categorical variables where there can be more than two categories or classes. The most common type of algorithm for optimizing the cost function for this model is gradient descent. In this project, I implemented logistic regression using conjugate gradient descent (CGD). I used the *20 Newsgroups* data set collected by Ken Lang. I compared the results with those for existing implementations of gradient descent. The conjugate gradient optimization methodology outperforms existing implementations.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my deepest gratitude to my advisor, Dr. William Hsu, for his excellent guidance in completing this project and for his constant support during my graduate study at K-state. I would also like to thank Dr. Mitch Neilsen and Dr. Eugene Vasserman for serving on my M.S. committee. Finally, I thank my parents, Dr. Namburi Prasada Raju, Namburi Ramavathi and my brother Namburi Aditya Varma for their love and support.

Chapter 1

Introduction

With the increasing availability of electronic documents, news sites and blogs, automatic classification of documents has become a key method for categorizing data. Document classification aims to assign a document to a class or category. Machine learning and natural language processing techniques are applied to achieve this task.

Supervised learning builds a model that approximates the mapping between the input and the output, and can predict the output of the system given new inputs.[1;2] If the output takes a finite set of discrete values that indicate the class labels of the input, the learned mapping leads to the classification of the input data. It creates a model based on the training data. Categories are predefined, and documents within the training dataset are manually tagged with one or more category labels. A classifier is then trained on the dataset which predicts the category of a new document.

Building a multiclass classifier is a complex task, especially when compared with binary classification.[3] In multiclass classification, each training point belongs to one of N different classes. The goal is to construct a function which, given a new data point, will correctly predict the class to which the new point belongs. One-vs-Rest strategy is used to convert a binary classifier into a multiclass classification algorithm .[4]

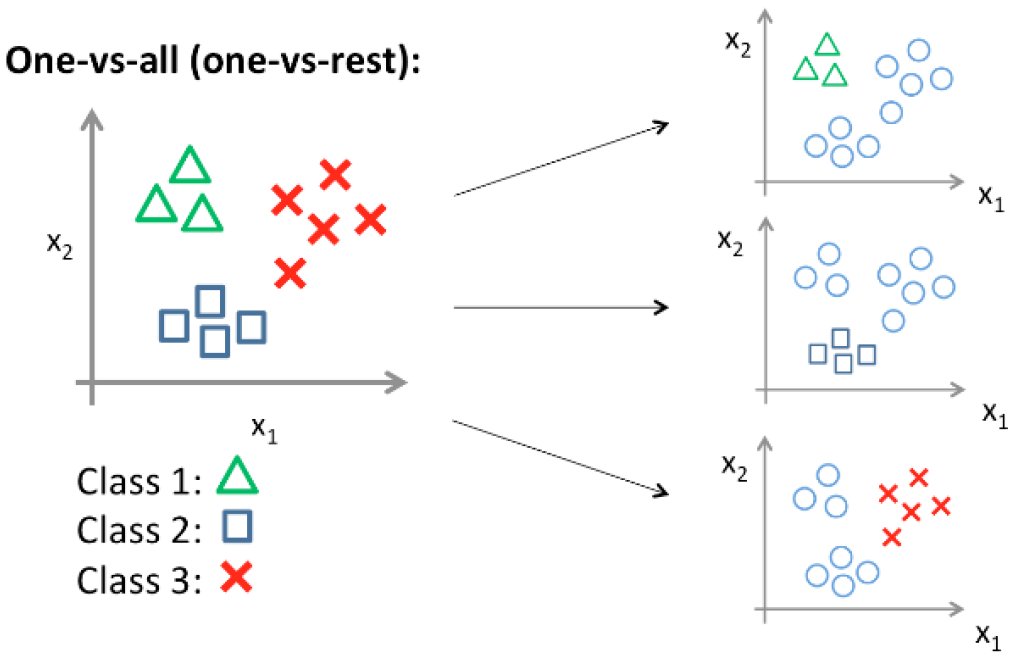

Figure 1.1: *Illustration of one-vs-rest strategy with 3 classes. Adapted from (Andrew Ng, 2009).*

For each of the N classes, a binary model is created. Each of these binary models for the individual classes is assessed against its complement (all other classes in the model) as though it were a binary classification issue. Prediction is then performed by running these binary classifiers, and choosing the prediction with the highest confidence score.

Chapter 2

Background and Related Work

This chapter introduces the classification techniques used in this project. I have used the logistic regression model to classify documents. The representation of the logistic model and its optimization techniques are described.

2.1 Logistic Regression

Logistic regression is a model for function estimation that measures the relationship between independent variables and a categorical dependent variable, and by approximating a conditional probabilistic density function using a logistic function, also known as a sigmoidal function.[5;6]

Logistic regression does not make many of the key assumptions of the general linear models regarding linearity and normality,[7] which are based on ordinary least squares algorithms. It does not need a linear relationship between the dependent and independent variables. It applies a non-linear log transformation to the predicted odds ratio and hence can handle all sorts of relationships. However, it requires that the independent variables are linearly related to the log odds. Otherwise, the test underestimates the strength of the relationship, and rejects the relationship as insignificant, too easily.

2.1.1 Hypothesis Representation

The *hypothesis function*, $h_\theta(x)$, is used to represent the approximation to the target concept function that is returned by learning algorithm on a classification task. The classifier outputs values in the range $0 \leqslant h_\theta(x) \leqslant 1$. The predicted value lies between 0 and 1. The hypothesis function for logistic regression is given by

$$h_\theta(x) = g(\theta^T x)$$

where x is a column vector of input variables/features and θ is a column vector of parameters.

The sigmoidal/logistic function, g(z), where z is a real valued number is given by

$$g(z) = \frac{1}{1 + e^{-z}}$$

Alternatively, this can be written as

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

The sigmoidal function can be graphically represented as follows

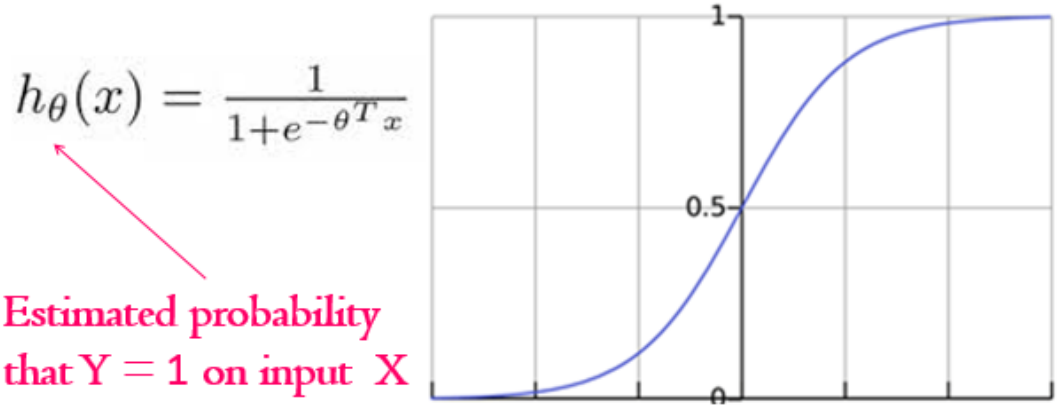

Figure 2.1: *Graphical representation of sigmoidal function. Adapted from (Andrew Ng, 2009).*

2.1.2 Interpretation of Hypothesis Output

The output of the hypothesis function is the estimated probability that $y = 1$ on input x. For example if $h_\theta(x)$ is 0.7, it means that the probability of the given input falling into a certain category is 70%. A decision boundary gives a better sense of what the hypoythesis function is computing. We can interpret Figure. 2.1 as follows: When the probability of y being 1 is greater than 0.5 then we can predict y = 1 else we can predict y = 0. Function g(z) is greater than or equal to 0.5 when z is greater than or equal to 0. Since $z = \theta^T x$, the hypothesis predicts y = 1 when $\theta^T(x) \geqslant 0$. We can build more complex decision boundaries by fitting complex parameters to this simple hypothesis.

2.2 Cost Function for Logistic Regression

The cost function calculates the cost the learning algorithm has to pay if the outcome is $h_\theta(x)$ and the actual outcome is y.[8] The cost function $J(\theta)$ is defined as follows:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x), y)$$

Here m is the number of training examples and each training example is a column vector of length $n + 1$.

Training set : $\{(x^1, y^1), (x^2, y^2), ..., (x^m, y^m)\}$; $x = \begin{bmatrix} x_0 \\ x_1 \\ ... \\ x_n \end{bmatrix}$

In the case of binary logistic regression, $y \in \{0, 1\}$. We can convert binary logistic regression to a multiclass classification algorithm by using the one-vs-rest strategy discussed in Chapter 1.The cost function defined above is a non-convex function for parameter optimization; i.e if we take $h_\theta(x)$ and plug it into the cost function, then plug the cost function into $J(\theta)$ and plot $J(\theta)$ we find many local optimum points.

The convex cost function can be defined as follows:

$$J(\theta) = \frac{1}{m}[\sum_{i=1}^{m} ylogh_\theta(x) + (1-y)log(1-h_\theta(x)]$$

If y = 1 and $h_\theta(x) = 1$; i.e, if hypothesis predicts exactly 1 and that is correct then that corresponds to 0. As $h_\theta(x)$ goes to 0, cost goes to ∞. If $h_\theta(x) = 0$ but $y = 1$, the cost function will penalize the learning algorithm with a massive cost.

2.3 Optimization Techniques for Cost Function

Machine learning is the study that gives computers the ability to learn and also the ability to think without being explicitly programmed. Machine learning algorithms are used for manipulating the data and predict the output for new data with high precision and low uncertainty. Optimization algorithms are used to make rational decisions in an environment of uncertainty and imprecision. The most common optimization technique is gradient descent.[9]

2.3.1 Gradient Descent

Gradient descent algorithms offer a very good perspective for solving problems related to data analysis. They are a class of algorithm that is used to minimize functions. Here we would like to minimize the cost function. Gradient descent repeatedly updates each parameter(θ_j) using a *learning rate* (α). The algorithm works as follows:

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^i) - y^i)x_j^i$$

}

If we have n features we have to update all the θ values (θ_0 to θ_n) simultaneously. This can

also be written as:

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

Here α is the learning rate and controls how big a step the algorithm can take. If α is big, gradient descent takes large steps, else it takes tiny steps.

2.3.2 Limitations of Gradient Descent

Gradient descent has the following limitations:

1. The gradient descent algorithm takes time to converge to the global optima based on the value of α chosen. The learning rate has to be chosen manually. The algorithm does not update the learning rate automatically.

2. The number of iterations should be chosen experimentally in the gradient descent algorithm and partial derivatives of the cost function must be calculated for each iteration.

Chapter 3

Proposed Model

This chapter introduces another optimization technique which overcomes the limitations discussed in Section 2.3.2

3.1 Conjugate Gradient Descent

The Conjugate Gradient Method is a first-order derivative optimization method for multi-dimensional nonlinear unconstrained functions.[10] It is related to other first-order derivative optimization algorithms such as gradient descent and steepest descent.

The information processing objective of the technique is to locate the extremum of a function. From a starting position, the method first computes the gradient to locate the direction of steepest descent, and then performs a line search to locate the optimum step size, α. The method then repeats the process of computing the steepest direction, computes direction of the search, and performs a line search to locate the optimum step size. A parameter β defines the direction update rule based on the gradient and can be computed using one of a number of methods. In most numerical methods, we find a direction and magnitude of change toward the minimum. Moving in that direction does not necessarily lead to convergence, though. Line search tells to walk to the minimum through that direction.

Conjugate gradient descent updates the values of α and β as follows:
Result: Updated values of α and β

Initialization: $\theta \in \mathrm{R}^n$;$\nabla J(\theta) = [\frac{\partial}{\partial \theta_j} J(\theta)]^T$; d = g = -$\nabla J(\theta)$

Repeat until convergence

$\alpha \leftarrow argmin_\alpha J(\theta + \alpha d)$;

$\theta \leftarrow \theta + \alpha d$;

$g^{'} \leftarrow g$;

$\beta \leftarrow max\{\frac{g^T(g-g^{'})}{g^{'T}g^{'}}\}$;

$d \leftarrow g + \beta d$;

Algorithm 1: Update rule for α and β

3.1.1 Advantages of Conjugate Gradient

The advantages of conjugate gradient descent over gradient descent are

1. Conjugate gradient descent does not require manual update of learning rate

2. It is often faster than simple gradient descent and also scales well for large data sets

3.2 Phases of Document Classification

The important phase in training the data for classification is feature selection.[11] It is the process of selecting a subset of the terms occurring in the training set and using only this subset as features in text classification. It makes training and applying a classifier more efficient by decreasing the size of the effective vocabulary. Feature selection often increases classification accuracy by eliminating noise features. The labels/classes are also fed to the learning algorithm.

The prediction involves feeding a new document to the classifier, which tries to categorize the data based on the experience it gained by updating a model on the basis of the training data.

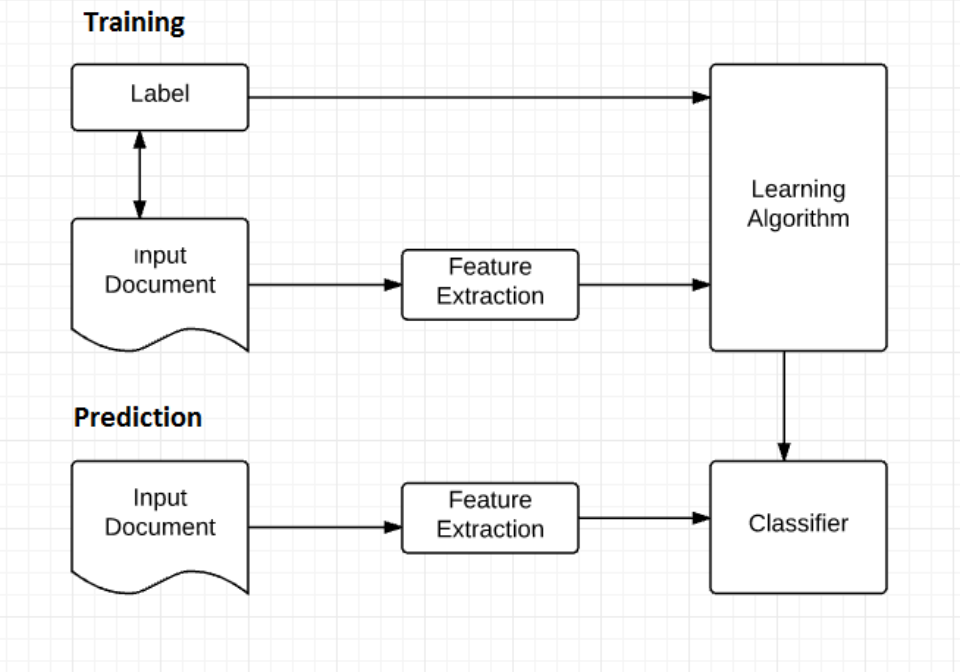

Figure 3.1: *Phases involved in document classification*

I used the logistic regression classification algorithm with a conjugate gradient descent optimization technique to classify the *20 Newsgroups* data set.

Chapter 4

Experiments

This chapter explains the data set used and the experiments designed to evaluate the approach used in this study. Experiments were conducted with various approaches mentioned in Section 2.1, Section 2.2 and Section 3.1

4.1 Data Overview

The *20 Newsgroups* data set is a collection of 18846 documents, partitioned evenly across 20 different newsgroups.[12] It was originally collected by Ken Lang. The data is organized into 20 different newsgroups, each corresponding to a different topic. Some of the newsgroups are very closely related to each other. Here is a list of the 20 newsgroups, partitioned according to subject matter.

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Figure 4.1: *The 20 Newsgroups data set partitioned according to subject matter*

4.2 Training, Cross Validataion and Test Datasets

Separating data into training and testing sets is an important part of evaluating classification models. Typically, we separate a data set into a training set, cross validation set, and testing set; most of the data is used for training, and a smaller portion of the data is used for cross validation and testing. The training set is used to build the model (determine its parameters) and the test set is used to measure its performance (holding the parameters constant).Cross validation set evaluates how the results will generalize to an independent data set (values in one sample reveal no information about those of the other sample). I divided *The 20 Newsgroups data set* into $80\% - 60\% - 20\%$, where 80% is training data, 20% is cross validation data and the remaining 20% is test data.

- Total Documents = 18846
- Training set = 11306
- Cross validation set = 3769
- Test set = 3769

4.3 Experimental Design

This section explains the design and implementation of experiments to evaluate logistic regression with conjugate gradient descent on the *20 Newsgroups* data set. The performance of this approach on the representative classification task is evaluated against those of existing gradient descent implementations.

4.3.1 Experimental Design for CGD

The algorithm discussed in Section 3.1 has been implemented in the library Spark MLlib. The output of running the classifier on test data is a confusion matrix. It is a table with two rows and two columns that reports the number of false positives, false negatives, true positives, and true negatives. These can be used to measure performance parameters such as precision, recall, and F-score.

4.3.2 Experimental Design for Gradient Descent

Approach 1: The data set has been evaluated using the `LiblineaR` package in R.[13] `LiblineaR` is a linear classifier for data with millions of instances and features. It supports multi-class classification using the one-vs-rest strategy for logistic regression.

Approach 2: The data set has been evaluated using `sklearn.linearmodel.LogisticRegression` package in the `scikit-learn` library. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the *multiclass* option is set to *ovr*.

Approach 3: The data set has been evaluated using `MATLABmnrfit` in MATLAB.[14] I used the `mnrval()` function to compute fitted probabilities, and then took the category with maximum probability.

Chapter 5

Results

This chapter explains the results for the experiments conducted in Section 4.3.1, and Section 4.3.2.

5.1 Logistic Regression with CGD

Table 5.1 shows the performance parameters (precision, recall and F-score) when logistic regression with conjugate gradient descent is performed on The *20 Newsgroups* data set.

Table 5.2 shows the average performance parameters (precision, recall and F-score) when logistic regression with conjugate gradient descent is performed on The *20 Newsgroups* data set. Many measurements of performance for classifiers and other systems exist in machine learning. This report focuses on precision, recall, and F-score. Precision is defined as the proportion of positively classified examples that are actually true positives. Recall, also known as sensitivity, is defined as rate of truly positive examples. F-measure is the harmonic mean of precision and recall. F-score (specifically, F1) gives a better measure of performance since there is often a tradeoff between recall and precision when data suffer from the class imbalance problem.[15]

Table 5.1: *Performance parameters for logistic regression with conjugate gradient descent*

Class Name	Precision	Recall	F-score
alt.athesim	0.95	0.93	0.94
comp.graphics	0.83	0.91	0.87
comp.os.ms-windows.misc	0.92	0.89	0.90
comp.sys.ibm.pc.hardware	0.84	0.86	0.85
comp.sys.mac.hardware	0.90	0.91	0.91
comp.windows.x	0.90	0.90	0.90
misc.forsale	0.89	0.90	0.89
rec.autos	0.96	0.93	0.94
rec.motorcycles	0.98	0.98	0.98
rec.sport.baseball	0.97	0.97	0.97
rec.sport.hockey	0.98	0.97	0.98
sci.crypt	0.97	0.96	0.97
sci.electronics	0.90	0.94	0.92
sci.med	0.93	0.94	0.93
sci.space	0.97	0.96	0.97
soc.religion.christian	0.95	0.94	0.95
talk.politics.guns	0.98	0.95	0.96
talk.politics.mideast	0.99	0.97	0.98
talk.politics.misc	0.94	0.95	0.95
talk.religion.misc	0.90	0.88	0.89

Table 5.2: *Average Performance parameters for logistic regression with conjugate gradient descent*

Performance Parameter	Average value
Precision	0.9359
Recall	0.9341
F-score	0.9344

Figure. 5.1 is a graphical representation of the average precision, recall and F-score values.

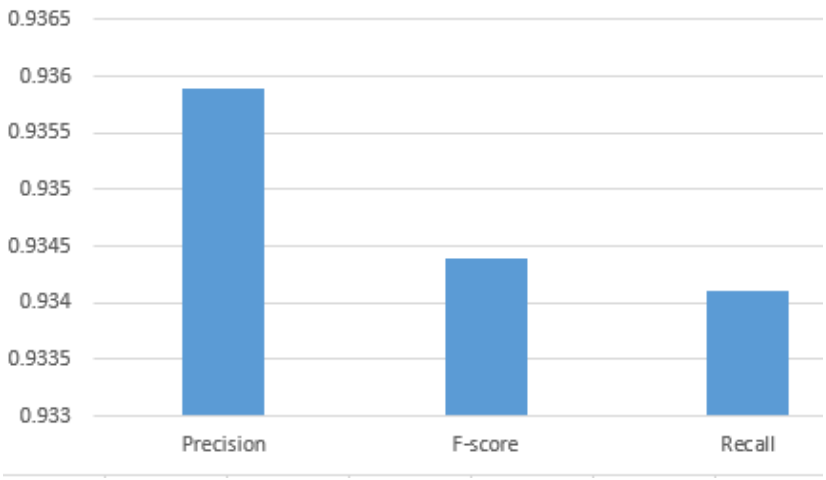

Figure 5.1: *Precision - Recall graph for logistic regression with conjugate gradient descent*

5.2 Logistic Regression with Gradient Descent

This section explains the results obtained for the approaches described in Section 4.3.2

5.2.1 Logistic Regression using `LiblineaR`

`LiblineaR` is a linear classifier for data with millions of instances and features. `LiblineaR` allows the estimation of predictive linear models for classification and regression, such as L1- or L2-regularized logistic regression, L1- or L2-regularized L2-loss support vector classification, L2-regularized L1-loss support vector classification and multi-class support vector classification.

Table 5.3: *Average Performance parameters for logistic regression using* `LiblineaR`

Performance Parameter	Average value
Precision	0.89
Recall	0.93
F-score	0.91

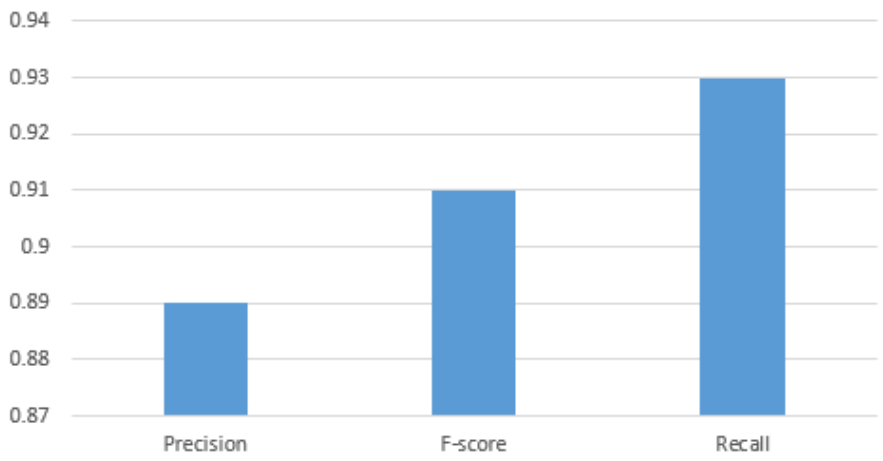

Figure 5.2: *Precision - Recall graph for logistic regression using* `LiblineaR`

5.2.2 Logistic Regression using `scikit-learn`

`scikit-learn` is a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems. This package focuses on bringing machine learning to non-specialists using a general-purpose high-level language. Emphasis is put on ease of use, performance, documentation, and API consistency. It has minimal dependencies and is distributed under the simplified BSD license. I used the `sklearn.linearmodel.LogisticRegression` package in this library.

Table 5.4: *Average Performance parameters for logistic regression using* `scikit-learn`

Performance Parameter	Average value
Precision	0.86
Recall	0.87
F-score	0.864

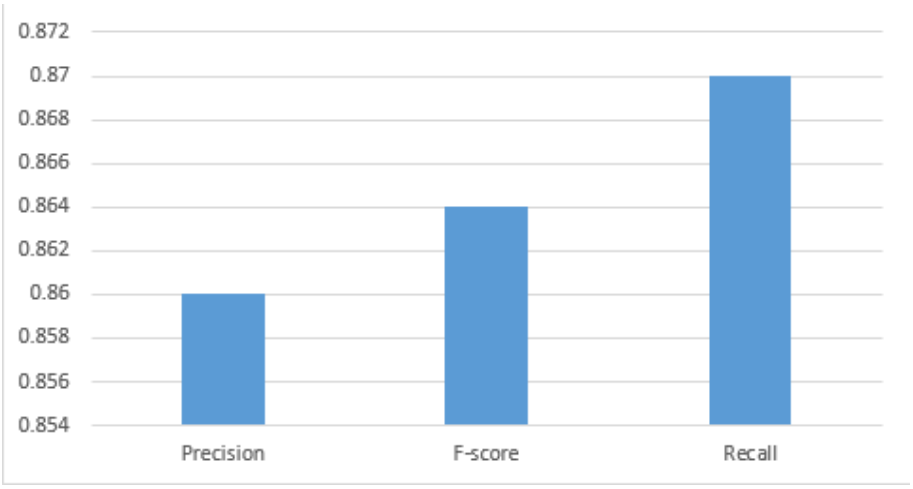

Figure 5.3: *Precision - Recall graph for logistic regression using* `scikit-learn`

5.2.3 Logistic Regression using `MATLABmnrfit`

A nominal response model is an extensions of the binary logit model and is called a multinomial logit model. The default link function `mnrfit` used for ordinal categories is the logit link function. The link function relates the mean of the response to the linear predictors in the model. This models the log cumulative odds. The 'link','logit' name-value pair specifies this in mnrfit.

Table 5.5: *Average Performance parameters for logistic regression using* `MATLABmnrfit`

Performance Parameter	Average value
Precision	0.8292
Recall	0.82
F-score	0.83

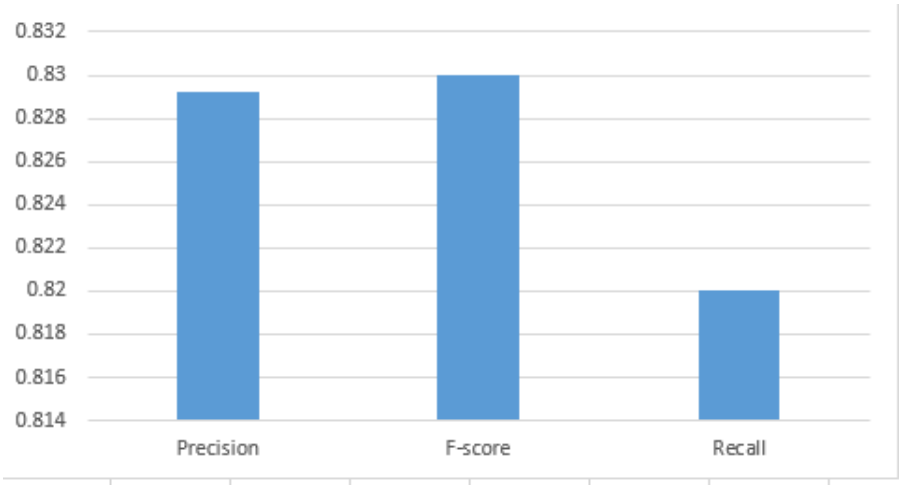

Figure 5.4: *Precision - Recall graph for logistic regression using* `MATLABmnrfit`

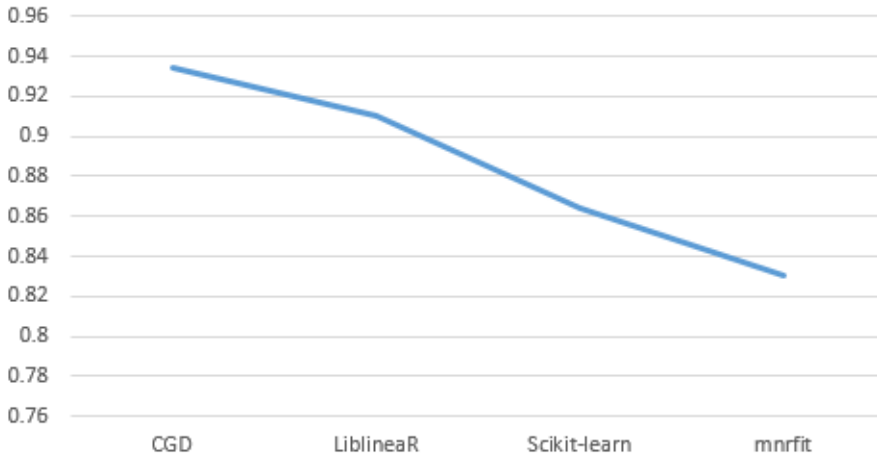

Figure 5.5: *Comparison of F-scores for all approaches*

Figure. 5.5 shows that conjugate gradient descent outperforms gradient descent implementations.

Chapter 6

Conclusion & Future Work

This chapter draws conclusions, and also addresses limitations of these approaches.

6.1 Conclusion

Most implementations of logistic regression in packages such as `LiblineaR`, `scikit-learn`, and `Mnrfit` use gradient descent as their optimization technique. However, experimental results show that conjugate gradient descent outperforms gradient descent because the method first computes the gradient to locate the direction of steepest descent, then performs a line search to locate the optimum step size, α. The method then repeats the process of computing the steepest direction, computes direction of the search, and performs a line search to locate the optimum step size.

6.2 Future Work

Advanced optimization techniques such as stochastic gradient descent or the Limited-Memory BroydenFletcherGoldfarbShanno Algorithm (L-BFGS) can be implemented instead of gradient descent as they are usually much more stable to train and easier to check for convergence. These methods enjoy parallelism by computing the gradient on GPUs. These methods, conventionally considered to be slow, can be fast, thanks to the availability of large amounts

of RAMs, multicore CPUs, and compute clusters with fast network hardware. MapReduce-style parallelism is an effective mechanism for scaling up performance to larger document corpora and other data sets. MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key.[16]

Bibliography

[1] Matthieu Cord Padraig Cunningham and Sarah Jane Delany. Machine learning for Multimedia. *Case Studies on Organization and Retrieval*, XVI, 2008.

[2] Shipeng Yu Vikas C. Raykar, Linda H. Zhao. Learning From Crowds. *Journal of Machine Learning Research*, 2010.

[3] Ryan Rifkin. Multiclass Classification. 2006.

[4] Sung-Bae Cho Jin-Hyuk Hong. A probabilistic multi-class strategy of one-vs.-rest support vector machines for cancer classification. 2008.

[5] Mitchel Klein David G. Kleinbaum. Logistic Regression: A Self-Learning Text. 3.

[6] Chilukuri K. Mohan Sanjay Ranka Anil Ravindran Menon, Kishan Mehrotra. Characterization of a Class of Sigmoid Functions . *Computer Sciences Commons*.

[7] Benjamin Tovar Cisneros.

[8] Andrew Ng. Courseera: Machine Learning. 2009.

[9] Using Gradient Descent for Optimization and Learning. Nicolas Le Roux. 2009.

[10] An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Jonathan Richard Shewchuk. 1994.

[11] An Introduction to Variable and Feature Selection. Isabelle Guyon, Andre Elisseeff. *Journal of Machine Learning Research*, 2002.

[12] Ken Lang. The 20 Newsgroups dataset. 2008.

[13] Cho-Jui Hsieh Xiang-Rui Wang Chih-Jen Lin Rong-En Fan, Kai-Wei Chang. LIBLINEAR: A Library for Large Linear Classification. 2008.

[14] P. McCullagh and J. A. Nelder. Generalized Linear Models. 1990.

[15]

[16] Jeffrey Dean and Sanjay Ghemawat.

[17] Apache spark. URL http://spark.apache.org/examples.html.

[18] Mllib. URL http://spark.apache.org/mllib/.

Appendix A

Table of notations

Table A.1: *Table of notations*

Symbol	Definition
n	number of features
m	number of training examples
x	feature vector
y	target variable
(x, y)	single training example
(x^i, y^i)	i^{th} training example
$h_\theta(x)$	hypothesis function which maps from x to y
x_j^i	value of feature j in ith*trainingexample*
j(θ)	cost function
θ	parameter vector
α	learning rate
d	descent direction
β	direction update rule

Appendix B

Technologies

Apache Spark is a powerful open source processing engine built around speed, ease of use, and sophisticated analytics.[17] It was originally developed at UC Berkeley in 2009.

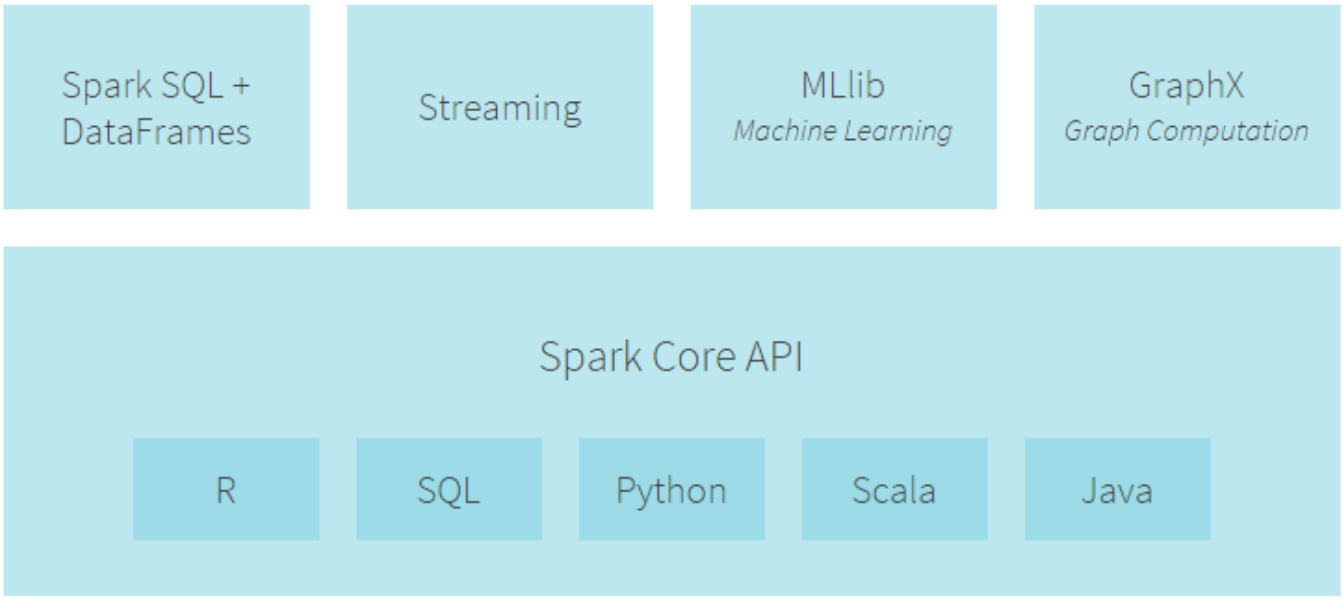

Figure B.1: *Spark Ecosystem. Adapted from Databricks*

Apache Spark is a fast and general-purpose cluster computing system. It provides high-level APIs in Java, Scala and Python, and an optimized engine that supports general execution graphs. An execution graph describes the possible states of execution and the states

between them. Spark also supports a set of higher-level tools including Spark SQL for SQL and structured data processing, MLlib for machine learning, GraphX for graph processing, and Spark Streaming.

MLlib is Spark's machine learning library, focusing on learning algorithms and utilities, including classification, regression, clustering, collaborative filtering, dimensionality reduction, as well as underlying optimization primitives.[18]