Feature-Based and Feature-Free Textual CBR: A Comparison in Spam Filtering

Sarah Jane Delany[1] and Derek Bridge[2]

[1] Dublin Institute of Technology,
sarahjane.delany@comp.dit.ie
[2] University College Cork
d.bridge@cs.ucc.ie

Abstract. Spam filtering is a text classification task to which Case-Based Reasoning (CBR) has been successfuly applied. We describe the ECUE system, which classifies emails using a feature-based form of textual CBR. Then, we describe an alternative way to compute the distances between cases in a feature-free fashion, using a distance measure based on text compression. This distance measure has the advantages of having no set-up costs and being resilient to concept drift. We report an empirical comparison, which shows the feature-free approach to be more accurate than the feature-based system. These results are fairly robust over different compression algorithms in that we find that the accuracy when using a Lempel-Ziv compressor (GZip) is approximately the same as when using a statistical compressor (PPM). We note, however, that the feature-free systems take much longer to classify emails than the feature-based system.

1 Introduction

Spam email has proved to be a problem that is enduring and difficult to solve. In January 2004, Bill Gates predicted that spam email would be eradicated as a problem within two years[3]. The fact that this prediction did not come to pass demonstrates the severity of the problem. Identifying spam is a difficult task for a number of reasons. Spam is a diverse concept: spam advertising cheap prescription drugs has little in common with spam offering investment opportunities. In addition, spam is constantly changing, new opportunities are persistently being exploited by spammers and seasonal effects have an impact. A key factor though in this concept drift is that spammers continually change the content and structure of spam email in order to bypass the mechanisms in place to stop them. There is also a subjective and personal aspect to identifying spam: what is considered to be spam by one individual may not be considered spam by others. Finally, mistakingly identifying a legitimate email as spam (known as a False Positive) is very significant in this domain and is unacceptable to most email users.

[3] http://www.theregister.com/2004/01/26/well_kill_spam_in_two/

Of the wide range of strategies that have been used to combat spam some of the more effective have been: whitelists and blacklists[4], authentication-based techniques[5], and spam filtering including both collaborative filters [1] and content-based filters. In this paper we focus on ECUE, a personalised, content-based filter that uses Case-Based Reasoning (CBR) to classify emails. ECUE has been shown to be successful at filtering spam [2] and at handling concept drift in spam [3].

The case representation used in ECUE is feature-based. In this paper we describe an alternate way to calculate the distances between cases which is feature-free, using a distance measure based on text compression. The distance measure performs considerably better than the feature-based measure and has the advantage of having no set-up cost and being resilient to concept drift.

The remainder of the paper is organised as follows. Section 2 outlines ECUE, the spam filtering system used for the evaluation in this paper. Section 3 discusses the feature-based approach to spam filtering, identifying how features are extracted, selected and represented. Section 4 then discusses the feature-free alternative approach, describing the compression-based distance measure that can be used in textual CBR. Evaluations of both the feature-based and feature-free approaches are described in Section 5 with a discussion of the results in Section 6. The paper concludes in Section 7 with an outline of possible future work.

2 Email Classification Using Examples

ECUE [2, 3] is a personalised case-based machine learning system that uses past examples of a user's email as training instances. A case base of examples of an individual's previously received emails, both spam and legitimate, is set up. New emails are classified against the case base using the k-Nearest Neighbour algorithm. The k cases that are the nearest neighbours i.e. the closest in distance, to the target case are returned and used to generate a classification for the target case. Due to the significance of False Positives (FPs), the classification process uses unanimous voting to bias the classifier away from FP classifications. This requires all k neighbours retrieved by the Nearest Neighbour algorithm to be of class *spam* before the target case can be classified as spam. A Case Retrieval Net [4] is used to speed up the retrieval process.

One of the challenges of using CBR for spam filtering is to manage the training data, choosing those training examples that are best at prediction. Prior to classification, case base editing is performed on the case base to reduce the number of cases. The case base is edited with the Competence-Based Editing (CBE) technique [5] which uses the competence properties of the examples in the case base to identify and remove noisy and redundant cases. CBE has been shown to conservatively reduce the size of a spam case base while maintaining and even improving its generalisation accuracy [5].

[4] www.email-policy.com/Spam-black-lists.htm
[5] www.emailauthentication.org/

3 Feature-Based Textual CBR

Each training instance in ECUE is a case e_j represented as a vector of feature values, $e_j = (f_{1j}, f_{2j}, \ldots f_{nj}, c)$ with c representing the class of the email, either *spam* or *nonspam*. The case representation is binary; if the feature exists in the email the feature value $f_{ij} = 1$, otherwise $f_{ij} = 0$. The features are identified by lexical analysis of the textual content of the email.

No stop-word removal or stemming is performed on the text. Email attachments are removed but any HTML text present in the email is included. As ECUE is a personalised filter, the header fields may contain useful information and a selection of header fields, including the *Subject*, *To* and *From* headers, are included in the tokenisation.

Three types of features are extracted: word features, character features and structural features (e.g. the proportion of uppercase characters, lowercase characters or white space in the email). The *feature extraction* process results in a large number of features. In addition, the representation of each email is sparse, with only a small number of the total feature set having a value other than zero. *Feature selection* using Information Gain (IG) [6] is performed to identify the features which are most predictive of spam or legitimate mails. As the case representation is binary, the IG value for character and structural features is also used as a threshold to indicate whether the feature should be set in the case representation or not [2]. Based on the results of preliminary cross-validation experiments, we chose to use 700 features for the evaluations in this paper.

4 Feature-Free Textual CBR

There is a feature-free alternative to feature-based textual CBR. As we will explain, we can define a distance measure based on text compression [7]. Distance measures based on data compression have a long history in bioinformatics, where they have been used, e.g., for DNA sequence classification [8]. Outside of bioinformatics, compression-based distance measures have been applied to *clustering* of time-series data [9] and languages [10, 11]. They have also been applied to *classification* of time series data [9]. But, to the best of our knowledge, they have not been applied to text categorisation in general or spam filtering in particular.[6]

Keogh *et al.* [9] and Li *et al.* [7] have both presented generic distance measures based on data compression and inspired by the theory of Kolmogorov complexity. The *Kolmogorov complexity* $K(x)$ of a string x can be defined as the size of the smallest Turing machine capable (without any input) of outputting x to its tape. The *conditional Kolmogorov complexity* $K(x|y)$ of x relative to y can be defined as the size of the smallest Turing machine capable of outputting x when given y as an input. This can be the basis of a distance measure. Informally, if $K(x|y) < K(x|z)$, then y contains more information content that is useful to outputting x than z does, and so y is more similar to x than z is.

[6] But see the discussion of the possibility of using compression in instance-based classification of email at www.kuro5hin.org/story/2003/1/25/224415/367

One possible way to define a normalised distance measure using Kolmogorov complexity is:

$$d_K(x,y) =_{\text{def}} \frac{K(x|y) + K(y|x)}{K(xy)} \tag{1}$$

where $K(xy)$ is the size of the smallest Turing machine for outputting y concatenated to x.

Unfortunately, Kolmogorov complexity is not computable in general, and so we must approximate it. One way of thinking of $K(x)$ is that it is the best compression we can achieve for x. So, we can approximate $K(x)$ by $C(x)$, the size of x after compression by a data compressor. Then distance can be defined as

$$d_C(x,y) =_{\text{def}} \frac{C(x|y) + C(y|x)}{C(xy)} \tag{2}$$

where $C(x|y)$ is the size of x after compression by a compressor that has first been 'trained' on y, $C(y|x)$ is defined analogously, and $C(xy)$ is the compressed size of y concatenated to x.

But d_C also has problems because standard compressors do not allow easy computation of $C(x|y)$. Hence, following Keogh *et al.* [9], we make a further simplification. Given strings x and y, a Compression-based Dissimilarity Measure (CDM) can be defined as follows:

$$\text{CDM}(x,y) =_{\text{def}} \frac{C(xy)}{C(x) + C(y)} \tag{3}$$

Even with the best possible compression algorithm, the lowest value this can produce is slightly above 0.5: even if $x = y$, $C(xy)$ will be slightly greater than $C(x)$. In principle CDM's maximum value is 1; this would occur when x and y are so different that $C(xy) = C(x) + C(y)$ and so compressing y within xy is not helped by having compressed x first.

It should also be noted that properties expected of distance measures do not hold. In general, $\text{CDM}(x,x) \neq 0$; $\text{CDM}(x,y) \neq \text{CDM}(y,x)$, i.e. CDM is not symmetric; and $\text{CDM}(x,y) + \text{CDM}(y,z) \not\geq \text{CDM}(x,z)$, i.e. the triangle-inequality does not hold. None of this prevents use of CDM in, for example, classification tasks, provided the classification algorithm does not rely on any of these properties. For example, an exhaustive implementation of k-NN (in which the algorithm finds the k nearest neighbours to the query by computing the distance between the query and *every* case in the case base) will work correctly. But retrieval algorithms that rely on these properties to save distance computations (e.g. k-d trees [12] and Fish and Shrink [13]) are not guaranteed to work correctly.

CDM is a feature-free approach to computing distance. Cases are represented by raw text: there is no need to extract, select or weight features; there is no need to tokenise or parse queries or cases. CDM works directly on the raw text. We discuss the advantages of this in Section 6.

5 Spam Filtering Experiments

We conducted an experimental evaluation whose objective was to replace the feature-based similiarity measure that ECUE uses with a compression-based similarity measure and to compare the two measures.

The datasets used in this evaluation were derived from two corpora of email. Each email corpus is a personal collection of the spam and legitimate email received by an individual over a period of approximately two years. The legitimate emails in each corpus include a variety of personal, business and mailing list emails. Two datasets (Datasets 1.1 and 1.2) were extracted from one corpus, while Datasets 2.1 and 2.2 were extracted from the other.

Each dataset consists of 1000 emails, 500 of each class, received over a period of approximately three months. Most individuals do not receive equal volumes of spam and legitimate email, but the actual distributions vary considerably from person to person. Weiss and Provost [14] conclude that a balanced distribution is a reasonable default when the true distribution is not available, and this is what we have chosen here.

Since FP classifications are significant in this domain, straightforward classification accuracy (or error) as a measure of performance does not give the full picture. The evaluation metrics we use include:

(i) The error rate, i.e. the overall proportion of emails that were not filtered correctly (*%Err*).
(ii) The FN rate, i.e. the proportion of spam emails that were missed (*%FNs*).
(iii) The FP rate, i.e. the proportion of legitimate emails that were classified as spam (*%FPs*).

We compare the performance of the different classifiers by calculating confidence levels using McNemar's test [15].

Fig. 1 compares feature-based similarity (FBS) with compression-based similarity on each of the four datasets. The graphs show the results of using feature-based similiarity on both a full case base and a case base edited using CBE. They also include the results of using CDM with GZip as the compressor on the full case base.

The beneficial effect of case base editing can be seen from the results. But, using CDM, the compression-based approach, improves the results even more, with the differences in the overall error (*%Err*) between CDM and the feature-based similarity on an edited case base significant in all cases at the 99.9% level, except for Dataset 1.1. Although the FP rate is also lower for CDM than for feature-based similarity on an edited case base, these differences are not significant. However, biasing the classifier away from FPs already results in a relatively low FP rate, so it would be difficult to achieve significance using McNemar's test.

Fig. 2 compares the use of two different compression algorithms in the CDM measure, GZip and PPM. GZip is a variant of Lempel-Ziv compression, in which a repetition of a string within a text may be replaced by a pointer to an earlier occurrence. In GZip, substitutions are confined to a 32 Kbytes sliding window. PPM, Prediction by Partial Matching [16], is an adaptive statistical compressor.

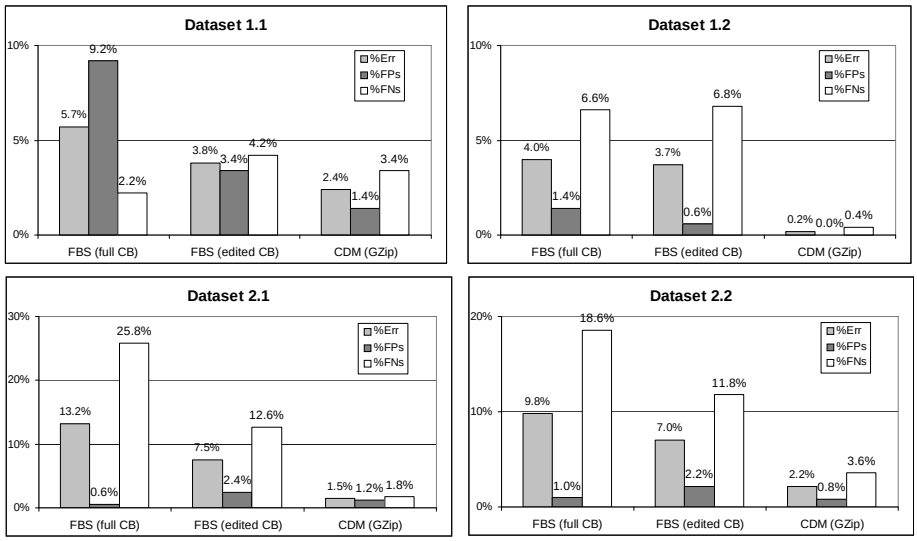

Fig. 1. Comparison between feature-based similiarity (FBS) and compression-based similarity, using 10-fold cross validation on each dataset.

A statistical compressor builds a probabilistic model from which it can predict the most likely next character in the stream, and encodes the more probable characters in fewer bits. If the model is of order n, then the next character is predicted based on the previous n characters. An adaptive compressor updates its model on the basis of the character frequencies seen so far, hence the bit pattern used to encode a character may change. PPM adaptively builds models of all orders up to n; it uses the model with largest order, but if a novel character is encountered, an escape symbol is included in the output and PPM switches to the model with next lowest order. A default model at level -1 ensures that every character can be encoded. In our experiments, we tried orders of 2, 4 and 8.[7] Orders above 6 or so generally do not increase the amount of compression [16].

In general, PPM is thought to achieve some of the best compression rates. However, on the emails in our corpora we found GZip to be slightly better: its average compression was 59% compared with 53.3% for PPM(2), 56.6% for PPM(4) and 56.9% for PPM(8). There is not much difference between the compression rates achieved for spam and non-spam emails, with approximately 0.5% difference either way. The better the compression rate, the closer $C(x)$ will approximate $K(x)$, the Kolmogorov complexity, which can be thought of as the best compression one can achieve on x. But this does not mean using the better compressor in CDM will result in a better approximation of d_K, the distance

[7] In these experiments, we use Bob Carpenter's implementation of PPM: http://www.colloquial.com/ArithmeticCoding/

measure based on Kolmogorov complexity. This is because the improvement in compression rate of the better of two compressors on the different terms in Equation (3) may not be the same [11].

In fact, the results in Fig. 2 show that there is little difference in classification error between the different compression algorithms. None of the differences is statistically significant using McNemar's measure. This suggests that the choice of compression algorithm does not matter greatly and supports the findings in [11], where results on *clustering* tasks were fairly robust over different compressors.

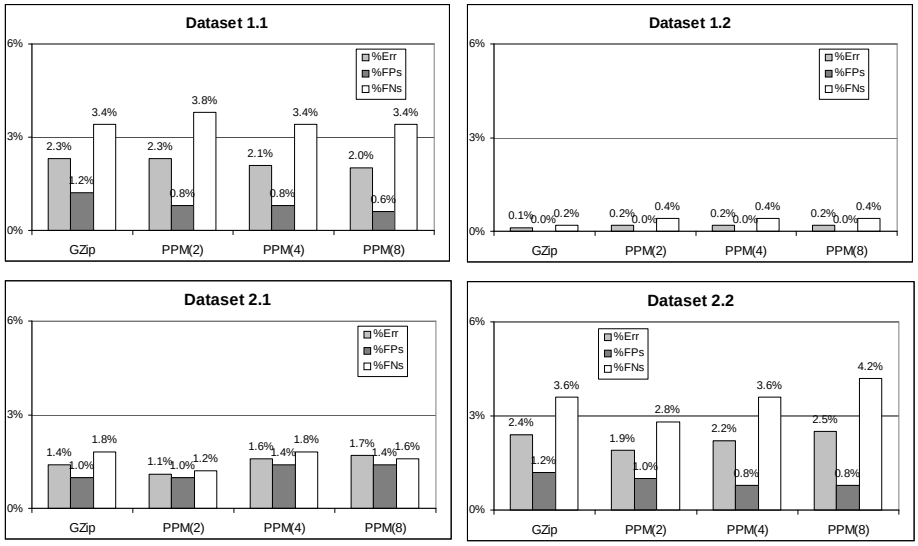

Fig. 2. Comparison between GZip compression and PPM compression for the CDM measure, using leave-one-out cross validation on each dataset.

A limitation of the compression-based approach is the time it takes to classify an email. Table 1 shows the time taken in seconds to classify a single email using feature-based and feature-free approaches with a case base of 1000 cases. The time to classify a single email using CDM is, at best, 180 times slower than using the feature-based similarity. The compression algorithm is computationally much more expensive than comparing feature values. Furthermore, compression-based similarity requires the target case to be compared with each case in the case base, which is not always necessary in feature-based similiarity if, for example, a Case Retrieval Net is used.

Using CDM with GZip performs significantly better in terms of computation time than CDM with PPM. CDM with GZip can be made somewhat faster by modifying the length of the email files to take into account the fact that the GZip algorithm uses a sliding window size of 32 Kbytes. Truncating the email files to 16 Kbytes each before calculating the CDM achieves speed-ups of between 9.5%

and 25% on the datasets evaluated. The CDM-GZip figures in Table 1 include this speed up. We found that the truncation of the email files does not have any real effect on the classification error results. The results in Fig. 1 also include this speed-up.

Table 1. Time to classify one email in seconds using different similarity measures.

Dataset	**Feature-based**	**CDM-Gzip**	**CDM-PPMZ(2)**	**CDM-PPMZ(4)**	**CDM-PPMZ(8)**
Dataset 1.1	0.01	2.00	30.8	37.2	69.2
Dataset 1.2	0.01	1.84	24.9	28.3	32.7
Dataset 2.1	0.01	1.82	24.2	28.1	32.4
Dataset 2.2	0.01	1.97	25.5	29.8	34.7

6 Discussion

A feature-free approach to spam filtering, such as that offered by CDM, has several advantages over a feature-based approach. The first, as we have seen in the previous section, is its remarkable accuracy. A second advantage is its low set-up costs: feature extraction, selection and weighting are all unnecessary. This is a major advantage when one considers that spam is a personal and diverse concept.

A third advantage, related to the second, concerns concept drift. Delany *et al.* [2] describe a three-level hierarchy of actions for coping with concept drift that would be needed in a production-quality feature-based spam filter. Level 1 is regular case base update, i.e. updating the case base with misclassified emails. Level 2, with lower frequency, is feature selection, i.e. periodically reselecting features from the most recently selected set of candidate features. Level 3, with lowest frequency, is feature extraction, i.e. periodically re-extracting a set of candidate features from the most recent training examples. A feature-free approach requires only Level 1 actions. This is a major advantage when one considers how constantly spam changes.

The feature-free approach also has its disadvantages. These are at least two-fold. First, CDM returns only a number, denoting distance. It does not return any factors that could be used to *explain* its judgements or to drive case adaptation. Adpatation is not relevant to spam filtering and it traditionally has lower importance in textual CBR, but lack of explainability could inhibit broader uptake.

A second disadvantage is computation time. Our experiments show that CDM with GZip takes up to 2 secs to classify an email compared to 0.01 secs for the feature-based system. Rarely will a user ever notice this cost: it is unlikely to matter if an email is available for viewing an extra 2 seconds after delivery. In any case, the CDM experiments are based on an unedited case base of 1000 cases. Classification time will be lower if the case base is smaller. In live experiments with ECUE, case base sizes were significantly smaller than 1000, at approximately 300 cases [17].

There may be other ways of reducing the number of distance computations. For example, the compression-based distance measure described in [7] comes close enough to satisfying the conventional properties expected of distance measures (triangle inequality, etc.) to allow use of retrieval algorithms (such as k-d trees and Fish and Shrink) that rely on these properties. Precomputation and caching of $C(x)$ for each case x as it enters the case base will also help.

Finally, we should consider how robust a spam filter that uses CDM might be. Spammers are constantly trying to outwit the latest spam filters. How easy will they find it to outwit a compression-based approach? One possibility, which spammers use even now, is to place all content into images, rendering it inaccessible to filters that look only at the textual content. Another possibility, which is also in current use, is to add large quantities of spam salad, i.e. random text, to the end of the message. To outwit spam filters, the spam salad in different spam emails would need to be adequately dissimilar.

7 Conclusion

In this paper we have shown that a compression-based distance measure achieves significantly higher classification accuracy than the normal feature-based distance measure typically used in textual CBR. The compression-based measure has many advantages including no set-up costs (feature extraction, selection and weighting) as the raw text files are used directly. In addition, the measure is resistent to concept drift as it works directly on the raw text.

However, using the compression-based distance measure has higher computation cost which varies with the compression algorithm used. Using GZip for compression allows some speed-ups and, depending on the domain, the computation cost may not be prohibitive. The lack of features can also effect the ability of a textual CBR system using this measure to provide explanations for its results.

Future work in this area will include work on the computation time issues, investigating algorithms to speed up retrieval time. We will also perform an empirical investigation of the CDM measure's resilience to the concept drift in spam including investigating how compression-based similarity affects competence-based case base editing on spam training sets. The work will extend to the application of this measure to texts other than emails, to tasks other than classification, and to text other than raw text, e.g. text that has undergone POS-tagging.

Acknowledgments

The authors are grateful to Pádraig Cunningham for discussions about this work.

References

1. Gray, A., Haahr, M.: Personalised, collaborative spam filtering. In: Proceedings of 1st Conference on Email and Anti-Spam. (2004)
2. Delany, S., Cunningham, P., Coyle, L.: An assessment of case-based reasoning for spam filtering. Artificial Intelligence Review **24** (2005) 359–378
3. Delany, S.J., Cunningham, P., Tsymbal, A., Coyle, L.: A case-based technique for tracking concept drift in spam filtering. Knowledge-Based Systems **18** (2005) 187–195
4. Lenz, M., Auriol, E., Manago, M.: Diagnosis and decision support. In Lenz, M., Bartsch-Sporl, B., Burkhard, H., Wess, S., eds.: Case-Based Reasoning Technology, From Foundations to Applications. Number 1400 in LNAI, Springer-Verlag (1998) 51–90
5. Delany, S.J., Cunningham, P.: An analysis of case-based editing in a spam filtering system. In Funk, P., González-Calero, P., eds.: 7th European Conference on Case-Based Reasoning (ECCBR 2004). Volume 3155 of LNAI., Springer (2004) 128–141
6. Quinlan, J.R.: C4.5 Programs for Machine Learning. Morgan Kaufmann Publishers Inc., San Mateo, CA. (1997)
7. Li, M., Chen, X., Li, X., Ma, B., Vitanyi, P.: The similarity metric. In: Proc. of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms,. (2003) 863–872
8. Loewenstern, D., Hirsh, H., Yianilos, P., Noordewier, M.: DNA sequence classification using compression-based induction. Technical Report 95-04, DIMACS (1995)
9. Keogh, E., Lonardi, S., Ratanamahatana, C.: Towards parameter-free data mining. In: KDD '04, Procs of the 10th ACM SIGKDD, Int Conf on Knowledge Discovery and Data Mining, New York, NY, USA, ACM Press (2004) 206–215
10. Benedetto, D., Caglioti, E., Loreto, V.: Language trees and zipping. Physical Review Letters **88** (2002) 048702
11. Cilibrasi, R., Vitanyi, P.: Clustering by compression. IEEE Transactions on Information Theory **51** (2005) 1523–1545
12. Wess, S., Althoff, K.D., Derwand, G.: Using k-d trees to improve the retrieval step in case-based reasoning. In Wess, S., Althoff, K.D., Richter, M., eds.: Topics in Case-Based Reasoning. Number 837 in LNAI, Springer (1994) 167–181
13. Schaaf, J.W.: Fish and shrink. A next step towards efficient case retrieval in large-scale case bases. In Smith, I., Faltings, B., eds.: Advances in Case-Based Reasoning. Number 1186 in LNAI (1996) 362–376
14. Weiss, G., Provost, F.: Learning when training data are costly: The effect of class distribution on tree induction. Journal of Artificial Intelligence Research **19** (2003) 315–354
15. Salzberg, S.: On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery **1** (1997) 317–327
16. Cleary, J.G., Witten, I.H.: Data compression using adaptive coding and partial string matching. IEEE Transactions on Communications **32** (1984) 396–402
17. Delany, S.J., Cunningham, P., Symth, B.: ECUE: A spam filter that uses machine learning to track concept drift. In: Proc of the 17th Eur. Conf. on Artificial Intelligence, (PAIS stream) to appear. (2006)