Gender Classification of Blog Authors

Ibrahim Mohammad
MS in Computer Science
University of Houston
imohammad@uh.edu (1618923)

Abstract

Gender classification is one of the most interesting and challenging problems in many commercial domains. Existing systems mainly use features such as standard word, n-gram and POS (part of speech) for classification. And, Feature selection using filter methods such as Chi squared test, Information gain and Correlation coefficient scores, Wrapper methods such as Recursive feature elimination algorithm and Embedded methods such as regularization methods. In this paper, I propose feature selection from a model based on feature importance after a certain threshold parameter. Empirical evaluation using a real-life blog data set shows that this technique improves the classification accuracy of the current state of the art methods significantly.

1. Introduction

Blog refers to online personal writings which are generally written in informal language. There has been exponential growth of blogs written online for the past decade, which provides vital information for many commercial applications. For instance, it helps to figure out what kind of products male or female buy the most or what kind of topics most discussed by male or female. This classification gives companies a better strategy for advertising for specifically targeted gender and for improved product development.

Online blogs are generally small and most of the time written in informal language, which contains grammar errors, wrong spellings, special characters representing emoticons, abbreviations and some words and phrases from other languages. This is one of the main

reasons gender classification from text has become not an easy problem to solve.

Recent gender classification of blog authors focusses on features such as standard words, Part of speech tags, and regularization methods. This paper is going to discuss feature selection from a Stochastic Gradient Descent applied Support Vector Machines model. Where features which are considered unimportant are removed based on feature importance. Then again trained on new set of reduced features with improved accuracy. Experimental results based on a real-life blog data scrapped from many blog hosting sites show that this technique significantly improved accuracy when applied to supervised learning algorithms and also to a neural network.

2. Classifier: Stochastic Gradient Descent on SVM

Optimization is at the heart of most machine learning algorithms. There are several algorithms to optimize convex functions such as Gradient Descent, Subgradient for non-differential convex functions and Projected Subgradient Descent etc. One such method is Stochastic Gradient Descent. In order to apply convex optimization, the objective function should be convex.

Let look at the convexity of SVM.

$$f(w) = C\sum_{i=1}^{n}\max\{0,1-y_i w^T x_i\} + \frac{1}{2}\|w\|^2$$

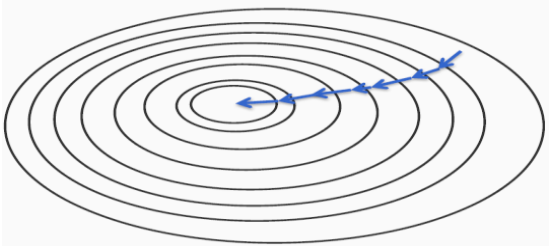
- First term is sum(max(linear)). Linear is convex and sum/max preserves convexity.
- In second term, Squared norm is convex and non-negative scaling preserves convexity.

Objective function:

$$\min_{w}\left\{C\sum_{i=1}^{n}\max\{0,1-y_i w^T x_i\} + \frac{1}{2}\|w\|^2\right\}$$

Hence, the objective function of SVM is convex function that can be optimized.

Gradient Descent on SVM objective requires summing over the entire training set. This causes optimization to slow and does not really scale.



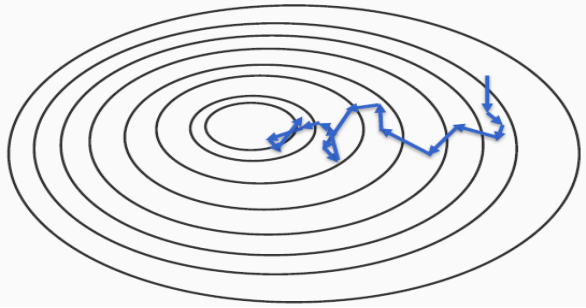
Stochastic Gradient Descent is a stochastic approximation of gradient descent optimization where samples are selected randomly instead of as a single group.

Algorithm:

Given a training set $S = \{(x_i, y_i)\}, x \in \mathbb{R}^n, y \in \{-1,1\}$

1. *Initialize* $w^0 = 0 \in \mathbb{R}^n$
2. *For epoch=1…*
   1. *Pick a random example* (x_i, y_i) *from training set S*
   2. *Treat* (x_i, y_i) *as a full data set and take the derivative of the SVM objective at the current* w^{t-1} *to be* $\nabla f^t(w^{t-1})$
   3. *Update* $w^t \leftarrow w^{t-1} - \gamma_t \nabla f^t(w^{t-1})$
3. *Return final w*

This algorithm is guaranteed to converge to the minimum of f if γ_t is small enough since objective $f(w)$ is a convex function.



SGD does many more updates than gradient descent, but each individual update is computationally less expensive.

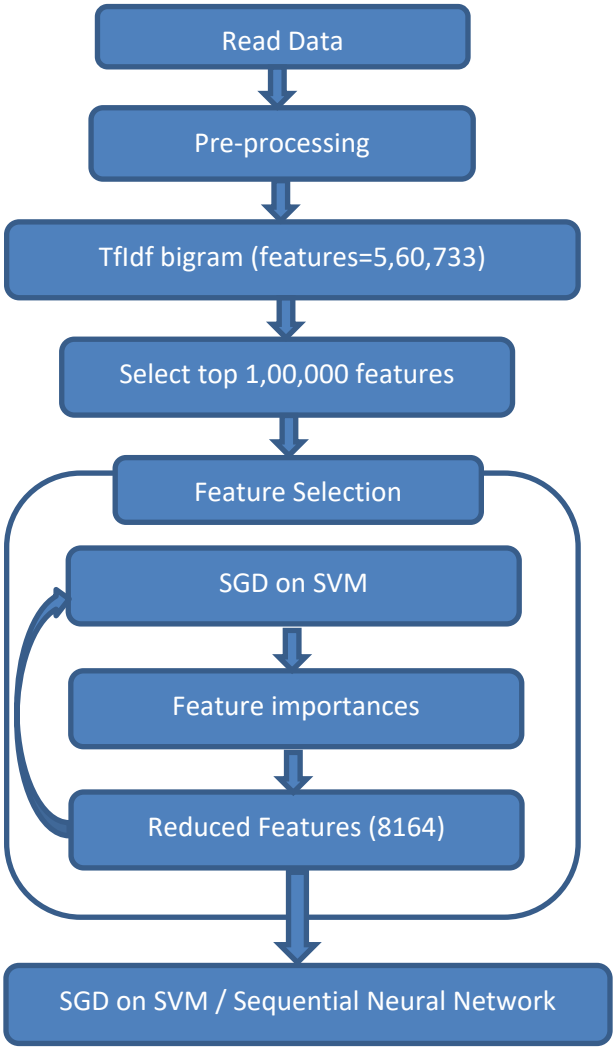
3. Feature Selection from Model:

Feature selection is usually used as a pre-processing step before doing the actual learning. Feature selection from model is a meta transformer that can be used along with any estimator to reduce the dimensionality of the data that has $coef\_$ or $feature\_importances\_$ attribute after fitting. The features are considered unimportant and removed, if the corresponding coefficient values are below the provide threshold parameter. There are several heuristics for finding a threshold such as *mean, median* etc.

We make use of a classifier to evaluate feature importances and select the most relevant features. Then we train on the transformed output, i.e. using only the relevant features. In this paper we will select the relevant features from the model SGD on SVM and use these reduced features to train on supervised algorithms and deep learning models to achieve the improved accuracy.

4. Implementation

The implementation of the model flows as shown below.



Real life blog data scrapped from many blog hosting sites consists of 3226 samples with target classes 'M' or 'F'.

Pre-processing: Pre-processing step is most crucial for any machine learning model to achieve better accuracy. In this process the document set consists of 6 samples with empty data in between which need to be dropped. Some of the target labels are lower cases like 'f' or 'm' and some are padded with spaces like ' f ' or 'M ' etc. Which are needed to take care while converting them into numerical values. Pre-processing step also includes the following.

1. Removed all stop words
2. Removed all special characters except the following.
    a. !!! – most used by Women
    b. ?? – most used by Women
    c. ??? – most used by Women
    d. :) – most used by Women
    e. :(– most used by Men
3. Removed patterns like.
   Numbers, 366x768, 10.02, 100th, 100lbs, 00am, 00pm, 100kg, etc.

Vectorizing: Baseline for the process is standard word unigram and bigram features. Calculated term frequency indefinite document frequency for each unigram and bigram from text corpus. Based on TFIDF selected the top 1,00,000 most important features which go to feature selection phase.

Feature Selection phase: Top 1,00,000 features are trained on SGD on SVM classifier. The features are considered unimportant and removed, if the corresponding coefficient values are below the provide threshold parameter. There are several heuristics for finding a threshold such as *mean, median* etc default being mean. After the first iteration features are reduced to 23442, which are again trained on same classifier. After the second iteration the features are reduced to 8164 which are final features for the model.

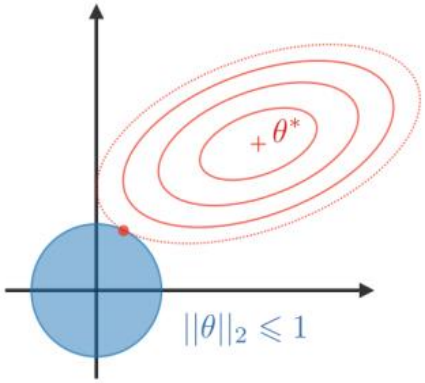
Final Classifier:

After the relevant feature are selected, trained the features on two classifiers.

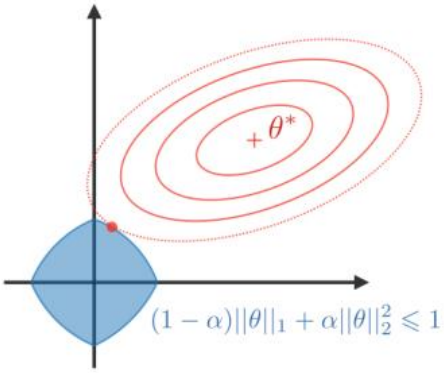
1. SGD-SVM
2. Sequential Neural Network model

1. SGD-SVM: Input data is split into Train and Test set at 70:30 ratio. Reduced 8164 features are trained with Grid Search 10-fold Cross validation for hyper parameter tuning on SGD-SVM classifier. Below are the regularization parameters along with $alpha$ that are tuned for the model.

L_2 norm: It is sum of the square of the weights. It makes coefficients smaller.



Elastic Net: This method linearly combines Lasso and Ridge regression.

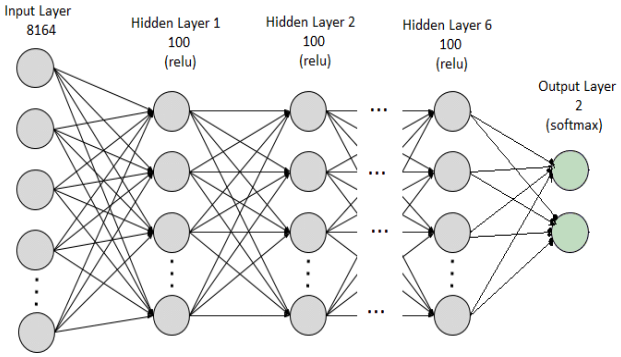


Elastic Net is a trade-off between variable selection and small coefficients.

It turned out that Elastic Net best suited for feature selection and l_2 regularization for the SGD-SVM final model, and best alpha value is 0.0001 among 0.0001, 0.001, 0.01, 0.1 and 1.0.

2. Sequential Neural Network: Neural networks account for interactions really well. Deep learning uses especially powerful neural networks.

Using the reduced features (8167) from the feature selection phase, built a sequential deep neural network of 6 hidden layers each with 100 nodes and an output layer with 2 nodes which uses 'softmax' activation.



Each hidden layer uses ReLU (Rectified Linear Activation) where it is defined as,

$$ReLU(x) = \begin{cases} 0 \; if \; x < 0 \\ x \; if \; x \geq 0 \end{cases}$$

The sequential model is compiled on 'adam' optimizer which is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications. Adam computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. The model is trained and tested on 70:30 ratio of data set for 10 epochs.

5. Experimental Results

This section evaluates the proposed techniques and sees how they affect the classification accuracy. And compare with the existing state-of-art algorithms and systems. Previous methods on gender classification achieved accuracies as shown in below table.

Settings	NB	SVM	SVM_R
All features	63.01	68.84	70.03
All features, no POS patters	60.73	65.17	66.17
POS 1, 2, 3-grams + EFS	71.24	82.71	83.86
POS patterns + EFS	73.57	86.24	88.56

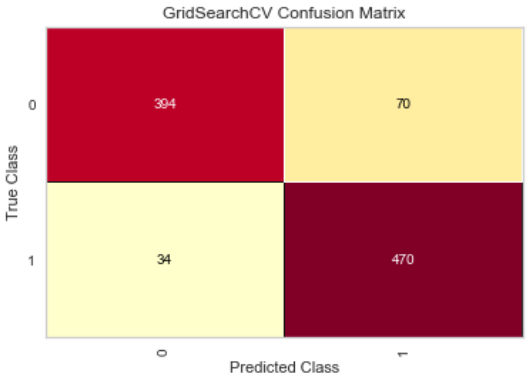
Table 1: Accuracies of SVM, SVM_R and NB with different feature selection methods

While no method used Stochastic Gradient Descent on SVM, proposed model in this paper has shown significant accuracy improvements. Results of the model are shown in below table.

Settings	SGD-SVM	Sequential Neural Network model
1, 2-gram features from model	89.25	91.9

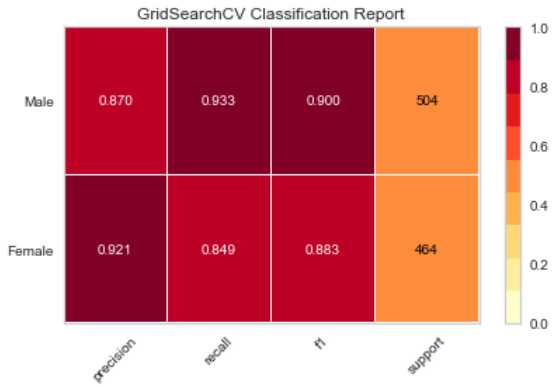
Table2: Accuracies of proposed method in this paper

Confusion Matrix: It reports how each of the test values predicted classes compare to their actual classes.

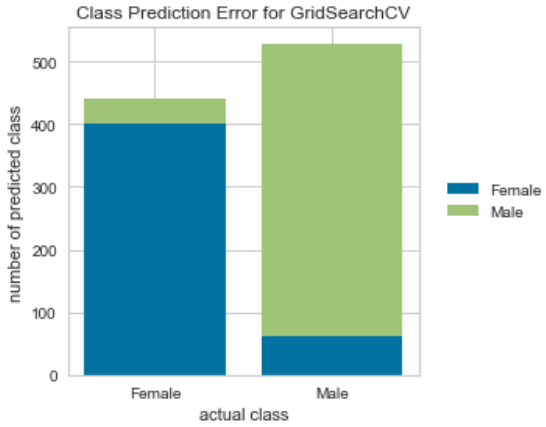


Here 0 represent Female and 1 represents Male.

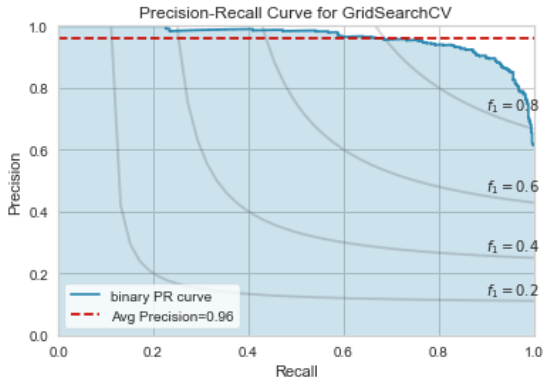
Classification Report: The classification report shows a representation of the main classification metrics on a per-class basis. This gives a deeper intuition of the classifier behaviour over global accuracy which can mask functional weaknesses in one class of a multiclass problem.



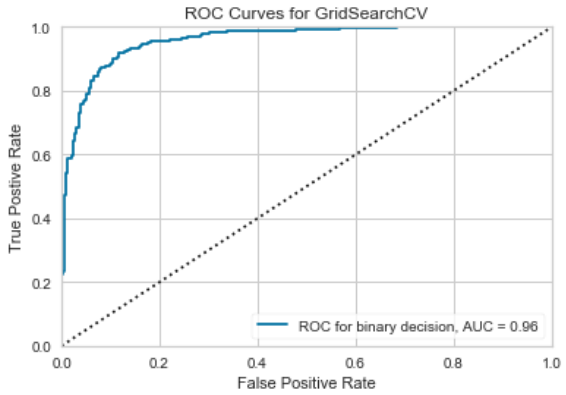
Class Prediction Error: The class prediction error chart provides a way to quickly understand how good your classifier is at predicting the right classes.



Precision Recall Curve: This metric used to evaluate a classifier's quality, particularly when classes are very imbalanced. The precision-recall curve shows the trade-off between precision, a measure of result relevancy, and recall, a measure of how many relevant results are returned.

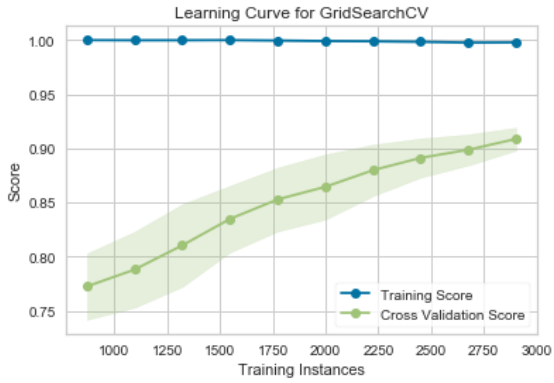


Area Under the Curve: AOC is a computation of the relationship between false positives and true positives. The higher the AUC, the better the model generally is. However, it is also important to inspect the "steepness" of the curve, as this describes the maximization of the true positive rate while minimizing the false positive rate.

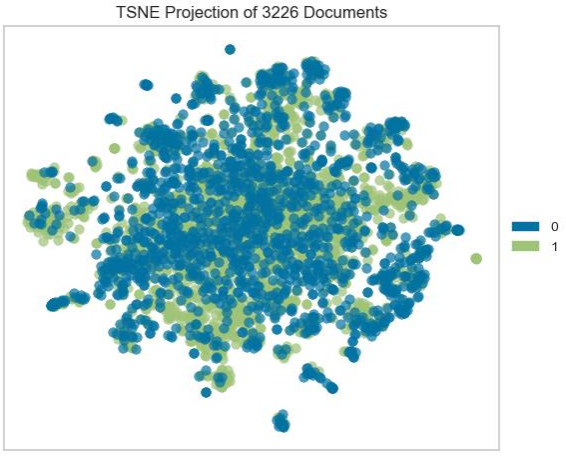


Learning Curve: A learning curve shows the relationship of the training score vs the cross validated test score for an estimator with a varying number of training samples. This visualization is typically used two show two things:

1. How much the estimator benefits from more data (e.g. do we have "enough data" or will the estimator get better if used in an online fashion).
2. If the estimator is more sensitive to error due to variance vs. error due to bias.



t-SNE Corpus Visualization: One very popular method for visualizing document similarity is to use t-distributed stochastic neighbour embedding, by decomposing high-dimensional document vectors into 2 dimensions using probability distributions from both the original dimensionality and the decomposed dimensionality.



6. Resources

The execution time for the proposed method for the data set of 3226 documents is 6.40 minutes

- Intel I5 7th gen processor.
- 12GB RAM
- Size of the project: 752 MB
- Dependencies
    - Python 3.6.5
    - Scikit-learn 0.20.2
    - Keras 2.2.4
    - Numpy 1.16
    - Pandas 0.24.1
    - Tensorflow 1.13.1
    - Yellowbrick 0.9.1
    - Pip 19.03
    - Matplotlib 3.0.3

7. Conclusion

This paper studied the problem of gender classification. Although there have been several existing papers studying the problem, the current accuracy is still far from ideal. In this paper I have proposed, Stochastic Gradient Descent on SVM for feature selection from the model. Iteratively selecting features from a trained model will give most import features for the problem. And implemented sequential neural network model using reduced features. Experimental results based on real-life blog data set gave effectiveness of the proposed method. This method significantly produced higher accuracy than the current state-of-the techniques and systems.

References

https://www.cs.uic.edu/~liub/publications/EMNLP-2010-blog-gender.pdf

https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

https://scikit-learn.org/stable/modules/feature_selection.html

https://www.datacamp.com/community/tutorials/feature-selection-python

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

https://scikit-learn.org/stable/modules/grid_search.html

https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

https://www.scikit-yb.org/en/latest/api/classifier/confusion_matrix.html

https://www.scikit-yb.org/en/latest/api/classifier/rocauc.html

https://www.scikit-yb.org/en/latest/api/classifier/prcurve.html

https://www.scikit-yb.org/en/latest/api/classifier/class_prediction_error.html

https://www.scikit-yb.org/en/latest/api/model_selection/learning_curve.html

https://www.scikit-yb.org/en/latest/api/text/tsne.html

https://keras.io/getting-started/sequential-model-guide/