Extending Classifiers for Incomplete Data

A dissertation submitted in partial fulfillment of
the requirements for the degree of
BACHELOR OF ENGINEERING in Computer Science
in
The Queen's University of Belfast
by

Christopher McKee

INCLUDE DATE PLS

February 20, 2017

SCHOOL OF ELECTRONICS, ELECTRICAL ENGINEERING and COMPUTER SCIENCE

CSC3002 – COMPUTER SCIENCE PROJECT

Dissertation Cover Sheet

A signed and completed cover sheet must accompany the submission of the Software Engineering dissertation submitted for assessment.

Work submitted without a cover sheet will **NOT** be marked.

Student Name: Student Number:

Project Title:

Supervisor:

Declaration of Academic Integrity

Before signing the declaration below please check that the submission:

1. Has a full bibliography attached laid out according to the guidelines specified in the Student Project Handbook
2. Contains full acknowledgement of all secondary sources used (paper-based and electronic)
3. Does not exceed the specified page limit
4. Is clearly presented and proof-read
5. Is submitted on, or before, the specified or agreed due date. Late submissions will only be accepted in exceptional circumstances or where a deferment has been granted in advance.

I declare that I have read both the University and the School of Electronics, Electrical Engineering and Computer Science guidelines on plagiarism - http://www.qub.ac.uk/schools/eeecs/Education/StudentStudyInformation/Plagiarism/ - and that the attached submission is my own original work. No part of it has been submitted for any other assignment and I have acknowledged in my notes and bibliography all written and electronic sources used.

Student's signature *Date of submission*

Abstract

Abstract goes here

Contents

Chapter 1

Introduction

1.1 Background Information

Machine learning is the application of statistical techniques to data, allowing conclusions to be drawn about existing data, or the prediction of values in new data. It is a subset of artificial intelligence. It effectively allows the machine to make some decisions, without exhibiting some of the biases which a human would. The reliability of a model is still heavily dependent on the data which was used to train it though, which makes it important for a human to choose this carefully. A dataset is a collection of related data, which has the same format. In our case, it will be in a tabular form. The columns will be referred to as attributes, or particular characteristics of a particular row. Each of these columns will have a singular data type, but may have multiple allowed values if the data supplied is categorical in nature. A row will be referred to as an instance, and should contain either a valid value or missing value for every attribute. Datasets with missing values are an interesting problem, since we have no way of knowing what the missing value represents. There are several approaches to solving this problem, but the most common solutions involve either ignoring the instance containing the missing value, or else making an educated guess about what the missing value represents. Both mentioned approaches have positives as well as negatives. Ignoring an instance prevents you from using unreliable data, but also may lead to valuable data in the other attributes of that instance being discounted. Attempting to predict a value will probably result in you getting the value wrong, but it may be worth the risk to gain value from the other complete attributes.

1.2 Problem Description

This project aims to investigate the usage of machine learning techniques on incomplete data sets of discrete data. It will attempt to fill in missing data in certain data sets by using training data to build a model, which we will then apply to the target data set. We will keep using this model to generate predictions on produced data sets recursively, and will stop whenever the produced data set stops changing. It is expected that the prediction will trend towards the mean, and more reliable results will be achieved. Given sufficient time, it is also intended that this approach can be used to infer 'hidden variables', similarly to how neural networks produce hidden layers to create relationships between data. This will be achieved by adding a new attribute to a dataset, and initially filling it full of random valid data. We will then repeatedly build, apply and rebuild models to classify this attribute until it stops changing, in an identical manner to how were filling in new columns. New data will be added to the target data set by this method, and we will investigate how useful this hidden data is.

A package in Java for the machine learning platform Weka will be created to accomplish the goals described above. Weka is an open source machine learning platform which was initially developed at the University of Waikato, in New Zealand. It has a lot of pre-existing implementations of popular machine learning algorithms, as well as a wide range of plugins which implement more specialised functionality. It is currently my intention to package our solution as a plugin, and to distribute it. However, this depends on the utility of the results. Graphical representations of the results which are observed using this method will be produced, and compared against results which are observed using other standard methods. This functionality is built into the Weka Experimenter, and should allow the project to easily be tested against other algorithms which are already implemented within Weka. Input datasets should be in the .arff format which has been developed for Weka. This is effectively the same as a .csv file, except with some additional data at the top and using the ? symbol to represent values which are missing. Input datasets should only have nominal attributes (attributes which fit into categories), at least initially. This is for two main reasons:

1. Only having nominal attributes should simplify the implementation. This will allow us to determine whether or not the experimental technique is valid more easily, so that it could be extended to numeric types in future.

2. It should ensure convergence. With numeric data, it is probable that our data would never stop changing as we create and reapply the model.

This appear as a predicted result fluctuating around a particular value from prediction to prediction, without ever settling on a number. While it would be possible to write checks and validation to prevent this, it adds extra complexity without adding much additional value.

The goal of this project is to study the effects of iteratively building common classifiers and comparing the performance of classifiers built this way against the performance of the same classifier, but without attempting to impute missing training data. The main aims of the software to be developed will thereore be:

- A Weka plugin will be developed. This should integrate smoothly into any version of Weka ¿ 3.6, and will be made freely available.
- The classifier developed within the plugin should be usable from both the CLI and GUI.
- Running experiments using the plugin should be easy, and have well explained documentation.
- Allow any currently implemented Weka classifier which is valid to use for a given dataset to be used upon it, and for this to be easily configurable.
- Run a number of experiments to compare the performance of the iteratively trained classifier against some control cases on the same data sets.

Chapter 2

Design and Implementation

2.1 Design

This body of work is mainly interested in common classifiers, and in trying to use them in a novel way. It therefore makes sense to extend a package in which these are already implemented, and in which they are easily accessible. The Weka project therefore suits this purpose since it has well maintained, efficient implementations of most machine learning algorithms, as well as handling file I/O, and having both command line and graphical user interfaces. Since these are written in Java, it follows that the plugin should also be written in Java.

It also follows that any files which are used for testing should be in either .csv, or .arff format. The former is commonly used in data processing while the latter is a proprietary Weka format which is very similar, but differs in that it contains additional data at the top of the file pertaining to the dataset. Both are supported by Weka, and therefore either is usable.

2.2 Implementation

Chapter 3

Experiments

Chapter 4

Conclusion