Machine learning approach for the binary classification of biomedical literature

Anna Price (PriceA35@cardiff.ac.uk)
Cardiff University https://orcid.org/0000-0002-0769-0417
Matthew Mort
Cardiff University
David N. Cooper
Cardiff University
Kevin E. Ashelford
Cardiff University

Technical advance

Keywords: machine learning, text mining, natural language processing

Posted Date: March 9th, 2020

DOI: https://doi.org/10.21203/rs.3.rs-16326/v1

TECHNICAL ADVANCE

Machine learning approach for the binary classification of biomedical literature

Anna Price[1*], Matthew Mort[2], David N. Cooper[2] and Kevin E. Ashelford[3]

*Correspondence:
PriceA35@cardiff.ac.uk
[1]School of Biosciences, Cardiff University, Cardiff, UK
Full list of author information is available at the end of the article

Abstract

Background: We have applied machine learning techniques to automate the screening of biomedical literature prior to the manual curation of clinical databases such as performed by the Human Gene Mutation Database (HGMD).

Methods: We have developed two machine learning models, one based on title and abstract data only, the other on the full text of the article. The models were built using a Natural Language Processing (NLP) pipeline and a logistic regression classifier. Our pipelines are implemented in Python and can be run using Docker. They are made available to the wider community via GitHub (https://github.com/annacprice/nlp-bio-tools) and Docker Hub.

Results: During testing, both models performed well, correctly predicting HGMD relevant articles more than 93% of the time and correctly discarding irrelevant articles more than 96% of the time, with Matthews Correlation Coefficients (MCC's) of over 0.89. Evaluation of the finalised model using an unseen validation dataset demonstrated that the full text model correctly predicted HGMD-relevant articles more than 97% of the time, an accuracy 9.5% higher than that obtained with the title/abstract model.

Conclusions: Through this work we have demonstrated that machine learning models can act as an effective pre-screen of biomedical literature, with the results indicating that a full text approach to screening biomedical literature is preferable to using just the title/abstract data.

Keywords: machine learning; text mining; natural language processing

Background

The ability to classify research articles at scale is an important pre-processing step for many databases seeking to curate and collate clinical articles to add value for re-

search and healthcare. For example, the Human Gene Mutation Database (HGMD) [1] must screen many tens of thousands of clinical research articles each year in order to collate all known gene lesions responsible for human inherited disease, along with disease-associated variants and functional polymorphisms and thereby providing a unique and important resource for diagnostic healthcare and clinical research. Trained curators undertake the time-consuming and laborious task of classifying the articles themselves. Until now, the unstructured nature of clinical articles has prevented an initial automated screen of the literature. However, the task of determining HGMD-relevant articles provides an ideal subject for machine learning.

Machine learning is a subfield of artificial intelligence. It uses algorithms to build models with the ability to automatically learn from data without explicit programming. Once built, these models are used to classify new data. Machine learning algorithms classify by inferring a function to map the input data to their discrete classes. The classification algorithms learn from training data and can be split into three main types: supervised [2], unsupervised [3] and reinforcement [4]. Supervised algorithms are trained using a labelled training set (i.e. the classes of the data are already known). Unsupervised algorithms are trained using an unlabelled training set and try to infer a function that describes the underlying structure of the training dataset. Reinforcement machine learning algorithms use a trial and error approach to building models; they learn from feedback from interactions with an external environment.

Machine learning is applicable to many domains e.g. from facial recognition on phones [5] to identifying disease-causing mutations from Next Generation Sequencing Data [6]. Natural language processing (NLP) represents a field within machine learning that involves the analysis of natural language. One common use of NLP is in the classification of textual data such as required during spam email detection [7]. Early work in text classification started in the 1960s [8], with machine learning approaches gaining popularity in the 1990s [9, 10]. NLP use has increased in recent years due to the growth in digital documents and computing power.

Text documents can be classified by methods based on Naive Bayes [11, 12, 13], k-nearest neighbours [11, 13], decision trees [13], support vector machines [9, 14, 15], neural networks [16, 17] and regression models [15, 18]. Comparisons between these

different classifiers can be widely found in literature [13, 19]. A detailed review of text classification is provided by Sebastiani, 2002 [10].

In recent years, there has been a marked increase in the text mining of biomedical literature. NLP has the potential to aid this research by extracting information from texts. For example, Wei, 2013 [20] uses text mining to extract sequence variants from biomedical literature. In another example, Li, 2015 [21] approaches the problem of using named entity recognition (NER) for recognising diseases and on how to extract information on chemical-induced disease relations.

Increasingly, biomedical text mining has been applied to precision medicine, which aims to tailor medical care to the individual [22]. For example, Liu, 2015 [23] demonstrates an online text mining system for identifying relationships between different biomedical entities such as genes, drugs and toxins. Singhal, 2016 [24] describes a tool for extracting relationships among disease-related mutations relationships from literature to support work in precision medicine. A review of past work in biomedical text mining, and discussions on its future can be found in Huang, 2015 [25] and Gonzalez, 2015 [26] respectively.

In this study, we outline the development of machine learning classification models, to assist the HGMD in the curation of their database. We have built two supervised machine learning models that act to pre-screen clinical literature from PubMed, to predict whether articles contain reports of human disease-causing mutations which would therefore be relevant to the HGMD. This is an example of binary classification, where a positive class represents scientific articles that are relevant to the HGMD whilst a negative class represents papers that are not of interest. A logistic regression [27] algorithm is used to build our models. Through the use of these machine learning models we aim to improve the coverage of the database and save on curator time. Currently, HGMD curators manually screen PubMed articles based on the title and abstract text alone. We investigate two models, one based on title and abstract data only, the other on the full text of the article. In doing so we determine whether there is benefit in using a full text approach that provides more data.

Although our models are built for HGMD use, our methods can be used to build models for the binary classification of any text based document. Our work is therefore relevant far beyond the immediate requirements of the HGMD. The ability to

classify unstructured text documents at scale is increasingly important for healthcare, not least because it enables the incorporation of a much wider range of patient data into anonymised databases that can then serve as a resource for precision medicine research and healthcare. For example, much useful information is embedded within clinician letters, electronic health records, and clinical reports, that combined with other data types such as molecular data, bioimaging data, socioeconomic data, and routine clinical data, would be useful for stratifying patients for more targeted healthcare. The use of machine learning techniques to classify clinical text, indeed any unstructured text has long been recognised, but with many algorithmic choices available, it is important to ensure that the technique used is appropriate and optimised for the task in hand. Furthermore, the unstructured nature of clinical records in general, and the clinical literature in particular, means that any machine learning application must be guided by relevant background clinical knowledge.

Our pipelines are freely available under an open-source licence and can be found on GitHub (https://github.com/annacprice/nlp-bio-tools). Our programs allow users to select different machine learning algorithms and vectorisers in order to build their own machine learning models. In addition, we have built ready-made docker images for each program, which can be found at Docker Hub (https://hub.docker.com/u/annacprice). The programs are provided in the hope that they will be beneficial to other researchers and act as a useful machine learning toolkit to benefit future research and healthcare delivery.

Methods

We prepared training, testing and validation datasets of research paper pdfs to train and evaluate our machine learning model. Each dataset contained papers categorised as either positive or negative. The positive papers, called the HGMD class, contained articles already present in the HGMD. The negative papers were subdivided into two further classes called COSMIC and RANDOM. The COSMIC class of papers are present in the Catalogue Of Somatic Mutations In Cancer (COSMIC) database [28] and contain somatic and mitrochondrial mutations that are irrelevant to the remit of the HGMD. Papers labelled RANDOM class, were a random collection of papers from PubMed, known to be irrelevant to both the HGMD and COSMIC.

To avoid training bias, [29] the training set was balanced with an equal number of positive and negative articles, with the negative articles further split equally between papers belonging to the COSMIC and RANDOM classes.

The testing and validation datasets were separately prepared to assess how well our model performed on unseen data. The testing dataset acted as an early evaluation of our model in the latter stages of training. The validation dataset serves as our final test data containing completely 'unseen' papers not used during the training process. The validation dataset was used on our finalised model to give an unbiased evaluation of the fit of our model.

The size of the datasets for training, testing and the final validation set are given in Table 1. The articles for the training dataset were randomly selected from the total available dataset using the Unix command shuf. The remaining articles were then used as a testing dataset. A training dataset of 11,200 articles was used, equally spilt between 5,600 positive and negative examples of articles. We chose this size to avoid oversampling the smaller COSMIC dataset composed of 4,940 and 2,981 articles for the full text and title/abstract datasets respectively.

It should be noted that the same articles were present in both the full text training set and the title/abstract training set.

Our machine learning model was built in Python 2.7 and uses the Natural Language Toolkit, NLTK [30], scikit-learn [31] and pdfminer [32] libraries. We produced three programs. The first is pdf2nlp which uses pdfminer to convert the pdf to utf-8 plain text and then passes the plain text through the NLP pipeline which uses NLTK. The second is mlpipe which uses scikit-learn to build the machine learning model. The final program is loadmodel which loads the saved machine learning model from mlpipe and uses it to evaluate new data. The scope of each program is detailed in Fig. 1. The full codebase can be found on GitHub at https://github.com/annacprice/nlp-bio-tools. Each program also has its own docker image, which can be used to spin up a docker container to run the program.

The workflow for building and using our machine learning model is given in Fig 1. There are two paths to our workflow: one for training and one for prediction. The first two steps of the workflow are the same for both training and prediction. We first extracted the text from the pdf articles. The text was then passed through our NLP Pipeline. In training our classifier, we first selected the features that we

wanted the classifier to fit to. Once extracted from the text, the features were used to build a term-document matrix. This was then passed to the classifier, along with the class labels for each document, for training. For the prediction workflow, we fit to the features selected during training, and the resulting term-document matrix was passed to the trained model to output the predicted class for each document.

Natural Language Processing (NLP) Pipeline

For text classification problems, the main issue is how we choose to represent our document, as the model's accuracy depends greatly on the quality of the input data. The aim of the NLP pipeline in pdf2nlp is to produce a clean and concise representation of each document. Each stage of the NLP pipeline is given in Fig. 2.

The final output of the NLP pipeline is a collection of stemmed words, or tokens, for each document. Stemming reduces words to their base form. The use of stemming in information retrieval tasks solves the problem of vocabulary mismatch, ensuring variant words which have the same stem (and therefore similar meanings) are counted as the same token. This improves the precision and recall of a machine learning system [33]. It also has the added benefit of shrinking the size of the corpus, reducing both storage space and the size of the term-document matrix which needs to be calculated.

We stemmed our tokens using the snowball stemmer [34], part of the NLTK [30]. We then used feature selection to select which tokens the classifier would fit to, and built a term-document matrix, which acted as a numerical representation of the abundance of features in each document.

Feature Selection

Feature selection defines which tokens our algorithm fits to. The selected tokens need to represent our data well and contain enough information for the model to predict the output accurately. Feature selection allows the number of tokens for text classification problems to be reduced by the tens of thousands. This is important because when the input feature vectors are too large this can result in the model struggling to fit to the data because of high variance.

Many methods for feature selection exist [35], including: bag-of-words (i.e. basic term frequency counting) [36, 37], tf-idf (term frequency-inverse document fre-

quency) [38, 39], information gain [40] and chi-square [41]. A study of the effectiveness of different feature selection methods can be found in Forman, 2003 [35].

We used tf-idf to select our features. From the results of the tf-idf weighting, we selected the top 600 features. In our implementation of tf-idf, we included binary terms alongside singular tokens when selecting the top features for fitting. The tf-idf weight is comprised of two terms: the term-frequency (TF) and the inverse document frequency (IDF). The term frequency is simply the number of times a term appears in an individual document. This is weighted by the inverse document frequency which weighs down highly frequent terms (such terms are often insignificant words such as auxiliary and modal verbs). The tf-idf weighting was implemented using scikit-learn. To reduce document-length bias each term's tf-idf score is proportionally scaled by dividing by the squared average of the document.

Note that, because features were selected by applying tf-idf to the training sets for full text and title/abstract separately, the full text and title/abstract corpuses were inevitably fitted to different features.

Term-Document Matrix

Hashing creates a numerical representation of the features to pass to the classifier. As we reduced our features to only 600, we simply used a direct mapping to build a term-document matrix (or hash table), that represented the weight of each feature in each document. For example, for a matrix X, the matrix element x_{ij} represents the weight of the term/feature j in the document i. For larger corpuses it is often necessary to implement methods such as the hashing trick to scale-up machine learning algorithms [42]. Note that the same hash table is used for training and prediction, i.e. we fit to the same features.

Building a machine learning model

Machine learning algorithms generalise from training data to make accurate observations on new, unseen, data. Algorithm choice is therefore crucial. Several models should be tested in order to find which works best. For text classification, algorithms such as Naive Bayes, logistic regression and nearest-neighbour models have been shown to perform well [10]. In the preliminary stages of building our model, we investigated all three algorithms and found logistic regression to be most effective

(Fig. 3). However, the difference between the classifiers is marginal, suggesting that in our case feature selection is more important than choice of model.

Formally, our training set is given by $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_n, y_n)\}$ where n is the total number of documents. Each document is represented by a feature vector $\mathbf{x}_i = (x_{i1}, ..., x_{if})^T$, where f is the total number of features and $i = 1, ..., n$. Each feature vector belongs to a binary class $y_i \, \epsilon \, \{0, 1\}$, where 0 is the negative class (not HGMD) and 1 is the positive class (HGMD). If X is the set of all documents x, then the goal of the machine learning algorithm is to infer this mapping $f : X \rightarrow \{0, 1\}$.

Logistic regression estimates the log odds of an event with a binary outcome. Defining the parameter vector $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_f)$, then the probability of a document belonging to the positive class (1) is

$$p(y = 1|\mathbf{x}_i; \ \boldsymbol{\beta}) = \frac{1}{1 + exp(-\boldsymbol{\beta}^T . \ \mathbf{x}_i)} \tag{1}$$

For logistic regression, learning the inferred mapping function is equivalent to finding the parameter vector which maximises the log-likelihood function for the training set. The log-likelihood function is given by [43, 18]

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} [y_i . \ \boldsymbol{\beta}^T . \ \mathbf{x}_i - log(1 + exp(\boldsymbol{\beta}^T . \ \mathbf{x}_i))] \tag{2}$$

Once the algorithm has determined the best parameter vector, the model can be saved and used to classify unseen documents.

Results

We compare the results from the title/abstract and full text classifiers by analysing receiver operator characteristic (ROC) curves, percentage accuracy, Matthews correlation coefficients (MCC) [44], precision and recall, and F1-score. The ROC curves for the title/abstract model are given in Fig. 4 for both the (a) training and (b) testing datasets. The ROC is a plot of the true positive rate (TPR) vs the false positive rate (FPR) for various predictions of the model at different thresholds between 0 and 1. The TPR, or sensitivity, is defined as the number of true positives (TP) divided by the sum of the number of true positives and the number of false

negatives (FN), hence $TP/(TP + FN)$. The false positive rate (FPR) is defined as the number of false positives (FP) divided by the sum of the number of false positives and the number of true negatives (TN), $FP/(FP + TN)$. FPR is also sometimes referred to as $1 - specificity$. On average, a skilled model will assign a higher probability to a random true positive occurrence than a true negative (curves bow up to the top left of the plot). The area under the ROC curve (AUC) indicates how likely it is that the model will predict a higher probability for the true positive cases than the true negative cases. The larger the AUC, the greater the skill of the model, with an AUC of 1.0 indicating a perfect model.

Fig. 4a shows the k-fold cross validation of the training set for the title/abstract dataset. In the early stages of training, we used k-fold cross-validation as a resampling method to estimate the skill of the model on new data, using just the training dataset. The dataset was randomly divided into k groups. The classifier then used (k-1)/k of the set for training holding back 1/k for testing, iterating through the entire dataset so each fraction of the dataset was used as a testing set once and was used to train the model (k-1) times. The k groups are constant for the entire process. We chose a value of k=10, as it has been shown to give test error-estimates that have neither an excessively high bias nor a high variance [45].

After the initial stages of training, we used a testing dataset to evaluate the performance of the model. Fig. 4b shows the ROC for the testing dataset. Note that for Fig. 4b we have randomly selected an equal number of papers from the positive and negative datasets (with the negative dataset further equally split between COSMIC and RANDOM) to produce a balanced dataset required for ROC curves.

The corresponding ROC curves for the full text dataset are not shown as they produce almost identical plots to those shown in Fig. 4. For the training set, both our title/abstract and full text models had AUC values of over 0.98 for each fold, indicating that the models are overfitting to the training set. However, despite this the models still performed well on evaluation of the testing set, with both models again achieving an AUC of over 0.98. The percentage accuracy for each class of the testing set is given in Table 2. The title/abstract and full text models show similar performance, with the COSMIC datasets showing around 97% accuracy and the RANDOM datasets showing around 96% accuracy. The title/abstract model showed slight improvements for the HGMD class, with a 94.5% accuracy, a 1.4%

improvement over the full text model. However, bias is possible as the datasets were built by screening only the title/abstract data and not the full text of the article. Hence, one might expect to see improvements in the title/abstract model versus the full text model.

The predicted probability for a binary classification problem can be interpreted with a threshold defining whether an article will be classified as negative or positive. The default threshold in machine learning problems is 0.5, with probability in the range of $[0, 0.5)$ indicating that the article belongs to the negative class (i.e. not HGMD), and a probability in the range $[0.5, 1.0]$ indicating that the article belongs to the positive class (belongs to the HGMD). By changing the threshold, we can tune the model for different rates of false positives vs. false negatives.

Varying the threshold is important in problems where either false positives or false negatives are more important than the other, or in models where there is disproportionately one type of error over the other. We calculated the MCC for various thresholds between 0.4 and 0.6 and the results are presented in Table 3. The MCC has a range from -1 to 1, where a value of -1 would indicate a classifier which is completely wrong and a value of 1 would indicate a completely correct classifier. The MCC acts as a balanced measure of the fit of the model and can be used for datasets were the classes are different sizes. For both models, the training and testing datasets all show an MCC of at least 0.88 at each threshold, indicating that the models are fitting well to both the training and the testing datasets. As the difference between the MCC for different thresholds is negligible we kept the default threshold of 0.5.

Table 4 shows the precision, recall, and the F1 score for the training and testing datasets. For the positive class, the precision is the ratio of true positives to the total predicted positives $TP/(TP+FP)$. The recall (or sensitivity) is then the ratio of true positives to the total of true positives and false negatives $TP/(TP+FN)$. A high precision and low recall for the positive class would indicate a greater number of false negatives than false positives. A low precision and high recall for the negative class would also indicate the same. The F1-score acts as a weighted average of the precision and recall $(2R + 2P)/(R + P)$. The precision and recall act as useful evaluations of imbalanced datasets (i.e. datasets with different class sizes) such as our testing dataset.

For the training dataset, the precision and recall are around the same for both models. In particular, the precision and recall for each model (and hence the F1-score) are almost equal, indicating an equal number of false positives and false negatives. This is confirmed by the confusion matrices (a) and (b) in Fig. 5. The high precision and recall of greater than 0.95 indicates that the model has few false results.

For the testing datasets, both the title/abstract and full text model show a difference in the values for precision and recall. For the positive class, the precision is greater than the recall, and for the negative set the recall is greater than the precision. This indicates that, for the testing set, there are a greater number of false negatives than false positives. This is confirmed by the confusion matrices (c) and (d) in Fig. 5. For the full text dataset, the number of false negatives is almost 3 times greater than the number of false positives, and for the title/abstract dataset the number of false negatives is almost 2 times greater.

In terms of percentage accuracy, the title/abstract and full text models showed very similar performance. However despite this, there were differences in the features the models fitted to. In total, of the 600 features that each model fitted to, the models shared 477 features in common. Fig. 6 shows the top 20 features as ranked by the logistic regression classifier for both positive and negative class for the two models. For the top 20 features of the positive class, the two models share 15 features. For the top 20 features of the negative class, the two models share 14 features

Overall, the full text model did show improved feature selection over the title/abstract model i.e. fitting to more significant tokens such as intron and splice, whereas the title/abstract model was more prone to fitting to less significant verbs such as cause and detect. This indicates the possible advantages of the full text approach for building more complex models and implies that the full text screening approach is more likely to produce accurate models than the screening of the title/abstract alone.

Validation Dataset

Once we had finished training and testing the two models, we created a finalised model for each by passing all of the available training and testing data to the

classifier for training, fitting to the same 600 features as before. We then evaluated the performance of these models using a final unseen validation dataset, containing papers which were not used for either training or testing. The validation dataset consists of papers from the HGMD and RANDOM negative papers from PubMed.

The percentage accuracy of classification for the validation dataset is given in Table 5. For the RANDOM class, the title/abstract and full text model are both around 94% accurate. However, for the HGMD class the percentage accuracy is 97.3% for the full text model, but only 87.8% for the title/abstract model. This indicates that the use of full text data as opposed to just the title/abstract of a paper, is advantageous and provides better features for the machine learning algorithm to fit to.

The precision, recall and f1-score for the validation dataset are given in Table 6. The precision and recall for both models are high, indicating there are few false results. For the title/abstract model, the precision and recall are almost equal in both the HGMD and RANDOM classes. This indicates a similar number of false positives (FP) and false negatives (FN). This result is confirmed by the confusion matrix in Fig.7a which shows the number of FP and FN are 12 and 11 respectively. The precision and recall are greater for the RANDOM class, indicating the higher accuracy for the RANDOM class. The full text model exhibits a significant difference between the precision and recall for the HGMD class, with a precision of 0.84 and a recall of 0.97. This indicates a far greater number of FP than FN, which is confirmed by the confusion matrix in Fig. 7b. The number of FP is 21, but the number of FN is only 3.

Discussion

Aside from solving our immediate challenge of how to automate the pre-selection of research articles for further investigation by HGMD curators, we have outlined the general methods for building a supervised machine learning model for any binary classification of biomedical research articles. Indeed, our approach can be used to classify many types of text document and this makes it a potentially powerful tool for rapidly screening all sorts of healthcare data.

The ability to classify unstructured text documents at scale enables routinely collected patient data to be incorporated into databases that can serve as a resource

for precision medicine research and healthcare. Binary classification methods of the kind we showcase here will therefore play an important role in helping to create clinical data resources of the future.

Our pipelines have been uploaded to GitHub (https://github.com/annacprice/nlp-bio-tools) and Docker Hub (https://hub.docker.com/u/annacprice) with the hope they provide useful to other researchers. A short wiki is also provided on GitHub (https://github.com/annacprice/nlp-bio-tools/wiki) to assist researchers in using the software to build their own machine learning models.

Conclusions

We produced two models for classifying papers into the HGMD, both based on logistic regression: one for title/abstract data and one for full text data. During testing, both models performed well, correctly predicting HGMD-relevant articles more than 93% of the time and discarding irrelevant articles more than 96% of the time, with MCC's of over 0.89. However, on analysis of features that both models were fitted to it was found that the full text model was fitting to more significant features, indicating that using the full text data should provide greater accuracy. This was confirmed on evaluation of the final model using a validation dataset, the full text model showed significant improvement over the title/abstract model, correctly predicting HGMD-relevant articles more than 97% of the time, showing a 9.5% improvement on the title/abstract model. This along with the more significant features, indicates that a full text approach would be preferable when building more sophisticated models.

List of abbreviations

HGMD = Human Gene Mutation Database

NLP = Natural Language Processing

COSMIC = Catalogue Of Somatic Mutations In Cancer

NLTK = Natural Language ToolKit

TF-IDF = Term Frequency-Inverse Document Frequency

ROC = Receiver Operator Characteristic

MCC = Matthews Correlation Coefficients

TP = True Positive

FP = False Positive

383 TN = True Negative

384 FN = False Negative

385 TPR = True Positive Rate

386 FPR = False Positive Rate

387

Declarations

Ethics approval and consent to participate

Not applicable

Consent for publication

Not applicable

Availability of data and materials

The software used to build the machine learning models and the finalised machine learning models can be found at `https://github.com/annacprice/nlp-bio-tools`.

Competing interests

The authors declare that they have no competing interests.

Funding

AP is funded by the Supercomputing Wales project, which is part-funded by the European Regional Development Fund (ERDF) via Welsh Government. DNC's and MM's work with the HGMD is financially supported by the QIAGEN plc through a License Agreement with Cardiff University. KEA is funded by Welsh Government through Health and Care Research Wales via Wales Gene Park, a research infrastructure support group embedded within Cardiff University.

Authors' contributions

MM, DNC and KEA conceived and planned the project. MM prepared the datasets used to build the machine learning models. AP wrote the software, built the machine learning models and performed the analysis on the models' performance. AP, KEA and MM wrote the manuscript. All authors have read and approved the manuscript.

Acknowledgements

Not applicable

Author details

[1]School of Biosciences, Cardiff University, Cardiff, UK. [2]School of Medicine, Institute of Medical Genetics, Cardiff University, Cardiff, UK. [3]School of Medicine, Wales Gene Park, Cardiff University, Cardiff, UK.

References

1. Stenson, P.D., Mort, M., Ball, E.V., Evans, K., Hayden, M., Heywood, S., Hussain, M., Phillips, A.D., Cooper, D.N.: The human gene mutation database: towards a comprehensive repository of inherited mutation data for medical research, genetic diagnosis and next-generation sequencing studies. Human genetics **136**(6), 665–677 (2017)
2. Kotsiantis, S.B., Zaharakis, I., Pintelas, P.: Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering **160**, 3–24 (2007)
3. Hofmann, T.: Unsupervised learning by probabilistic latent semantic analysis. Machine learning **42**(1-2), 177–196 (2001)
4. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT press, ??? (2018)
5. Tang, Y.: Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239 (2013)

6. Capriotti, E., Altman, R.B.: A new disease-specific machine learning approach for the prediction of cancer-causing missense variants. Genomics **98**(4), 310–317 (2011)
7. Rădulescu, C., Dinsoreanu, M., Potolea, R.: Identification of spam comments using natural language processing techniques. In: 2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 29–35 (2014). IEEE
8. Maron, M.E.: Automatic indexing: an experimental inquiry. Journal of the ACM (JACM) **8**(3), 404–417 (1961)
9. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: European Conference on Machine Learning, pp. 137–142 (1998). Springer
10. Sebastiani, F.: Machine learning in automated text categorization. ACM computing surveys (CSUR) **34**(1), 1–47 (2002)
11. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 42–49 (1999). ACM
12. Rennie, J.D., Shih, L., Teevan, J., Karger, D.R.: Tackling the poor assumptions of naive bayes text classifiers. In: Proceedings of the 20th International Conference on Machine Learning (icml-03), pp. 616–623 (2003)
13. Li, Y.H., Jain, A.K.: Classification of text documents. The Computer Journal **41**(8), 537–546 (1998)
14. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. Journal of machine learning research **2**(Nov), 45–66 (2001)
15. Zhang, T., Oles, F.J.: Text categorization based on regularized linear classification methods. Information retrieval **4**(1), 5–31 (2001)
16. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: AAAI, vol. 333, pp. 2267–2273 (2015)
17. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems, pp. 649–657 (2015)
18. Ifrim, G., Bakir, G., Weikum, G.: Fast logistic regression for text categorization with variable-length n-grams. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 354–362 (2008). ACM
19. Singh, A., Thakur, N., Sharma, A.: A review of supervised machine learning algorithms. In: Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference On, pp. 1310–1315 (2016). IEEE
20. Wei, C.-H., Harris, B.R., Kao, H.-Y., Lu, Z.: tmvar: a text mining approach for extracting sequence variants in biomedical literature. Bioinformatics **29**(11), 1433–1439 (2013)
21. Li, J., Sun, Y., Johnson, R., Sciaky, D., Wei, C.-H., Leaman, R., Davis, A.P., Mattingly, C.J., Wiegers, T.C., Lu, Z.: Annotating chemicals, diseases, and their interactions in biomedical literature. In: Proceedings of the Fifth BioCreative Challenge Evaluation Workshop, pp. 173–182 (2015)
22. Collins, F.S., Varmus, H.: A new initiative on precision medicine. New England Journal of Medicine **372**(9), 793–795 (2015)
23. Liu, Y., Liang, Y., Wishart, D.: Polysearch2: a significantly improved text-mining system for discovering associations between human diseases, genes, drugs, metabolites, toxins and more. Nucleic acids research **43**(W1), 535–542 (2015)
24. Singhal, A., Simmons, M., Lu, Z.: Text mining for precision medicine: automating disease-mutation relationship extraction from biomedical literature. Journal of the American Medical Informatics Association **23**(4), 766–772 (2016)
25. Huang, C.-C., Lu, Z.: Community challenges in biomedical text mining over 10 years: success, failure and the future. Briefings in bioinformatics **17**(1), 132–144 (2015)
26. Gonzalez, G.H., Tahsin, T., Goodale, B.C., Greene, A.C., Greene, C.S.: Recent advances and emerging applications in text and data mining for biomedical discovery. Briefings in bioinformatics **17**(1), 33–42 (2015)
27. Hosmer Jr, D.W., Lemeshow, S., Sturdivant, R.X.: Applied Logistic Regression vol. 398. John Wiley & Sons, ??? (2013)
28. Forbes, S.A., Beare, D., Gunasekaran, P., Leung, K., Bindal, N., Boutselakis, H., Ding, M., Bamford, S., Cole, C., Ward, S., *et al.*: Cosmic: exploring the world's knowledge of somatic mutations in human cancer. Nucleic

acids research **43**(D1), 805–811 (2014)
29. Provost, F.: Machine learning from imbalanced data sets 101. In: Proceedings of the AAAI 2000 Workshop on Imbalanced Data Sets, pp. 1–3 (2000)
30. Bird, S., Loper, E.: Nltk: the natural language toolkit. In: Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, p. 31 (2004). Association for Computational Linguistics
31. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al.*: Scikit-learn: Machine learning in python. Journal of machine learning research **12**(Oct), 2825–2830 (2011)
32. Shinyama, Y.: PDFMiner: Python PDF parser and analyzer (2014)
33. Aphinyanaphongs, Y., Fu, L.D., Li, Z., Peskin, E.R., Efstathiadis, E., Aliferis, C.F., Statnikov, A.: A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization. Journal of the Association for Information Science and Technology **65**(10), 1964–1987 (2014)
34. Porter, M.F.: Snowball: A language for stemming algorithms (2001)
35. Forman, G.: An extensive empirical study of feature selection metrics for text classification. Journal of machine learning research **3**(Mar), 1289–1305 (2003)
36. Harris, Z.S.: Distributional structure. Word **10**(2-3), 146–162 (1954)
37. Bekkerman, R., Allan, J.: Using bigrams in text categorization. Technical report, Technical Report IR-408, Center of Intelligent Information Retrieval, UMass (2004)
38. Robertson, S.: Understanding inverse document frequency: on theoretical arguments for idf. Journal of documentation **60**(5), 503–520 (2004)
39. Ramos, J., *et al.*: Using tf-idf to determine word relevance in document queries. In: Proceedings of the First Instructional Conference on Machine Learning, vol. 242, pp. 133–142 (2003)
40. Lee, C., Lee, G.G.: Information gain and divergence-based feature selection for machine learning-based text categorization. Information processing & management **42**(1), 155–165 (2006)
41. Chen, Y.-T., Chen, M.C.: Using chi-square statistics to measure similarities for text categorization. Expert systems with applications **38**(4), 3085–3090 (2011)
42. Weinberger, K., Dasgupta, A., Attenberg, J., Langford, J., Smola, A.: Feature hashing for large scale multitask learning. arXiv preprint arXiv:0902.2206 (2009)
43. Albert, A., Anderson, J.A.: On the existence of maximum likelihood estimates in logistic regression models. Biometrika **71**(1), 1–10 (1984)
44. Matthews, B.W.: Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure **405**(2), 442–451 (1975)
45. Kohavi, R., *et al.*: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Ijcai, vol. 14, pp. 1137–1145 (1995). Montreal, Canada

Figure legends

Figure 1 Workflow summarising the three programs used to run our analyses, showing the relationships between the training and prediction workflows. Each dotted box represents a separate program.

Figure 2 Workflow showing the natural language processing pipeline implemented in pdf2nlp, using NLTK. Text is separated into individual tokens and passed through a US-English to British-English dictionary. We want our model to fit only to English words, so tokens such as punctuation, numbers, and gene symbols are removed. Common stopwords (e.g. a, and, or, the) are also removed as they have no semantic value. Words that are common to all academic articles (e.g. department, school, abstract, references) are also removed, as well as journal names and common names. Finally, the remaining tokens are stemmed.

Figure 3 ROC curves for the testing dataset for: logistic regression, 5-nearest-neighbours, Multinomial Naive Bayes and Bernoulli Naive Bayes classifiers.

Figure 4 ROC curves for the title/abstract dataset for (a) 10-fold cross-validation of the training dataset and (b) testing dataset.

Figure 5 Confusion matrices showing the number of classified articles which were True Negatives (TN), False Positives (FP), False Negatives (FN) and True Positives (TP) for the (a) full text training dataset, (b) title/abstract training dataset, (c) full text testing dataset and (d) title/abstract testing dataset

Figure 6 Top 20 positive and negative features as ranked by the logistic regression classifier for the (a) title/abstract and (b) full text model. Blue indicates the positive class and red indicates the negative class.

Figure 7 Confusion matrices for the (a) title/abstract validation dataset and (b) full text validation dataset.

Tables

Table 1 Size of datasets available for training, testing and final validation

	TRAINING		TESTING		VALIDATION	
	Title/Abstract	Full Text	Title/Abstract	Full Text	Title/Abstract	Full Text
HGMD (+)	5600	5600	15697	26934	90	112
Cosmic (-)	2800	2800	181	2140	-	-
Random (-)	2800	2800	11714	18016	224	328

Table 2 Percentage accuracy of classification of the testing datasets for each class for a threshold of 0.5

TESTING		
	Title/Abstract	Full Text
HGMD (+)	94.5	93.1
COSMIC (-)	97.2	97.2
RANDOM (-)	96.1	96.7

Table 3 Matthews correlation coefficient at different thresholds for the training and testing datasets

	TRAINING		TESTING	
THRESHOLD	Title/Abstract	Full Text	Title/Abstract	Full Text
0.4	0.918	0.931	0.909	0.898
0.45	0.922	0.933	0.905	0.897
0.5	0.921	0.933	0.903	0.893
0.55	0.918	0.928	0.898	0.887
0.6	0.909	0.922	0.909	0.881

Table 4 Precision, recall and f1-score for the training and testing datasets for a threshold of 0.5

TRAINING						
	Title/Abstract			Full Text		
CLASS	Precision	Recall	F1-score	Precision	Recall	F1-score
Positive (HGMD)	0.97	0.95	0.96	0.97	0.97	0.97
Negative (Not HGMD)	0.96	0.97	0.96	0.97	0.97	0.97
TESTING						
	Title/Abstract			Full Text		
CLASS	Precision	Recall	F1-score	Precision	Recall	F1-score
Positive (HGMD)	0.97	0.94	0.96	0.97	0.93	0.95
Negative (Not HGMD)	0.93	0.96	0.95	0.91	0.97	0.94

Table 5 Percentage accuracy of classification for the validation dataset for each class for a threshold of 0.5

VALIDATION		
	Title/Abstract	Full Text
HGMD (+)	87.8%	97.3%
RANDOM (-)	94.6%	93.6%

Table 6 Precision, recall and f1-score for the validation datasets for a threshold of 0.5

VALIDATION						
	Title/Abstract			Full Text		
CLASS	Precision	Recall	F1-score	Precision	Recall	F1-score
Positive (HGMD)	0.87	0.88	0.87	0.84	0.97	0.90
Negative (Not HGMD)	0.95	0.95	0.95	0.99	0.94	0.96

Figures

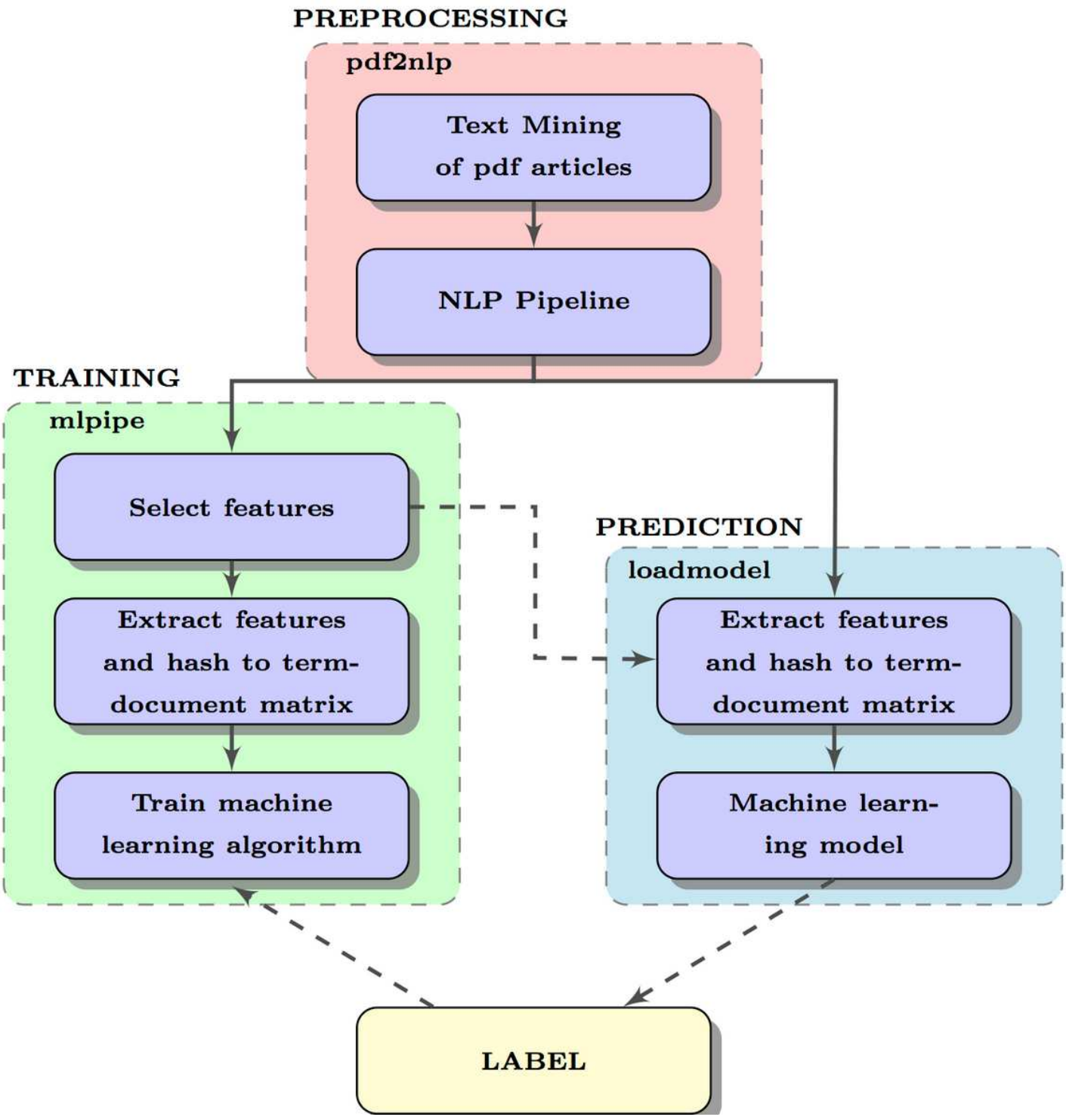

Figure 1

Workow summarising the three programs used to run our analyses, showing the relationships between the training and prediction workows. Each dotted box represents a separate program.

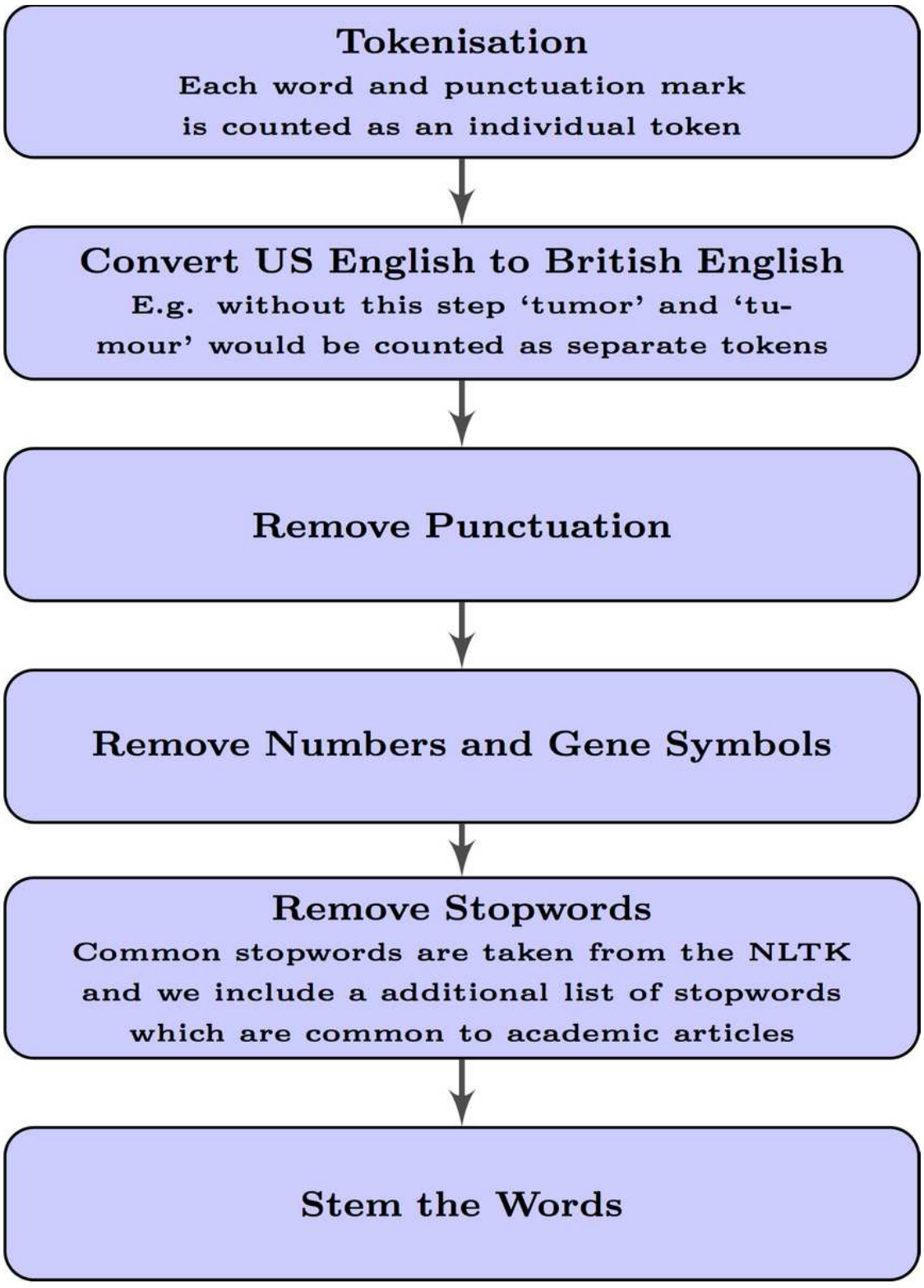

Figure 2

Workow showing the natural language processing pipeline implemented in pdf2nlp, using NLTK. Text is separated into individual tokens and passed through a US-English to British-English dictionary. We want our model to _t only to English words, so tokens such as punctuation, numbers, and gene symbols are removed. Common stopwords (e.g. a, and, or, the) are also removed as they have no semantic value. Words that are common to all academic articles (e.g. department, school, abstract, references) are also

removed, as well as journal names and common names. Finally, the remaining tokens are stemmed. Price et al. Page 17 of 18

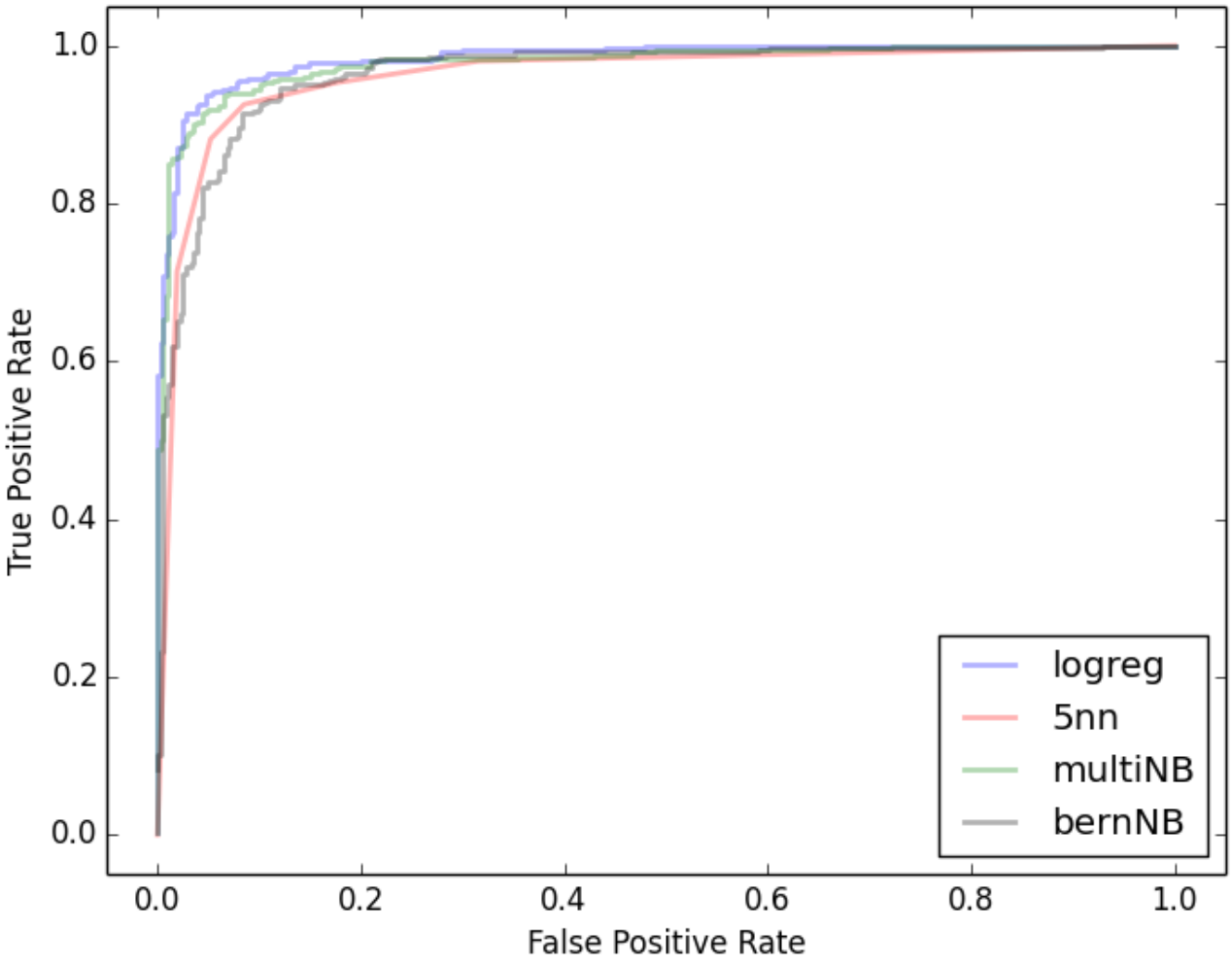

Figure 3

ROC curves for the testing dataset for: logistic regression, 5-nearest-neighbours, Multinomial Naive Bayes and Bernoulli Naive Bayes classi_ers.

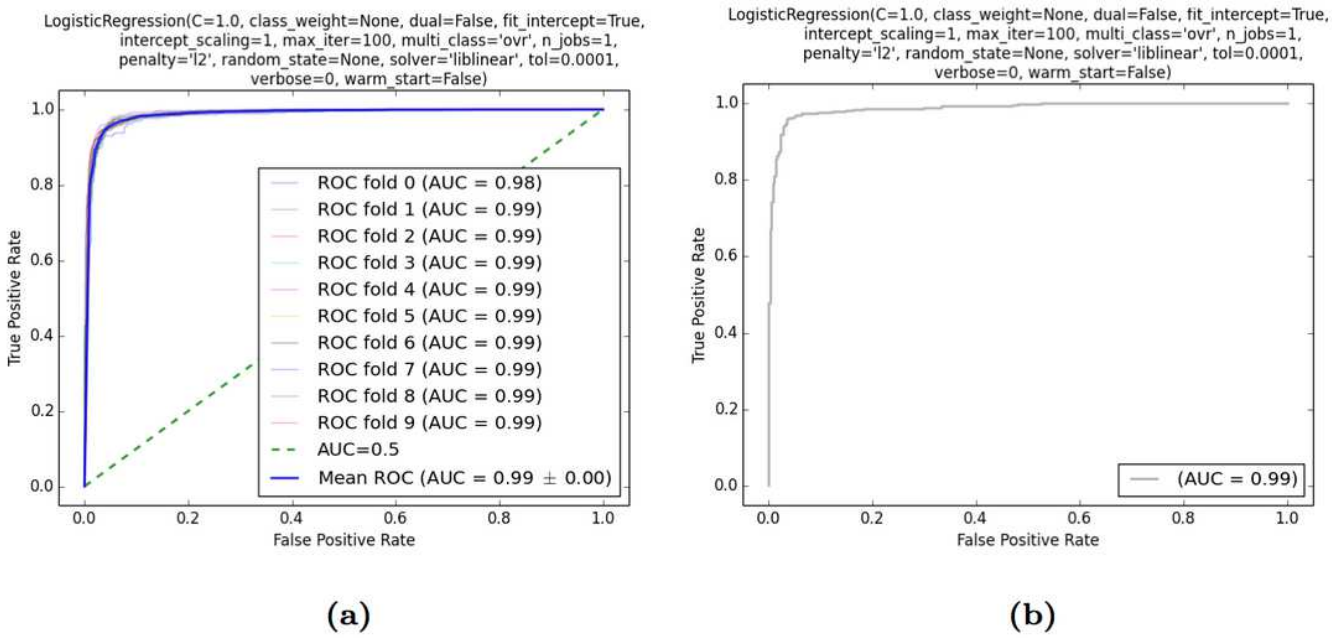

Figure 4

ROC curves for the title/abstract dataset for (a) 10-fold cross-validation of the training dataset and (b) testing dataset.

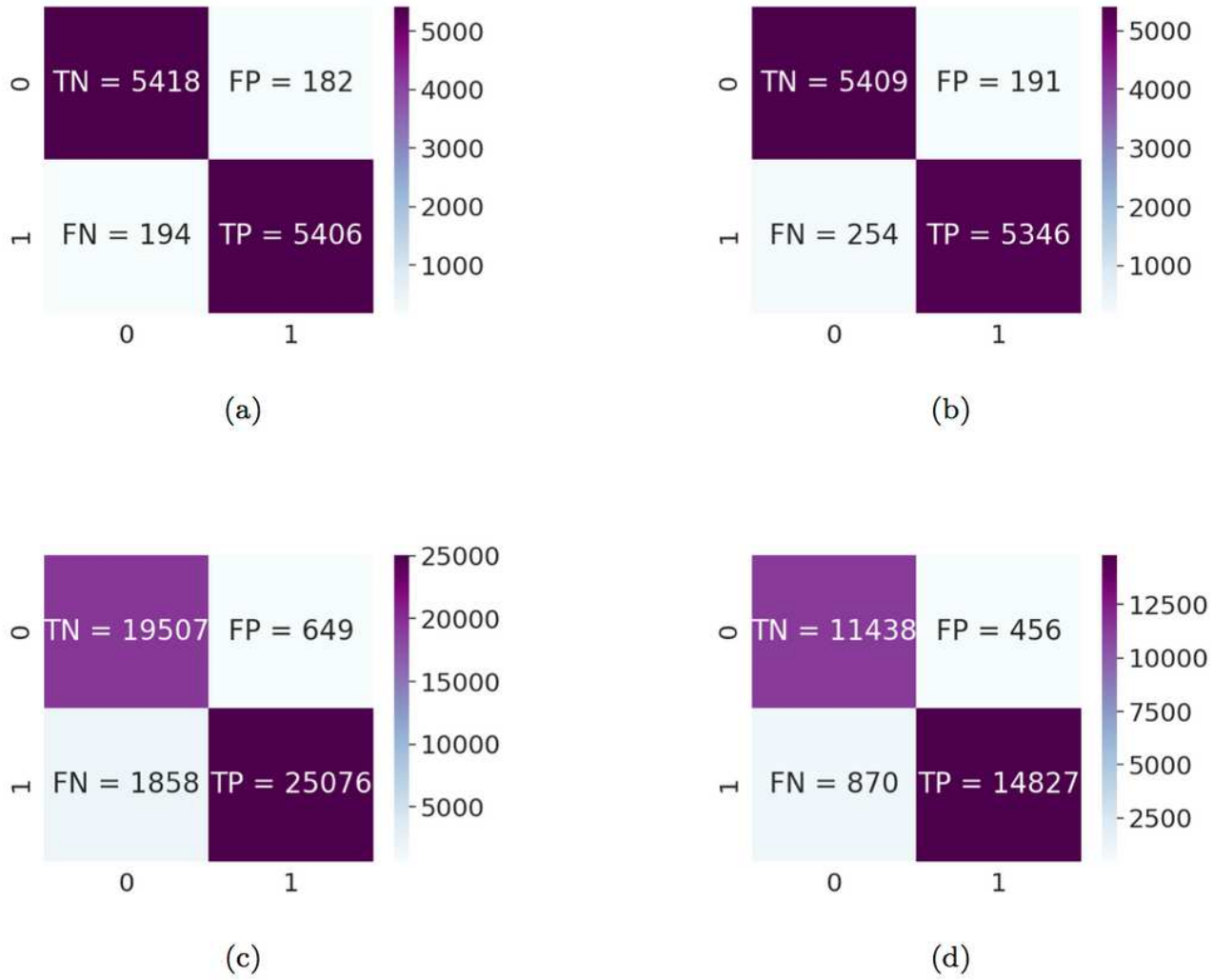

Figure 5

Confusion matrices showing the number of classi_ed articles which were True Negatives (TN), False Positives (FP), False Negatives (FN) and True Positives (TP) for the (a) full text training dataset, (b) title/abstract training dataset, (c) full text testing dataset and (d) title/abstract testing dataset

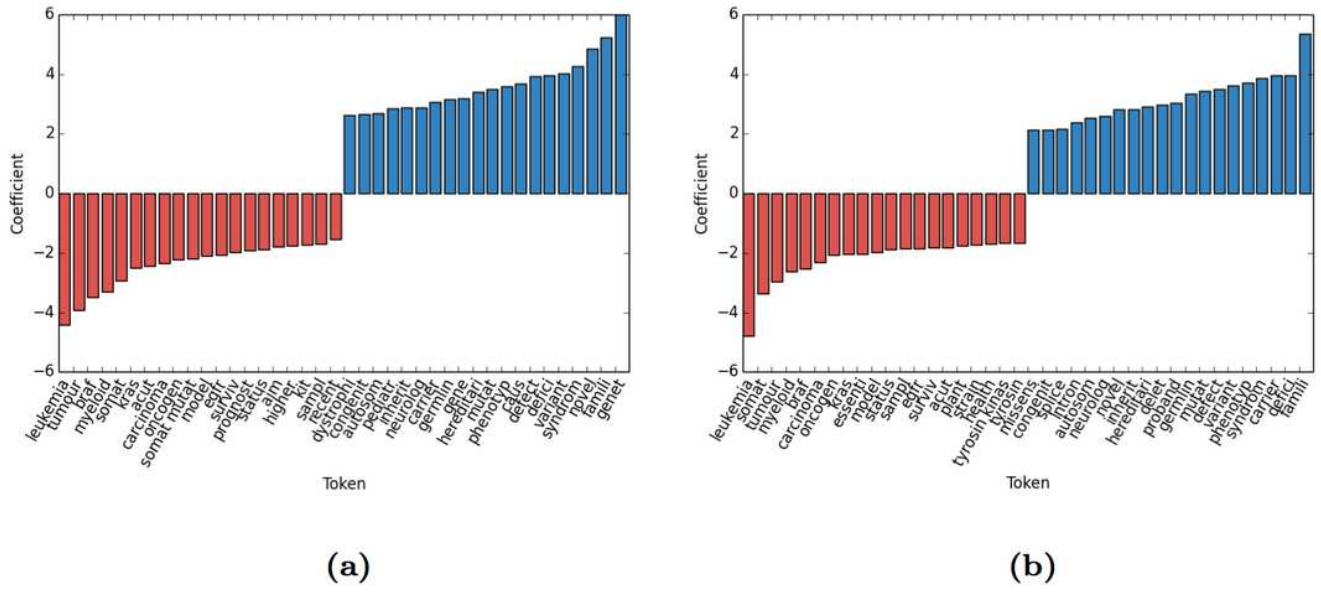

Figure 6

Top 20 positive and negative features as ranked by the logistic regression classi_er for the (a) title/abstract and (b) full text model. Blue indicates the positive class and red indicates the negative class.

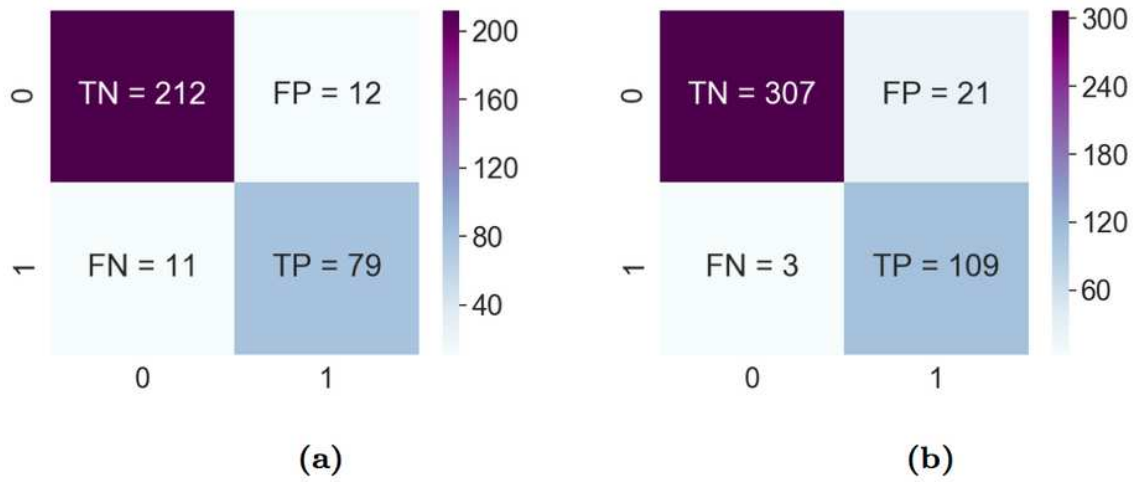

Figure 7

Confusion matrices for the (a) title/abstract validation dataset and (b) full text validation dataset.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- MLapproachbinaryclass.tex