Technisch Wissenschaftliches Arbeiten

State of The Art

Literature Research Report

for the Bachelor of Science in Applied Computer Science
at the Cooperative State University Stuttgart

by

Paul Jaehne **Andreas Jauch**

April 2013

Student ID, Course	4488174, TIT10AIB
	6114006, TIT10AIB

1 Introduction

In our study thesis at Cooperative State University, Stuttgart, we use text classification to detect whether people are alcohol intoxicated or sober. Generally the state of the art and the literature research, focus on the topic of text classification. As classification is a very huge topic as a whole we picked one state of the art classification algorithm - logistic regression - which is described in detail. The base data was the so called Munich Alcohol Language corpus, which is therefore briefly introduced as well.

2 Literature Research

2.1 Text Classification with Logistic Regression

The following steps were taken to find relevant information on the topic of text classification and logistic regression.

1. Watch some online lectures of the Stanford *Natural Language Processing* class to get a basic understanding of text classification.
2. Watch all online lectures on *Machine Learning* from Stanford University available on Coursera.
3. Search the internet using normal web search on Google with special focus on lecture notes and university presentations.
4. Search literature databases Springer Link and ACM Digital Library.
5. Search published conference papers from IEEE Xplore.
6. Read some of the papers shipped with the alcohol language corpus.

First we watched the *text classification* section of the Stanford Natural Language Processing class [12]. Afterwards we watched the *Maschine Learning* [17] online class from Stanford to get a fundamental and detailed understanding of maschine learning basics and to understand the details of logistic regression. The advantage of this was that the material was already well prepared for beginners and fitted the problem quite well, so the general understanding of text classification and the logitstic regression algorithm could be build up from these two ressources. For a more detailed understanding a literature research on all the involved subtopics had to be performed. We started searching the ACM Digital Library, IEEE Xplore as well as Springer Link for useful papers. Unfortunately the found papers were either too constrained on particular applications, e.g. medical application etc., or they were not accessible for free. Thus the search results were valuable because we could find some of the found papers by their title and author using Google. Here Google's function to restrict

searches to certain filetypes was very useful. Generally it is to say that the IEEE Xplore databse delivered the most valuable results like [14]. Luckily for us there were some useful presentations and lecture notes (e.g. [20], [16], [1], [10], [4]) from universities which were not as constrained as the papers. Therefore they build the biggest group of used ressources.
The following list shows the main substopics and thus served either alone or in combination as main search terms:

[formal definition of classification, feature extraction, feature selection, information gain, decision boundary, gradient decscent, logistic regression, classification cost function, cross validation, overfitting]

2.2 Alcohol Language Corpus

To research information on the Munich Alcohol Language Corpus the corpus website [8] provides a potentially complete list of publications about the corpus ([6], [7]). Hence further literature search was not required.

Text Classification using Logistic Regression

Detecting Alcohol Intoxication in Speech

State of the Art

Andreas Jauch Paul Jähne

University of Cooperative Education, Stuttgart
AJAUCH@de.ibm.com
PJAEHNE@de.ibm.com

Abstract

Classfication describes the problem of determining the class an object or unit of its kind belongs to. This paper describes the general approach of classifiers, beginning with fundamentals, data representation and features and going more into detail with information gain and training of classifiers. Special focus is set on logistic classification, which will be explained very detailed including the underlying functions. The basis used for all classification throughout this paper is the Munich Alcohol Language Corpus, which will also be briefly explained.

Classification generally describes the "systematic arrangement in groups or categories according to established criteria" [17]. Generally applicable classification algorithms are learning algorithms, most of them are supervised machine learning algorithms. These types of algorithms learn from existing sample data how to classify new, unseen samples. Therefore a collection of sample data is required - a so called corpus. In the corpus each sample, called an instance, is labled with the corresponding class. Thus a corpus is a collection of labelled instances.
The corpus used as basis for the classification described in this article is called *Munich Alcohol Language Corpus*. Because of the fact that classification is performed on transcribed speech, this kind of classification is called *text classification* - a kind that is used rather often [20]. Throughout the paper, classification is equivalent to text classification, as this paper only concentrates on this topic.

1 The Munich Alcohol Language Corpus

The *Munich Alcohol Language Corpus* (ALC) is a german speech corpus, which contains intoxicated and sober recordings of 162 male and female speakers from Germany. The idea behind this corpus is to establish new methods to test whether a driver is intoxicated or sober [5]. Each test person did 60 recordings in sober state and an additional 30 ones while being intoxicated. In addition to these 14580 recordings another 600 sober recordings have been done, to provide some data for control purposes. Thus the number of transcripts is 15180, with a distribution of approximately $\frac{1}{3}$ intoxicated transcripts and $\frac{2}{3}$ sober ones.
There are different types of speeches that are recorded, but every one of them has something

to do with driving. The tests consist amongst others of reading an address, reacting to a spontaneous command or describing a picture. A complete list of tests is displayed below.

LN	list numbers
LT	list tongue twister
LS	list spelling
RT	read tongue twister
RR	read read command
RA	read address
DQ	dialogue question
DP	dialogue picture
MQ	monologue question
MP	monologue picture
EC	elicited command

Table 1: list of tests [6]

Apart from the transcribed speech, each record documents additional information such as irregularities, meta information concerning the test, or speaker information. The following example of a tongue twister shall describe the transcription:

```
die K"ochin mit dem Tupfenkopftuch kocht
Karpfen in dem Kupferkochtopf
```

Listing 1: sober trancript

```
die K"ochin mit dem -/#Tufenk/- -/#tu/-
-/#topf/- <"ah> -/#Tupfenkoch/- <P>
Tupfenkopftuch kocht Karpfen in dem
Kupferkochtopf
```

Listing 2: intoxicated transcript

In the transcripts all irregularities are marked by /#, whereas words that do not belong in that place are enclosed by angel braces (e.g. <"ah>).

2 Classfication Fundamentals

While the last section introduced the general problem of classification and the used corpus, this section focuses on classification algorithms and their application. Using classification in a program usually involves four major steps: extracting the data from given sample input, preprocessing or reworking this data, training and optimization of a classifier for this kind of data and finally measuring the accuracy of the just created classifier with unseen data.

2.1 Classification Algorithm

A classifier Y is a function that is defined from the so called feature space into the class space

$$Y : X \rightarrow C.$$

Its purpose is finding the most probable class $c \in C$ for a given sample $x \in X$. The given set of classes C consists of at least two classes. If there are only two classes, one speaks of *binary* classification whereas more than two classes fall under the definition of *multinomial* classfication tasks [2, p. 181ff].

2.2 Data Representation and Features

In programs, classifier implementations usually follow a similar approach: as a first step *features* have to be extracted for the object to classify. A feature represents one piece of relevant information for a given task. Usually several features are extracted from a given text which are combined to a feature vector. This can for instance represent the presence of certain relevant words, the average number of words per sentence or other things that can be extracted from the text. This constructed vector of features is the classification algorithm's input data.

2.3 Feature Selection and Extraction

Good accuracy in classification heavily depends on the used features. Extracting features from a text usually leads to a giant amount of features out of which many are not carrying relevant information for classification or cause the computational requirements to explode. If one would i.e. have to determine the topic of newspaper articles one might build up the features of the first version using the *bag of words* approach. Hereby the feature vector contains a binary value for every word occuring in any of the training samples indicating the presence of each word in a specific newspaper article. This leads to a tremendous amount of features. Hence the process called *feature selection* is the point where the art of classifier application begins. Feature selection is about selecting the most valuable features from a given set of features. This can be done manually but is usually done by one or more programmatic methods.

2.3.1 Information Gain

An established but yet state of the art approach for feature selection is called *information gain*. This method is based on Shannon's Theory on information theory and entropy [7, p. 176ff]. In information theory entropy measures the uncertainty of a random variable. To state a concrete example, the random variable X could represent the toss of a coin. If the coin is equally balanced between head and tail, meaning that

$$p(X = tail) = p(X = head) = 0,5$$

then knowing the outcome of this experiment is a valuable information. Therefore as the uncertainty on the outcome of this experiment is high, the entropy is high. If the coin would be unbalanced and say the probability of head is much higher than the probability of tail, the corresponding entropy decreases. To spin this example further, if a coin had two heads, the outcome of the random experiment would be entirely predictable, so the corresponding entropy decreases to 0 [9]. Entropy is usually measured in bit [3].

[1]and the feature X is a discrete variable

Information gain applies this theory to classification problems. Given a feature X, $I(Y;X)$ measures the degression of uncertainity about the class variable Y given the feature X is applied to it. Its first component is given by the entropy, or uncertainty, of the class variable Y. Stated that the class variable Y resides in the range $y \in \{y_1, ..., y_k\}$, a text is classified in k different classes. Thus the entropy is given by [15]

$$H(Y) = -\sum_{i=1}^{k}[P(Y = y_i) \cdot log(P(Y = y_i))]$$

To compute the information gain of feature X, the uncertainty about the class variable Y is reduced by a certain amount if the realization of the feature X is known. The remaining uncertainty after X became known is represented by the conditional entropy $H(Y|X)$. Assuming the training set contains l different values on the feature X[1] this conditional entropy term is given by

$$H(Y|X) = -\sum_{j=1}^{l} P(X = x_j) \cdot H(Y|X = x_j)$$

$$H(Y|X) = -\sum_{j=1}^{l}\langle P(X = x_j) \cdot \sum_{i=1}^{k}[P(y_i|x_i) \cdot log(P(y_i|x_i))]\rangle$$

Finally these two values are subtracted to measure the information that has been gained:

$$I(Y;X) = H(Y) - H(Y|X)$$

In order to use this measurement for feature selection a threshold value can be defined. In this case the feature vector is reduced by those components that are below the threshold.
Information gain can generally be applied any time before beginning with the training procedure of a classifier, because it is completely independant from the chosen classifier.

2.4 Training Process

The training or learning process of a classifier aims at finding an optimal configuration of the classifier's parameters. These parameters can be combined into one vector, often called the parameter vector θ. Every value in this vector can be seen as a weight that is related to the corresponding value in the feature vector. Hence the classifier trains to find the optimal weights for all given features.

One important illustration of the learning process of a classifier is the so called *decision boundary*. It partitions the underlying feature space into as many sets as there are classes. Depending on which side of the boundary a sample is located, the classifier makes its decision [18]. The decision boundary can be either linear or non-linear, and it can be of arbitrary complexity depending on the type of classifier used and the parameters learned [11, p.9f]. The straight line partitioning the set of data into two classes that [*as shown in Figure 1*] constructs a very simple example for a linear decision boundary.

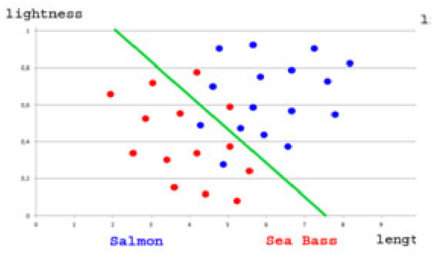

Figure 1: basic decision boundary example [19]

For many classifiers this process of training can be seen as a minimization process of an underlying *cost function*. This parameterized function measures the average distance between the predicted values of the training samples to the samples' real (or labeled) values. The parameter θ_{min} that minimizes the cost function indicates that setting the classification algorithm gives the most accurate results.

If one would give a classifier all of the available labeled instances for training, the resulting classifier could perfectly adopt to these samples. Though there is no guarantee that the classifier will perform well on unseen instances. This problem is called *overfitting* [13]. To prevent a classifier from overfitting to a static training set, a given set of labeled instances is partitioned into a training set and a test set. The training set is used to train the classifier and the test set is used to measure the accuracy achieved. This partitioning can be fix, e.g. 80% / 20%, however in most application a more dynamic method is used. Thereby the data is partitioned into k subsets. In the first run the first subset is used as test data and the remaining $k-1$ subsets are used to train the classifier, in the second run the second subset is used as test data while the first and the last $k-2$ subsets are used to train the classifier and so on. This leads to a method called *k-fold cross validation* which gives a more accurate performance measure of the trained classifier [10].

3 Logistic Regression

A famous and widely used classification algorithm is logistic regression which is among others used in this study work to determine if a person is intoxicated or not. Most other algorithms used are far too complicated to be described in a short paper, thus logistic regression is taken as an example to explain some of the key concepts of classification algorithms in more detail.

3.1 Hypothesis and the Sigmoid Function

The logistic regression algorithm is based on a hypothesis of the form

$$0 \leq h_\theta(x) \leq 1$$

stating the probability that a given sample s characterized by its n-dimensional feature vector x is of the positive class from the set of classes C. In a binary classification the two classes are assigned to be either the positive class or the negative class. Projected on the ALC corpus mentioned on page 3 this would mean that $C = \{alc, nonalc\}$, with $y = alc$ representing the positive class. As feature vector x one can think of a column vector in which each line represents a word from the corpus and the number indicates the presence of the word in the given sample. The parameter vector θ represents the parameters, which the algorithm shall learn during the training process. Hence these parameters are to be optimized to achieve the highest possible accuracy. In probability theory the statement of the given hypothesesis can be written as follows:

$$h_\theta(x) = P(y = alc|x;\theta)$$

This implies that the probability for the given sample x to be of the other (in this case negative class) can be easily determined by

$$P(y = nonalc|x;\theta) = 1 - h_\theta(x)$$

In a final step these thoughts can be turned into a true binary classifier if the hypothesis is wrapped into a classification function Y from the feature space to the class space

$$Y : X^n \to C$$

$$Y(x) = \begin{cases} 1 & \text{if } h_\theta(x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

While the theory presented so far can be applied to other monotonically nondecreasing functions, too, the heart of this classifier lays in the hypothesis function $h_\theta(x)$. The function used in logistic regression is called the *logistic* or *sigmoid* function and is shown in *Figure 2* [13].

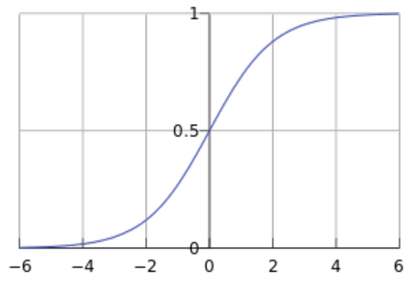

Figure 2: logistic / sigmoid function

As an asymptotic function bounded by 1 on the positive and 0 on the negative side it suits well as foundation for a classifier. An algebraic definition of the sigmoid function is given by

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

This formula illustrates the asymptotic behaviour as for $x \to \infty \quad h_\theta(x)$ comes close to 1 and for x approaching negative infinity $h_\theta(x)$ comes close to 0.

3.2 Training Process and Cost Function

In order to use these constructions as a working and well performing classifier the parameter θ needs to be trained. As described above, training means minimizing a certain cost function that models the distance between the measured value of a sample and the value predicted by the classifier. This cost function is then minimized in the next step. *Figure 3* shows a very easy example that uses a one dimensional feature vector x, which corresponds to a single feature giving it a one dimensional weight paramether θ.

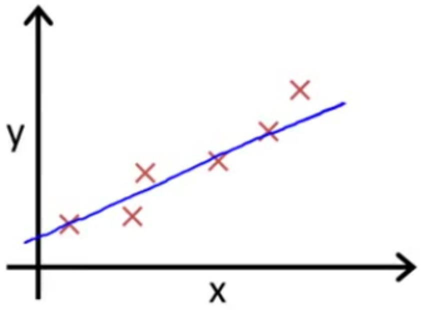

Figure 3: example of a one dimensional feature, with sample values and decision boundary [14]

The y-axis refers to the set of available classes. The red crosses show several labeled training instances. During the training process this classifier has been trained to predict the blue line as predicted output for y. So the blue line represents $h_\theta(x)$. An easy yet suitable approach could measure the sum of the individual distances each training sample has to the blue line [13, p. 4]. Given the number of training samples is m, this leads to the following simple cost function, called the *squared error function*:

$$cost(\theta) = \frac{1}{m} \cdot \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

This approach can easily be transferred to higher order feature and parameter vectors.

3.3 Cost Function for Logistic Regression

However if you plug in the sigmoid function (which is used by the logistic regression algorithm as its hypothesis) into this squared error cost function the result is a non-convex function [16].

$$cost(\theta) = \frac{1}{m} \cdot \sum_{i=1}^{m} (\frac{1}{1+e^{-\theta x_i}} - y_i)^2$$

The properties but also the problem of a non-convex function is illustrated in *Figure 4(a)*.

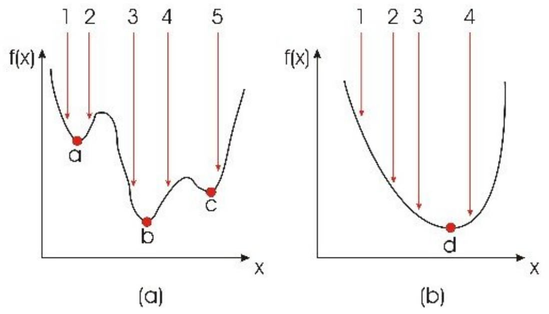

Figure 4: (a) non-convex (b) convex function [4]

While a convex function has global minimum that is easy to find, a non-convex function has several local minima and maxima. Thus minimizing this sort of functions is much harder than minimizing a convex function. Therefore this type of cost function is not suitable for logistic regression.
However, using the logarithm function gives a suitable cost function for logistic regression, but it uses a different approach [13, p. 18].

$$scost(x_i; y_i; \theta) = \begin{cases} -log(h_\theta(x_i)) & \text{if } y_i = 1 \\ -log(1 - h_\theta(x_i)) & \text{if } y_i = 0. \end{cases}$$

The idea behind this $scost$ function is to penalize values of $y_i = 1$ less than other values, if the predicted output $h_\theta(x_i)$ is close to 1. Conversely this means that the penalty increases evermore, the nearer $h_\theta(x_i)$ comes to 0 [1]. This works because the more $h_\theta(x_i)$ is converging towards 1, the more the cost function converges towards 0 and thus this choice of θ is associated with a 0 penalty.

$$\lim_{h_\theta(x_i)\to 1} = -log(h_\theta(x_i)) = -log(1) = 0$$

On a negative example where the classifier predicts a hypothesis being close to 0 this leads to cost increasing up to infinity. A similar argument can be given for the case of $y_i = 0$. $Scost$ measures only the cost of the i-th predicted sample. Finally *Figure 5* shows what this function looks like.

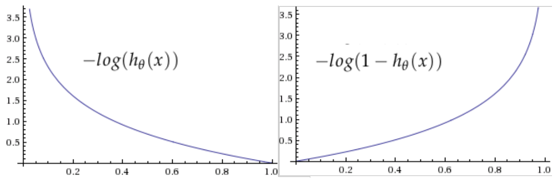

Figure 5: log-based cost function (l) y = 1 (r) y = 0

In order to turn this into a cost function for all given training samples, this function can be plugged into the same framework as described for the squared error function:

$$cost(\theta) = \frac{1}{m} \cdot \sum_{i=1}^{m} scost(x_i; y_i; \theta)$$

The final step to finish the training process is to minimize $cost(\theta)$.

$$\min_{\theta}[cost(\theta)]$$

If this function uses higher order vectors of x and θ, it can be minimized by using an algorithm called *gradient decent* [12] [8] that slowly converges towards the minimum. Anyway, the description on how to minimize this function goes beyond the scope of this paper and can be found in the literature.

References

[1] Jacky Baltes. Machine learning linear regression. presented at University of Manitoba, Canada, available at `http://www4.cs.umanitoba.ca/~jacky/Teaching/Courses/COMP_4360-MachineLearning/current/Lectures/L03_Regression.pdf`. [Online; accessed 19-Mar-2013].

[2] C.M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[3] L. Brillouin. *Science And Information Theory*. Dover phoenix editions. Dover Publications, Incorporated, 1956.

[4] Bram Demeulenaere. Convex optimization. `http://people.mech.kuleuven.be/~bdemeule/convex.html`, 2008. [Online; accessed 19-Mar-2013].

[5] Sabine Barfuesser Florian Schiel, Christian Heinrich. *Alcohol Language Corpus*. Springer New York Heidelberg, 2011. Language Resources and Evaluation.

[6] Bavarian Archive for Speech Signals. Corpus documentation. available at `http://www.phonetik.uni-muenchen.de/Bas/BasALCREADME`, 2008. [Online; accessed 27-Mar-2013].

[7] Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lotfi A. Zadeh. *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[8] Kris Hauser. Gradient descent. available at `http://homes.soic.indiana.edu/classes/spring2012/csci/b553-hauserk/gradient_descent.pdf`. [Online; accessed 27-Mar-2013].

[9] S. Ihara. *Information theory for continuous systems*. Series on Probability and Statistics Series. World Scientific Publishing Company, Incorporated, 1993.

[10] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. pages 1137–1143. Morgan Kaufmann, 1995.

[11] Chulhee Lee and D.A. Landgrebe. Decision boundary feature extraction for neural networks. *Neural Networks, IEEE Transactions on*, 8(1):75–83, Jan.

[12] A. Meister. *Numerik linearer Gleichungssysteme*. Vieweg, 2005.

[13] Andrew Ng. lecture notes on supervised learning. `http://cs229.stanford.edu/`

notes/cs229-notes1.pdf. [Online; accessed 19-Mar-2013].

[14] Andrew Ng. Cost function. a video from an online class at Stanford University, available at https://class.coursera.org/ml/lecture/download.mp4?lecture_id=6, 2012. [Online; accessed 19-Mar-2013].

[15] Thomas D. Schneider. Information theory primer. available at http://schneider.ncifcrf.gov/paper/primer/. [Online; accessed 30-Mar-2013].

[16] Igmar Schuster. Classification using logistic regression. presented at University of Leipzip, Germany, available at hhttp://asv.informatik.uni-leipzig.de/uploads/document/file_link/530/TMI05.2_logistic_regression.pdf. [Online; accessed 19-Mar-2013].

[17] Merriam Webster. classification. available at http://www.merriam-webster.com/dictionary/classification. [Online; accessed 27-Mar-2013].

[18] wikipedia.org. Decision boundary. available at http://en.wikipedia.org/wiki/Decision_boundary. [Online; accessed 27-Mar-2013].

[19] Philipp Wintersberger. Phonegestik erkennen: Kuenstliche intelligenz fuer alle! available at http://www.codefest.at/post/2013/03/01/Teil-2-Phone-Gestik-erkennen-\Kunstliche-Intelligenz-fur-Alle.aspx. [Online; accessed 19-July-2008].

[20] Y. Yang and T. Joachims. Text categorization. *Scholarpedia*, 3(5):4242, 2008.