Data analysis of the *Web Quality Assessment* data set

Chris Sipola
s1667278
University of Edinburgh
s1667278@sms.ed.ac.uk

Elias Mistler
s1675946
University of Edinburgh
s1675946@sms.ed.ac.uk

Contents

Project Report, Data Mining and Exploration, Spring 2017 April 4, 2017

1 Introduction

This report is to document the authors' exploratory data mining project at the University of Edinburgh and will be handed in as coursework submission for the course *Data Mining and Exploration*, Spring 2017. We will explore a pre-processed data set of annotated websites, perform dimensionality reduction on the data and assess the performance of several statistical classification models on the task of annotating unseen web sites.

1.1 The Web Quality Assessment data set

This paper will use the ECML/PKDD 2010 Discovery Challenge data set[1] to conduct an exploratory data analysis and train several classifiers to determine whether the sites fall into each of six selected classes. The data set contains information on 23 million webpages in 99 thousand hosts in the `.eu` domain, downloaded in early 2010 by the European Archive Foundation [1]. However, the data used for this paper will focus on the 1,351 labeled English websites, along with their respective term frequency-based (or "text-based") features.

The websites were placed into categories according to a manual process following a prescribed set of guidelines [1]. The six class labels used in this analysis—*Commercial*, *Educational/Research*, *Discussion*, *Personal/Leisure*, and *Media*—are not mutually exclusive. For example, a website can have a positive label for both *Educational/Research* and *Media*. Classification will therefore be separate for each label since predicting a single label out of the six does not make sense.

1.2 Relevant background and previous work

Previous research makes use of most of the available features in the ECML/PKDD 2010 Discovery Challenge data set, including data on content statistics, link-related features (both at the page and the host level), and text features. The most common treatment of the text features was to calculate the *term frequency–inverse document frequency (tf-idf)* metric for each website. The *tf-idf* metric is a popular "embedding" (or vector representation) used to represent "documents" (or in our case, websites), and is calculated using the formula

$$w_{t,d} = \big(1 + \log(tf_{t,d})\big)\log\Big(\frac{N}{df_t}\Big) \tag{1}$$

where tf_d is the term frequency for document d, N is the number of documents, and df_t is the number of documents in which term t occurs. Other researchers then performed feature selection using information gain as the selection criterion to determine which *tf-idf* dimensions to keep; this method "attempts to remove non-informative terms" to improve generalization performance [2, 3].

For classification, randomized decision trees were found to work better than competing models—such as the class-feature centroid classifier and support vector machines—given that they output class probabilities that are close to the respective class probabilities in the training data. Text-based features (i.e., the *tf-idf* metric) provided the most predictive power, while link features provided the least. However, using *all* feature types gave improved performance over just using the text-based features, suggesting that the different feature types provide complementary information [2, 3].

1.3 Relevance of website classification

The task of automatically determining the quality and classification of websites is important for applications of web processing and archiving. For example, popular search engines must automatically detect the quality and category of results before serving them to users. Additionally, some online ad publishers automatically assign "quality scores" to ads, penalizing advertisers for having low-quality landing pages and therefore diminishing trust between the publisher and the user. These methods can also be used in detecting phishing and spam, making web browsing a safer experience.

[1] This is the *Web Quality Assessment* data set from the University of Edinburgh's Data Mining and Exploration (DME) course notes: `http://www.inf.ed.ac.uk/teaching/courses/irds/miniproject-datasets.html#prj8`.

2 Data preparation

For our features, we used the term and document frequencies data, which is contained in an 822 MB text file. The data is sparse, meaning that there is only data for which term frequencies were nonzero. Each line is in the format `hostid wordid1 tf1 df1 wordid2 tf2 df2`, where `hostid` is the identifier for the URL, `wordid1` is the identifier for word 1, `tf1` is the term frequency for word 1 and `df1` is the document frequency for word one.[2] There were 50,000 potential words for which each URL could have term and document frequency data. As mentioned earlier, only 1,351 (of just over 60,000) URLs had labels, so in data processing we only kept lines for which we had a corresponding URL in the label data.

Given this data, we calculated *tf-idf* according to equation 1, which was preferred over simply using term frequency because pilot runs and prior research showed that it resulted in better predictions. Since many machine learning models do not take sparse data, the sparse vectors were converted to full.[3] Terms with a frequency of zero were assigned the minimum *tf-idf* value in the data set. Each column was then scaled to have zero mean and unit variance.

Since using feature vectors of size 50,000 was infeasible, the dimensionality was reduced using *principal component analysis (PCA)*. Due to memory constraints, we used the Python `scikit-learn` implementation of *incremental principal component analysis*, [4] a batch-based approximation of PCA designed to work with constant memory requirement independent of the data volume. Note that the maximum number of principal components (PCs) was limited to the number of labeled web sites.

We then merged these labels with the input features. Forty-four rows of the merged data set had no labels assigned because they were deemed spam. We removed these, resulting in a final data set of 1,307 observations with columns containing data on both the PC scores and the label data.

3 Exploratory data analysis

Many of the class labels were imbalanced. For example, *Discussion* has 5.7% positive values, *News & Editorial* 3.8%, and *Media* just 1.5%. This imbalance will make modeling a bit more difficult for these classes. Since the labels are not mutually exclusive, summing the percentages will give 113%. This means that each website has, on average, 1.13 labels.

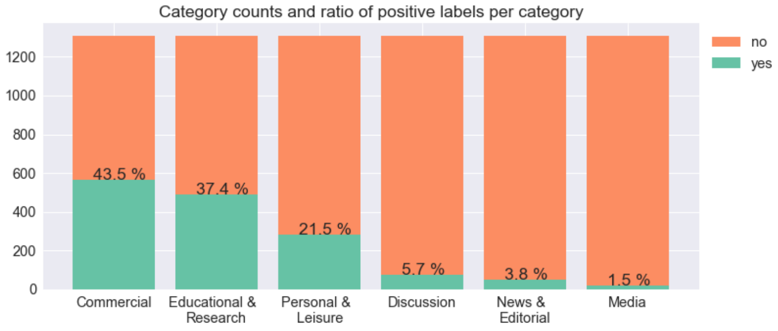

Figure 1: Positive labels per category

Figure 2 shows the variance explained by the PCs. By the 1,000th PC, 99.93% of the variance had been explained, i.e. the true dimensionality of the data is much lower than the number of *td-idf* features. The figure highlights in orange the PC values $k \in \{1, 2, 3, 4, 5, 7, 10, 25, 100, 250, 1000, 1307\}$ that were used in the analysis, since different values can lead to better or worse generalization performance in different models.

[2] `https://dms.sztaki.hu/node/350`

[3] That is, each sparse feature was expanded to have size 50,000, where entries without data were given the value of the minimum value in the matrix.

[4] `http://scikit-learn.org/stable/auto_examples/decomposition/plot_incremental_pca.html`

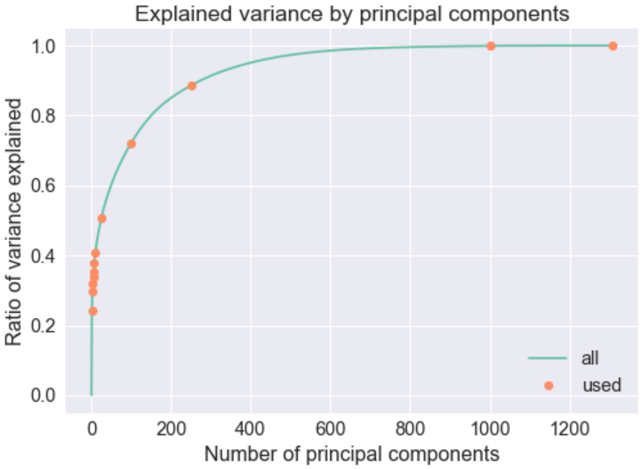

Figure 2: Variance explained by the number of PCs selected.

Figure 3 visualizes the first two PC scores for each of the six labels. Since labels are not mutually exclusive, we made the points transparent and plotted the relatively infrequent multi-label points on top of one another. This, along with the natural data overlap in scatter plots, may hide relationships within individual labels. Regardless, we can see that *Media* tends to have low PC 1 and PC 2 scores while *Commercial* and others have larger scores. Furthermore, *Commercial* tends to have higher PC 2 scores than PC 1 scores, while *Educational/Research* is the reverse. These observations indicate that the first two PCs have some discriminative power even though they only account for less than 25% of the variance.

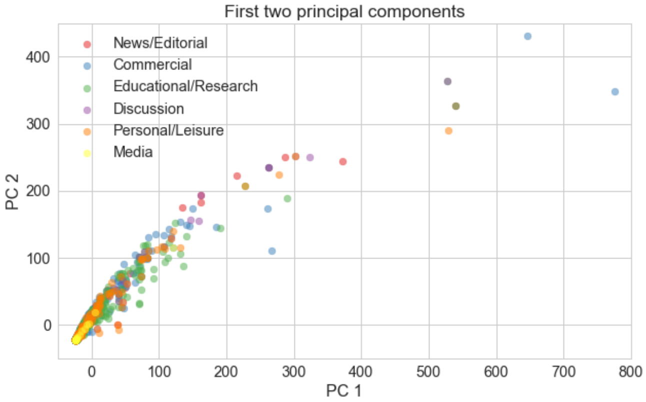

Figure 3: 2D representation of the data (excluding 1 outlier)

4 Learning methods

Because the six classes are not mutually exclusive, a binary prediction of the label class (a simple "yes" or "no") was performed separately for each website and class using the following classification models:

- **Dummy classifier**, which simply "generates predictions by respecting the training set's class distribution" (represented as `dummy` in the analysis).
- **Logistic regression**, with L2 regularization.

- **Naive Bayes**.
- **Multi-layer perceptron (MLP)**, or neural network, with two hidden layers. The number of hidden units in each layer was dependent on the number of PCs k used in the analysis, with the first layer having k hidden units and the second layer having $\lceil \sqrt{k} \rceil$. For the initialization and hyperparameter settings, we simply used the default arguments for the `scikit-learn` function `sklearn.neural_network.MLPClassifier`, which include a ReLU activation function, an Adam optimizer and a learning rate of 10^{-4}.
- **Random forest**, with 10 trees per forest.

The data analysis process follows algorithm 1, which builds a nested dictionary of performance statistics that are used for the rest of the analysis. In the procedure, we first loop over labels, splitting the data into training and validation sets. In the split, we stratify by the labels since some classes can be very imbalanced, as mentioned earlier.[5] We then perform loops over models, PC dimensions, and finally performance metrics, calculating the performance of each combination of variables in the loop. Given this data, we are then able to take the best value of a selected performance metric across values of k for each model and label.

Algorithm 1 Procedure for testing and validating models.

```
procedure TrainAndValidate($X, y, labels, models, metrics$)
    $stats_{all} \leftarrow$ dict()                                ▷ initialize statistics as dictionary
    for $label \in labels$ do                    ▷ iterate over labels, e.g., Commercial and Media
        $stats_l \leftarrow$ dict()                                       ▷ initialize stats for $label$
        $X^{(t)}, X^{(v)}, y^{(t)}, y^{(v)} \leftarrow$ split$(X, y)$      ▷ split into training and validation (stratified by $y$)
        for $model \in models$ do                              ▷ e.g., random forest and Naive Bayes
            $stats_m \leftarrow$ dict()
            for $k \in K$ do                  ▷ $K = \{1, 2, 3, 4, 5, 7, 10, 25, 100, 250, 1000, 1307\}$
                $stats_k \leftarrow$ dict()
                $X_l^{(t)} \leftarrow$ subset$(X^{(t)}, k)$                ▷ subset training data for first $k$ PCs
                $X_l^{(v)} \leftarrow$ subset$(X^{(v)}, k)$                    ▷ same, but for validation data
                $fit \leftarrow model(X_l^{(t)}, y^{(t)})$                        ▷ fit model with training data
                for $metric \in metrics$ do                               ▷ e.g., accuracy and F1-score
                    for $s \in \{v, t\}$ do               ▷ $s$ denotes split type: training or validation
                        $stats_k[(metric, s)] \leftarrow$ calcmetric$(metric, X_l^{(s)}, y^{(s)})$
                    end for
                end for
                $stats_m[k] \leftarrow stats_k$            ▷ store performance metric stats for this value of $k$
            end for
            $stats_l[model] \leftarrow stats_m$
        end for
        $stats_{all}[label] \leftarrow stats_l$
    end for
    return $stats_{all}$
end procedure
```

5 Results

The best combination of model, target label and number of PCs was logistic regression, *Commercial* and $k = 250$, respectively, achieving an F1 score of 0.66 on the validation set and beating the dummy classifier score of 0.48. While this performance is not outstanding, this is somewhat expected given that the data set has high dimensionality with a complex, non-linear underlying structure—as is the case with many natural language processing data sets. Additionally, there is a fair amount of ambiguity introduced by the nature of both the semantic word embeddings and the labeling process which may be very influenced by the labelling person.

[5] Although unnecessary for out implementation, this also granted the freedom to subset the data for labels differently—for example, if some labels treated null values differently from the others.

The following sections will compare the classification performance between models, target labels and the number of principal components.

5.1 Performance by model

While the highest validation F1 score was achieved by logistic regression, the MLP classifier performed better on average, both in terms of mean and median. Figure 4 shows the F1 score per model, indicating median score and the *interquartile range (IQR)* as box with whiskers showing the 1.5-fold of the IQR. Although the F1 scores of the classifiers are not very high, they all outperform the dummy classifier. The high performance of the MLP classifier can be explained by the model's ability to capture nonlinear feature dependencies, allowing it to capture the complexity of the data better.

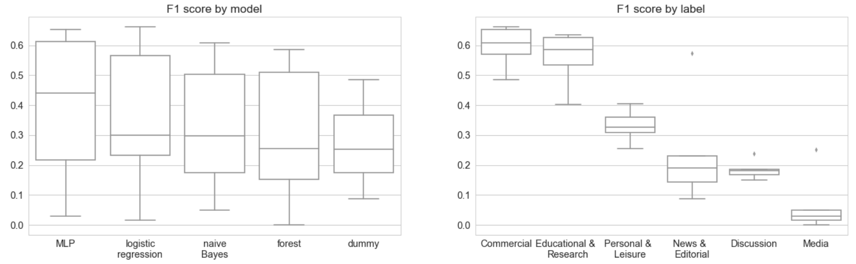

Figure 4: Overall performance per classifier and model when taking the best performance value across k.

5.2 Performance by label

Since balance differs greatly across labels, the classifiers perform very differently depending on the label. Unsurprisingly, the F1 score is higher on the more evenly distributed labels, which can be seen by comparing figure 4 and the class prior probabilities in figure 1. For example, *Media*, with just 1.5% positive labels, has a mean F1 score of 0.07, while the most balanced labels—*Commercial*, *Educational & Research*, and *Personal & Leisure*—have F1 scores above 0.30.

Figure 5 shows the maximim validation accuracies and F1 scores per classifier and target label. Given that the classes are imbalanced, the ordering of labels by accuracy and F1 score is almost opposite. The uneven probabilities of assigning a label are strongly biasing the model towards not assigning the label at all.

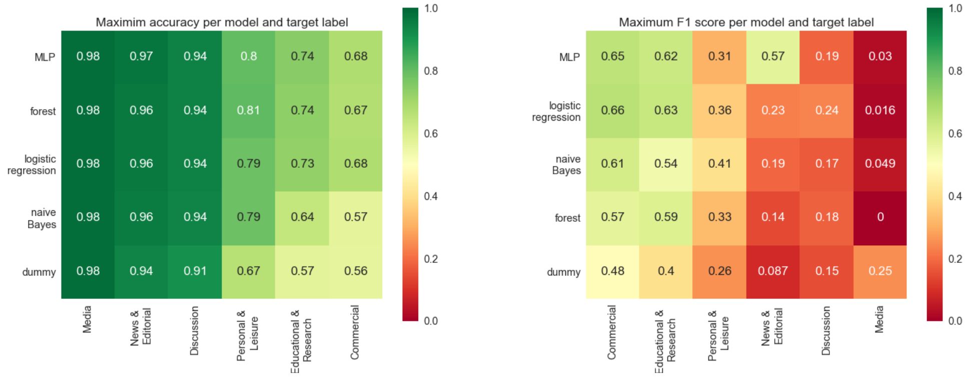

Figure 5: Maximum accuracy and F1 score per model and label. Only the best value of k is shown.

This effect can also be observed in the confusion matrices in figure 6, which are sorted by descending F1 score. The Labels *Commercial* and *Educational/Research* show the most confusion, but nevertheless yield the best results as they do classify some positive instances correctly. Conversely, no single *Media* website was identified as such, resulting in an F1 score of 0.

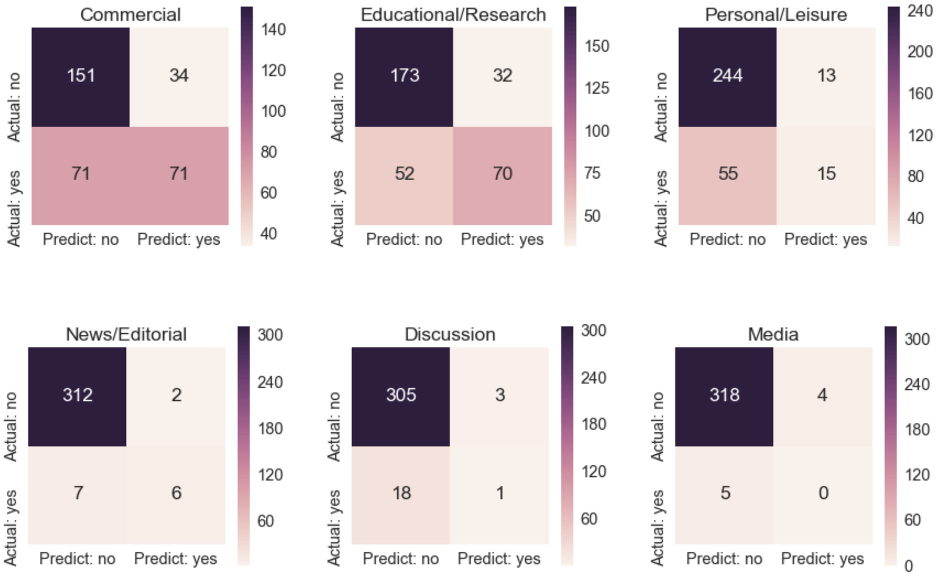

Figure 6: Confusion matrices for the MLP classifier on the validation set with $k = 100$ PCs and two hidden layers of 100 and 10 perceptrons, respectively.

5.3 Performance by number of principal components

The full trajectory of performance metrics with respect to k can be seen in figure 7, filtered on the label *Educational/Research*. The best value of k seems to be between 100 and 250, with the only exception being the Naive Bayes classifier which only starts to perform as well as the other classifiers when given a higher number of PCs. This, however, may strongly suggest that Naive Bayes is not a suitable model for this classification task.

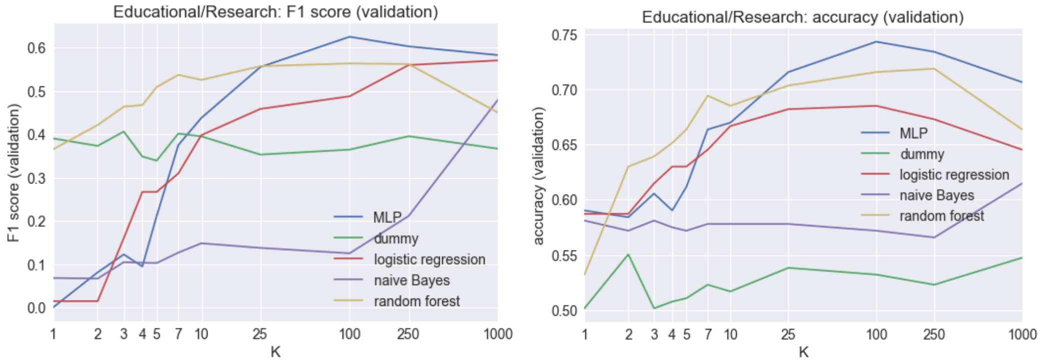

Figure 7: Maximum accuracy and F1 score per model, label and number of PCs

6 Conclusions

In this paper, we used the ECML/PKDD 2010 Discovery Challenge data set to conduct an exploratory data analysis and test several classifiers in the task of predicting the category label given to the websites by human annotators. To do so, we calculated the *tf-idf* as the embeddings for the websites, performed dimensionality reduction on these embeddings using incremental PCA, and visualized various aspects of the class labels and PCA variance. For the classification task, we trained and validated five models and analyzed the performance by label, model and number of PCs.

The analysis showed that the best combination of model, label and number of PCs was the logistic regression classifier, the *Commercial* label and $k = 250$ (respectively) when using F1 score as the performance metric. On average. however, the MLP classifier performed best. Performance by label was strongly driven by class balance, with the least balanced classes predictably having the highest accuracy but lowest F1 scores. Across labels and using the optimal number of PCs, all models performed better than the dummy classifier.

A more complete analysis may include adding the link-based features and performing hyperparameter optimization that is specific to each model (e.g., testing different learning rates for the MLP classifier), which was conducted in analyses by other researchers. There is also other label data—namely on the websites' neutrality, bias and trustworthiness—which can open up the analysis to additional classification tasks.

References

[1] András Benczúr, Carlos Castillo, Miklós Erdélyi, Zoltán Gyöngyi, Julien Masanes, and Michael Matthews. ECML/PKDD 2010 Discovery Challenge Data Set. Website available at: https://dms.sztaki.hu/en/letoltes/ecmlpkdd-2010-discovery-challenge-data-set. Crawled by the European Archive Foundation. Accessed 01 Mar 2017.

[2] Guang-Gang Geng, Xiao-Bo Jin, Xin-Chang Zhang, and De-Xian Zhang. Evaluating web content quality via multi-scale features. *arXiv preprint arXiv:1304.6181*, 2013.

[3] Elisabeth Lex, Inayat Khan, Horst Bischof, and Michael Granitzer. Assessing the quality of web content. *arXiv preprint arXiv:1406.3188*, 2014.

Sharing of work

Both members worked on the code to parse and prepare the data. They both wrote the report together, with Elias focusing on manipulating the data, creating graphs and conducting exploratory analysis, while Chris focused on writing the report and adding commentary to the graphs. We both worked together in the same room and made decisions regarding the paper together. It total, both members believe that the work was shared equally.