Machine Learning with Domestic Energy Use Data

Sam Stern (s1134468)

March 24, 2015

Abstract

Contents

	E	D	C2	C1	B	A
E	2.00	0.00	0.00	1.00	1.00	0.00
D	0.00	1.00	1.00	2.00	1.00	0.00
C2	0.00	1.00	21.00	9.00	1.00	0.00
C1	1.00	4.00	7.00	27.00	4.00	0.00
B	0.00	0.00	1.00	8.00	4.00	1.00
A	0.00	0.00	2.00	3.00	1.00	0.00

Chapter 1

Models

1.1 Overview

There are several classification algorithms that can be used to perform supervised learning tasks and vary in their computational complexity, implementation and assumptions that they make about the distributions of the data [1]. Three well known methods are used to classify the data: Logistic regression, random forrest and k-nearest neighbour.

All three methods are examples of discriminative classifiers. The discriminative approach is appealing in that it it directly models $p(y|\mathbf{x})$. Also, density estimation for the class-conditional distributions is a hard problem, particularly when $\mathbf{x}$ is high dimensional, so if we are just interested in classification, then the generative approach may mean that we are trying to solve a harder problem than we need to[2].

1.2 Logistic Regression

For a binary classification problem $y \in \{0, 1\}$, such as discriminating between households with children ($y = 1$) and households without ($y = 0$), the logistic regression model learns a weight vector $\mathbf{w}$ such that given some new household with feature vector $\mathbf{x}$, the posterior probability of that household being in class, $p(y = 1|\mathbf{x}) = g(\mathbf{x}; \mathbf{w})$ where $g(x)$ is the logistic (or sigmoid) function.

$$g(\mathbf{x}; \mathbf{w}) = \sigma(\mathbf{x}; \mathbf{w}) = \frac{1}{1 - e^{-(b+\mathbf{w}\cdot\mathbf{x})}}$$

There are numerous advantages to using logistic regression for the household classification task. Firstly, logistic regression is interpretable. After the model has been trained and the weight vectors established, they can be used to determine how important each feature is to the classifier. Secondly, the confidence of a prediction can be inferred, resulting in interpretable results. There are, however, also drawbacks to logistic regression. Since the maximum likelihood function does not have a closed form solution, an iterative process must be used instead to learn the weights, which is not guaranteed to converge.

1.2.1 Multi-class Logistic Regression

To extend the problem of logistic regression to the multi-class case, often times the the *softmax* is used as a generalisation of the logistic function (σ), the predicted class of an instance is then given by

$$P(y = Y_i|\mathbf{x}) = \frac{\exp^{-(b_i+\mathbf{w}_i\cdot\mathbf{x})}}{\sum_{j=0}^{J}\exp^{-(b_j+\mathbf{w}_j\cdot\mathbf{x})}}$$

Although this is a valid method of classifying the data, it fails to acknowledge the ordinal property of the classes and assumes the data to be nominal. Ideally we would be able to build a model that exploits the fact that some classes are more similar than others. For example, if the true label of a household is B, then we would rather misclassify the instance as A or C1 than as D or E. Luckily, ordinal logistic regression (or ordered ligit) can be used to build a model that incorporated the ordering of the classes.

Using McCullagh's proportional odds model [3], p_i is defined as the proportion of instances that are in class i, then the ordered logit model has the form

$$\begin{aligned}
logit(p_1) &= \log(\frac{p_1}{1-p_1}) = b_1 + \mathbf{w}\cdot\mathbf{x} \\
logit(p_2) &= \log(\frac{p_1+p_2}{1-p_1-p_2}) = b_2 + \mathbf{w}\cdot\mathbf{x} \\
&\vdots \\
logit(p_6) &= \log(\frac{p_1+p_2+p_3+p_4+p_5+p_6}{1-p_1-p_2-p_3-p_4-p_5-p_6}) = b_6 + \mathbf{w}\cdot\mathbf{x}
\end{aligned}$$

The model assumes *proportional odds* which is that each explanatory variable exerts the same effect on each cumulative logit. This translates to the assumption that the weights are the same for each cutoff, but rather the classes have different intercepts b. A pragmatic description of regression models for ordinal data in the context of machine learning is given by Herbrich et. al. [4].

1.2.2 Implementation

To build the binary logistic regression classifier, MATLAB's fitglm tool which fits a generalised linear model to the data. To make the resulting model logistic, a binomial distribution and logit link function were specified. For the case of multi-class classification, two models were created. One McCullagh's proportional odds model (treating the data as ordinal) and one treating the data as nominal. Both methods were implemented in matlab using the mnrfit tool

1.3 Random Forest

Random Forest is a classification method that grows an ensemble of decision trees from a set of training instances and determines the class of a new instance by allowing the trees in the ensemble to vote on the most popular class. For N training sets and M features, each tree is grown by:

- Randomly sample n training instances from the N training with replacement (this will be the training sample to grow the tree).
- At each node, m features are selected at random (where $m < M$). The best of the m features is used to split the node.
- The trees are grown to the largest possible size (no pruning takes place).

A new instance is then classified by running it through each tree, allowing each of the trees to assign the instance a class. The predicted class of the test instance is then taken given by the vote of each tree.

Although (in contrast to building a single decision tree), it is not easy to visualise a random forest, it is still possible to gain an estimate of the variables that are most important for classification and can be used on data sets with a large number of features (see section **??**). Random forests have been shown to perform particularly well on unseen data compared to other classification methods as they avoid overfitting by only ever looking at a random subset of features and data [5].

1.3.1 Implementation

MATLAB's builtin treeBagger class was used to build the random forest. Because bootstrap aggregation is used to randomly sample the training data, the out-of-bag estimates were used to optimise the model's parameters, instead of using cross-validation. The parameters to optimise are the number of trees and size of features m to consider for splitting each node.

1.4 K-Nearest Neighbour

K-nearest neighbor is a fundamental method for classification as it is intuitive and requires little *a priori* knowledge about the data. It is a non-parametric model that classifies an unlabeled input by finding the k-nearest training points in feature space, using the classes of the nearest points to predict the class of the unlabeled point [6].

1.4.1 Implementation

MATLAB's fitknn tool was used to build a nearest neighbour classification model and the optimum parameters were found using 5-fold cross-validation. The parameters to find were the distance measure, search method and k (the number of neighbours).

Bibliography

[1] Christian Beckel, Leyna Sadamori, Thorsten Staake, and Silvia Santini. Revealing household characteristics from smart meter data. *Energy*, 78:397–410, 2014.

[2] Carl Edward Rasmussen and Christopher K. I Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[3] Peter McCullagh. Regressuib models for ordinal data. *Journal of the Royal Statistical Society*, 1980.

[4] Klaus Ob ermayer Ralf Herbrich, Thore Graep el. Regression mo dels for ordinal data: A machine learning approach. Technical report, Technical University of Berlin, 1999.

[5] leo Breiman. Random forests. *Machine learning*, 2001.

[6] Leif E. Peterson. K-nearest neighbor, 2009.