

Machine Learning in the Real World

2012-02-16

David North & Richard Ashby

The company

CoreFiling is a software company
with main office in Oxford

50 people, 20 developers

Financial/business reporting
software

Mostly Java, some Scala, Python,
.NET, always willing to use the
best tool for the job

Us

The amusingly(?) named 'Team R&D', the smallest dev team in the company

Have worked together on various projects over the last 3 years

David – CS at Oxford (06-09), intern at CFL, liked it so much he came back

Richard

- Classics at Oxford 1997-2002
- CS at Sydney, 2006-07
- At CoreFiling since 2007

Why we're here

Because we want to recruit people!

Because what we're talking about is an interesting cross-over between industry and academia

The Problem

From April 2012, all UK corporation tax returns must be submitted to HMRC in XML.

- Previously, submissions have been on paper or as PDFs
- Current process and tools are geared towards producing accounts as MS Word documents

The long-term solution

Accounting software vendors
need to fix their software to
allow export of annual
returns in the XML format
required by HMRC

Meanwhile...

This will take years to
implement and trickle down to
users via upgrades

Lots of small firms do their
accounts in Excel (or even on
paper)

A stop-gap solution is called
for (and is a massive
opportunity for us)

Tagging

Take the existing Word document

Have the user (an accountant) go through and 'tag' the items HMRC expects to be reported

Use the tags to export the necessary XML

Seahorse

A web app which allows users to upload and tag their Word (or Excel) accounts, then export the XML.

We host this and sell it as SASS (Software As A Service)

Seahorse - Document Tagging - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://willapa.int.corefiling.com/Hippohorse/Document/Tag/6#4

naive bayesian classification

PD/Libraries/XBRL

Most Visited

Wiki

Decimate

Diary

SMRRM

TN

Versioning

TNT

Hippo

XIIF

Yeti

Puns

XBRL Book

mallet Changelog

TNI

Disable

Cookies

CSS

Forms

Images

Information

Miscellaneous

Outline

Resize

Tools

View Source

Options

Fires > PD/*

Status > PD/P...

yeti - Story U...

ChromeDrive...

attachment.c...

Yeti Admin

cfl - Comp So...

Bug 9094 - U...

Bug 8200 - S...

Seahorse

Seahorse - D...

SEAHORSE

The Lost Word in iXBRL Filing

Home

Manage Filers

Manage Filings

Manage Users

CoreFiling

dtnc@corefiling.com

Account

Help

Logout

Report of the directors
for the year ended 31 December 2010 (Continued)

Business Review (Continued)

Key Performance Indicators

The Group uses a range of financial performance measures to monitor the management of the business effectively; the most significant of these are the key performance indicators (KPI's).

The main KPI's are turnover, gross margin, operating profit before exceptional items, operating profit, profit before taxation, average working capital as a percentage of turnover, net funds, return on average capital employed, stock days, debtor days and creditor days.

The KPI's for the year ended 31 December 2010, with comparatives for the year ended 31 December 2009, are set out below:

Auto Tag

Bulk Review

Clear Table

| | 2010 | 2009 |
|-----------------------------------------------------|----------|---------|
| Turnover - excluding share of joint venture (£'000) | 593,927 | 541,232 |
| Gross margin | 11.2% | 11.1% |
| Operating profit before exceptional items (£'000) | 6,900 | 6,575 |
| Operating profit (£'000) | 4,839 | 6,412 |
| Profit before taxation (£'000) | 4,363 | 3,740 |
| Average working capital as a percentage of turnover | 15.3% | 13.5% |
| Net (borrowing) / funds (£'000) | (12,339) | 6,431 |
| Return on average capital employed | 5.9% | 6.3% |
| Stock days | 49 | 46 |
| Debtor days | 46 | 47 |
| Creditor days | 29 | 32 |

The KPI's have been discussed in the results for 2010 above, however where not we have the following comments:

- Gross margin has not changed at Group level. A small decrease in the Trading Division has been offset by an increase in the Fine Foods Division as a result of the weakening of the Euro.
- Average working capital as a percentage of turnover has deteriorated in the year, increasing by 1.8% to 15.3% mainly as a result of increased UK trading stocks following the set up of a fish and seafood branch, the acquisitions made in the second half of the year and prompt payment of year-end suppliers held over in the prior year as detailed below.
- Return on average capital employed which is calculated before the deduction of exceptional items decreased by 0.4% to 5.9% in the year as a result of the increased working capital being used by the Group not reflected in the year on

Suggestions

99.0% [MTS] Turnover / gross operating revenue

Browse Concepts

Concept

Label

[MTS] Turnover / gross operating revenue

Type

Duration / Monetary / Credit

Tag Details

Sign

As displayed

Dimensions

Dimension Set

Income data all dimensions

Consolidation

Company [default]

Operating activities

Total for all operating activities [def]

Amortisation and impairment adjustments

Not applicable [default]

Exceptional items adjustments

Not applicable [default]

Business segments

Not applicable [default]

Countries

Not applicable [default]

Confirm

Clear

Cancel

Done

FoxyProxy: Patterns

CoreFiling

Sub-problem

Tagging is **very** tedious

Could be several hundred table rows and text items in one filing

One person could be tasked with preparing dozens of these

Not feasible to do by hand

Machine Learning

Filings are of broadly similar shape to each other

(especially those all prepared by the same accountancy firm)

Can we use machine learning to identify where the tags need to go?

For legal/regulatory reasons,
a human must review anything
suggested automatically,
which allows us to use
supervised machine learning
methods – those where the
right answer is externally
confirmed and can be used to
improve the suggestions in
future

Scoping the problem

Can potentially use machine learning for:

- * Identifying tables in the document
- * Identifying facts in chunks of text
- * Picking the tag on a given chunk of text

Decided to only attempt the last of these – parsing unstructured data is very difficult to get right, and most of what we want to tag is in easily identifiable MS Word tables (or Excel)

Formalize the problem

Given a table, and the list of 'concepts' (possible HMRC-defined tags), suggest tags for each taggable row.

Need to define 'taggable'

This is a **classification** problem, well understood in principle (flowers into species etc).

A table to be tagged

| | 2010 | 2009 |
|-----------------------------------------------------|----------|---------|
| Turnover - excluding share of joint venture (£'000) | 593,927 | 541,232 |
| Gross margin | 11.2% | 11.1% |
| Operating profit before exceptional items (£'000) | 6,900 | 6,575 |
| Operating profit (£'000) | 4,839 | 6,412 |
| Profit before taxation (£'000) | 4,363 | 3,740 |
| Average working capital as a percentage of turnover | 15.3% | 13.5% |
| Net (borrowing) / funds (£'000) | (12,339) | 6,431 |
| Return on average capital employed | 5.9% | 6.3% |
| Stock days | 49 | 46 |
| Debtor days | 46 | 47 |
| Creditor days | 29 | 32 |

“Good programmers don't write
what they can steal”

(and good companies don't pay
for tools available as FOSS)

Are there any libraries for
solving classification
problems available as
free/open source software?

F0SS Machine Learning Tools

All developed in academia as one might expect...

Weka – as used in various CS machine learning practicals. Works very well, has one serious problem for us...

Hadoop – top-level Apache machine-learning project, lots of commercial backing and use, but v. complicated, big and heavy (more for Google and Yahoo than us)

Mallet – the happy medium.

Classification methods

Mallet has a common API
allowing us to swap in and
out different methods easily
(just change a couple of
lines of code)

Naive Bayesian Methods

As used by your spam filter
Assumes 'feature independence'
eg. Trying to decide if fruit
is an apple, assume shape,
size, taste are independent
predictors (not related),
take the product of
probabilities from each of
these

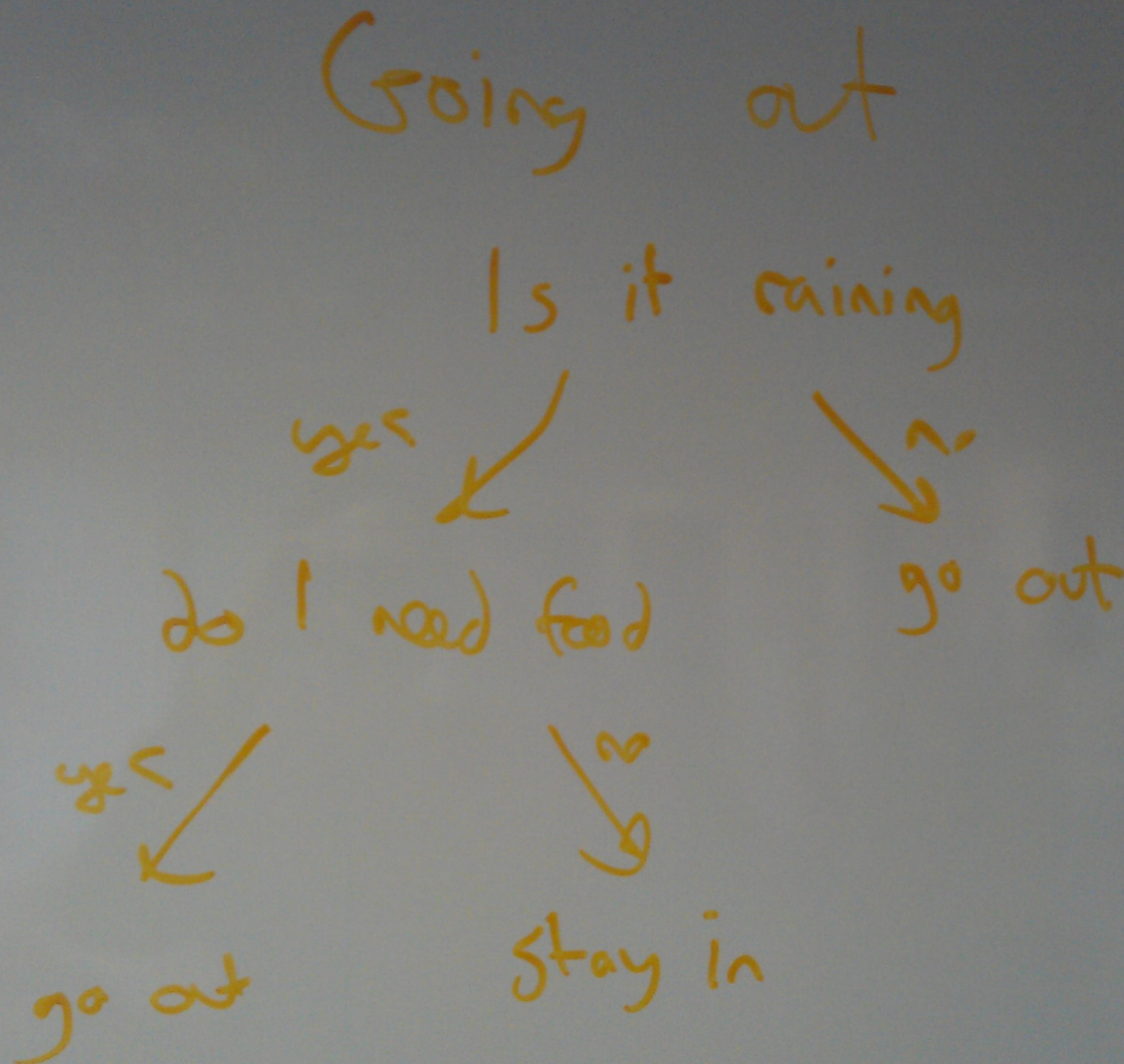
Works great on spam and other problems, especially yes/no questions

Computationally cheap

However, intuitively, for our problem, the assumption is rubbish

This is borne out by poor test results

Decision Trees



Maximum Entropy Modelling

(with apologies for hand-waving)

“Assume only as much as our available information tells us”

Classification of a given row is a discrete probability distribution over n mutually exclusive propositions (tags)

In the absence of any
information, assume all known
tags are equally likely

As we gather training data, we
want our prob. distribution
to 'agree' with it – various
means of extracting
constraints from the data

Having extracted our constraints,
brute-force the set of possible

prediction functions satisfying them
until we find the one which gives the
maximum entropy on the available data.

'Brute force' can actually be refined
to a spatial search in n dimensions.
Quite expensive (time/memory) to do,
but model cheap to use once built.

How do we know it's working?

No maths/formal methods

Test-driven development

Quantify the desired behaviour
(using our tame accountants)

Write automated tests
capturing this behaviour

Initially, they'll fail (we
haven't written any code yet)

Refine the implementation
until the tests pass and the
customer is satisfied wrt the
original spec

Guard against regression by
running the tests
automatically (not just
nightly, all the time –
continuous integration)

Yay, lots of tests all passing

| | | |
|----------------------------------------------------------------------------|---------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|
| PD/Products/Seahorse | | |
| Hippocampus Shared | Mallet | Hippocampus Classifier Builder |
| Build (JDK1.6): 2011-10-14 12:11 GMT (r13845) OK | Build (JDK1.6): 2011-09-29 12:27 GMT (422) OK | Build (JDK1.6): 2011-10-14 12:28 GMT (r13846) OK |
| | | Long Tests: 2011-10-18 01:06 GMT (r13869) OK |
| | | Classifier Build Timing Tests: 2011-10-06 19:11 GMT (r13682) OK |
| Hippocampus Metrics | Hippocampus | Seahorse+Hippocampus Integration |
| Learning Accuracy: 2011-10-07 10:17 GMT (r13685) OK | Release: 2011-10-14 13:16 GMT (r13847) OK | Hippocampus for Seahorse: 2011-10-17 15:27 GMT (r13760) OK |
| Search Accuracy: 2011-10-07 10:17 GMT (r13685) OK | DistSpec: 2011-10-14 13:35 GMT OK | Hippocampus Conformance (Head): 2011-10-18 14:57 GMT (r13897) OK |
| Fallback Accuracy: 2011-10-07 10:17 GMT (r13685) OK | Web Tests: 2011-10-14 13:35 GMT (r13847) OK | Hippocampus Conformance (Supported): 2011-10-18 14:57 GMT (r13897) OK |
| Accuracy & Timing Comparison: 2010-09-16 09:51 GMT OK | Taxonomy Config Tests: 2011-10-10 09:34 GMT (r13765) OK | |
| | Long Tests: 2011-10-10 18:02 GMT (r13783) OK | |
| | Concurrency Tests: 2011-10-14 13:35 GMT (r13847) OK | |
| | Long Running Threading Tests: 2011-10-16 01:04 GMT (r13859) OK | |
| | Stress Tests: 2011-10-07 01:05 GMT (r13682) OK | |
| | Published Releases: 2011-10-07 17:12 GMT (r13760) OK | |

Not just pass/fail

In the case of this problem,
measure accuracy as a
percentage and plot a graph.

Cross-fold testing: take known
tagged documents, leave one
out, train with others and
test with this one

Take the mean result

Choosing our metrics

The obvious: how often is the exact right answer selected?

How often is the right answer in the top ten?

How often are we under-confident in the right answer or over-confident in the wrong one?

As it turns out...

Right answer is in top 10 suggestions 78% of the time (mean x-fold across available data)

Right answer is top suggestion 63% of the time

This took two people just over a year (~ 2,000 man hours) to achieve