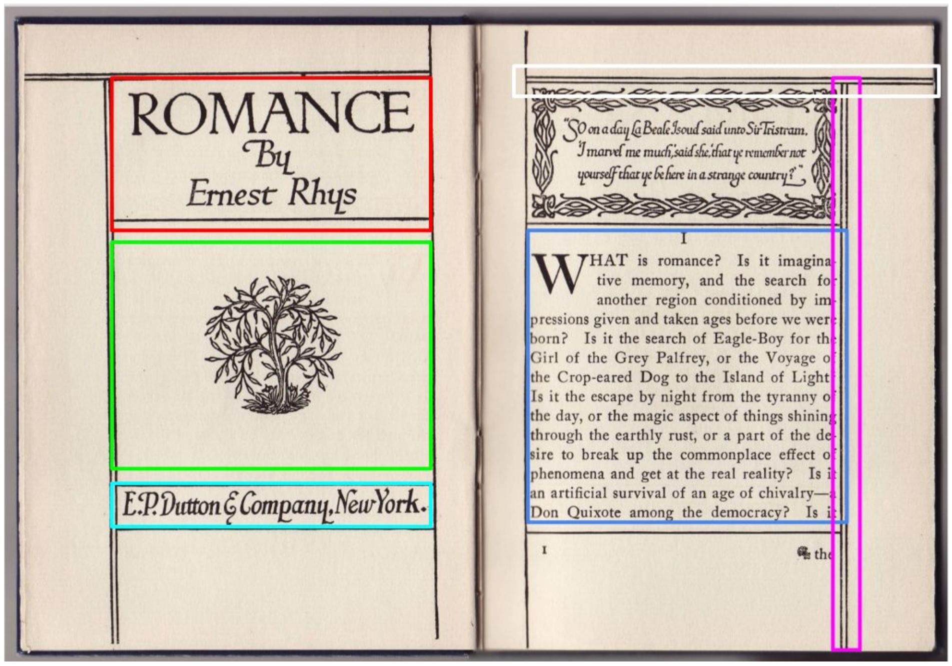
Page Blocks Classification

Abisado, Valaquio, Vargas

INTRODUCTION

Books has come a long way. It is a tool for us humans to record our knowledge and preserve it for future use. It consists of pages that is glued or sewn together along one side.



ROMANCE
By
Ernest Rhys

E.P. Dutton & Company, New York.

"So on a day La Beale Isoud said unto Sir Tristram, 'I marvel me much,' said she, 'that ye remember not yourself that ye be here in a strange country?'"

I

WHAT is romance? Is it imagina
tive memory, and the search fo
another region conditioned by im
pressions given and taken ages before we wer
born? Is it the search of Eagle-Boy for th
Girl of the Grey Palfrey, or the Voyage o
the Crop-eared Dog to the Island of Light
Is it the escape by night from the tyranny o
the day, or the magic aspect of things shinin
through the earthly rust, or a part of the de
sire to break up the commonplace effect o
phenomena and get at the real reality? Is i
an artificial survival of an age of chivalry—
Don Quixote among the democracy? Is i

I th

(Fig 1: Some page blocks in the pages are highlighted with different colored boxes)

Each page in a book has its own content and it may vary from text, graphics, picture or other auxiliary elements within a page like horizontal and vertical lines. Our goal in this machine problem is to classify these elements or what we call the page blocks.

The problem consists in classifying all the blocks of the page layout of a document that has been detected by a segmentation process. This is an essential step in document analysis in order to separate text from graphic areas. Indeed, the five classes are: text (1), horizontal line (2), picture (3), vertical line (4) and graphic (5).

This is important because it could be used to automatically extract texts and pictures within the pages of the book because we already know the general classification of that block. The people who will benefit from this are the people who wants to create applications that digitizes books. Also, it would benefit the blind or disabled who wants to have an automatic reader with natural language processing.

DATASET

The name of our dataset is “Page Blocks Classification Data Set” and that it is from this link: https://archive.ics.uci.edu/ml/datasets/Page+Blocks+Classification . We divided our dataset into training, validation and test set with 70%, 10% and 20% of the whole dataset respectively.

There are 5473 examples (rows) that comes from 54 distinct documents. Each observation concerns one block. All attributes are numeric. There are 11 column and the following table shows the summary of each attribute (column):

Attribute Name	Data Type	Description
height	integer	Height of the block.
length	integer	Length of the block.
area	integer	Area of the block (height * length);
eccen	continuous	Eccentricity of the block (length / height);
p_black	continuous	Percentage of black pixels within the block (blackpix / area);
p_and	continuous	Percentage of black pixels after the application of the Run Length Smoothing Algorithm (RLSA);
mean_tr	continuous	Mean number of white-black transitions (blackpix / wb_trans);
blackpix	integer	Total number of black pixels in the original bitmap of the block.
blackand	integer	Total number of black pixels in the bitmap of the block after the RLSA.
wb_transwb_trans	integer	Number of white-black transitions in the original bitmap of the block.
page_block_type	integer	Type of page block: text (1), horizontal line (2), picture (3), vertical line (4) and graphic (5)

This is the raw content of the first 10 lines of page-blocks.data:

```
 5   7    35  1.400  .400  .657    2.33    14     23   6    1
 6   7    42  1.167  .429  .881    3.60    18     37   5    1
 6  18   108  3.000  .287  .741    4.43    31     80   7    1
 5   7    35  1.400  .371  .743    4.33    13     26   3    1
 6   3    18   .500  .500  .944    2.25     9     17   4    1
```

5	8	40	1.600	.550	1.00	2.44	22	40	9	1
6	4	24	.667	.417	.708	2.50	10	17	4	1
5	6	30	1.200	.333	.333	10.00	10	10	1	1
5	5	25	1.000	.400	.520	10.00	10	13	1	1
5	7	35	1.400	.486	.914	8.50	17	32	2	1

CLASSIFIERS

In this machine problem, we used classifiers that is already implemented in the *scikit-learn* python library. We used the following classifiers:

- **KNeighborsClassifier**
  - **Classifier implementing the k-nearest neighbors vote.**

  K nearest neighbors is a simple algorithm that classifies new cases based on how similar it is to its nearest K neighbors using some kind of metric like a distance function. This algorithm is used for statistical estimation as well as pattern recognition in early 1970's.

  Distance functions

  Euclidean $$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

  Manhattan $$\sum_{i=1}^{k}|x_i - y_i|$$

  Minkowski $$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$$

  A case or an instance is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If K = 1, then the case is simply assigned to the class of its nearest neighbor.

- **Random Forest**

- **A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.**

Random Forest is a supervised learning algorithm. It creates a forest or ensemble of Decision Trees, most of the time trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result. In other words, Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

One big advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. It is less susceptible to overfitting compared to a single decision tree.

- **Multi-layer Perceptron classifier (Neural Network)**
  - **This model optimizes the log-loss function using LBFGS or stochastic gradient descent.**

A multilayer perceptron (MLP) is a kind of deep neural network. It has more than one perceptron. They are composed of an input layer to receive the signal, an output layer that makes a decision or prediction about the input, and in between those two, an arbitrary number of hidden layers that are the true computational engine of the MLP. MLPs with one hidden layer are capable of approximating any continuous function.

- **Voting Classifier (Ensemble)**
  - **Soft Voting/Majority Rule classifier for unfitted estimators.**

A voting classifier is a kind of ensemble classifier that is composed of multiple number of different classifiers. Each classifier within this ensemble can vote for their own classification and the voting classifier then considers all the votes. The class with the most votes or metric of votes will be the output of the voting classifier. Each vote from a classifier can we weighted depending on their accuracy or importance. The advantage of a voting classifier is that it combats overfitting by relying to different classifiers. It can sometimes generalize better than some classifiers.

EXPERIMENT RESULTS

1. KNeighborsClassifier

n_neigbors	**Validation accuracy (%) (weights=’uniform’**	**Validation accuracy (%) (weights=’distance’)**

	)	
n_neighbors=1	94.8811	94.8811
n_neighbors=2	94.5155	94.8811
n_neighbors=3	94.3327	95.0639
n_neighbors=4	94.1499	**95.2468**
n_neighbors=5	94.1499	94.8811
n_neighbors=10	93.4186	93.7842
n_neighbors=50	89.0311	92.8702
n_neighbors=100	88.2998	90.6764
n_neighbors=200	87.3857	89.3967
n_neighbors=500	87.3857	88.6654
n_neighbors=1000	87.3857	88.6654

In this case, it just so happens that K = 4 has the right mix of generalization. It does not overfit as with K = 1 as well as it not being affected by a lot of neighbors as noise.

2. Random Forest

max_depth	**n_estimators**	**Validation accuracy (%)**
None (nodes are expanded until all leaves are pure)	10	96.7093
None (nodes are expanded until all leaves are pure)	20	97.0749
None (nodes are expanded until all leaves are pure)	**30**	**97.6234**

1	10	86.6544
1	20	89.0310
1	30	87.9341
5	10	95.7952
5	20	96.3436
5	30	96.5265
10	10	96.7093
10	20	97.2577
10	30	97.2577

After further tweaking, the best configuration that was found was a Random Forest with **max_depth = None and n_estimators = 81 with 98.1718**.

This is the best configuration because the there is no max_depth so it will create nodes until the last node within a branch. It exhausts all the nodes and thus it will give better results usually but the drawback is that it takes more time and processing. Also, we saw a pattern where the higher the number of estimators the better. This is maybe because the more the estimator, there is less chance of overfitting.

3. Multi-layer Perceptron classifier (Neural Network)

solver : The solver for weight optimization.

- 'lbfgs' is an optimizer in the family of quasi-Newton methods.
- 'sgd' refers to stochastic gradient descent.
- 'adam' refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba

activation : Activation function for the hidden layer.

- 'identity', no-op activation, useful to implement linear bottleneck, returns f(x) = x
- 'logistic', the logistic sigmoid function, returns f(x) = 1 / (1 + exp(-x)).

- ‘tanh’, the hyperbolic tan function, returns f(x) = tanh(x).
- ‘relu’, the rectified linear unit function, returns f(x) = max(0, x)

solver	**activation**	**Validation accuracy (%)**
lbfgs	identity	87.0201
lbfgs	**logistic**	**97.4406**
lbfgs	tanh	96.8921
lbfgs	relu	86.8372
sgd	identity	87.3857
sgd	logistic	88.4826
sgd	tanh	94.1499
sgd	relu	91.0420
adam	identity	92.6873
adam	logistic	95.9780
adam	tanh	96.3436
adam	relu	95.7952

All of these experiments have hidden_layer_sizes=(600,). In other words it has only one hidden layer with 600 nodes. We found that having one hidden layer with 600 nodes gives the highest accuracy. This is maybe just by chance for our data set. We also saw that the higher the number of nodes the better but it only works with one layer. Thus, for this dataset, it is prefered that the neural network is wider and not deeper.

The prefered solver is lbfgs because it works well with smaller datasets as what is mentioned in scikit-learn’s website:

> *“For small datasets, however, ‘lbfgs’ can converge faster and perform better.”*

Lastly, logistic (sigmoid) activation function performed best maybe because it is the most natural activation function. It only outputs in a range of

[0,1] and thus advantages on different situations. It happens that because of this nature, it performs better.

Ensemble - Voting Classifier

flatten_transform :

Affects shape of transform output only when voting='soft' If voting='soft' and flatten_transform=True, transform method returns matrix with shape (n_samples, n_classifiers * n_classes). If flatten_transform=False, it returns (n_classifiers, n_samples, n_classes).

voting : str, {'hard', 'soft'}

If 'hard', uses predicted class labels for majority rule voting. Else if 'soft', predicts the class label based on the argmax of the sums of the predicted probabilities, which is recommended for an ensemble of well-calibrated classifiers.

parameters	Validation accuracy (%)
voting='hard'	**97.8062**
voting='soft', flatten_transform=True	97.2578
voting='soft', flatten_transform=False	97.2578

The difference of of the accuracy is not that significant and we can't tell if what works better overall.

Summary of Validation Accuracy Results (Best of each classifier)

Classifier	Validation accuracy (%)
KNeighborsClassifier	**95.2468**
Random Forest	**98.1718**
Multi-layer Perceptron classifier (Neural Network)	**97.4406**
Voting Classifier	**97.8062**

Summary of Test Set Accuracy Results (Best of each classifier)

Classifier	Test accuracy (%)
KNeighborsClassifier	**95.98173515981735**
Random Forest	**97.35159817351598**
Multi-layer Perceptron classifier (Neural Network)	**97.62557077625571**
Voting Classifier	**97.35159817351598**

Overall, our models performed very well even for the test set.

CONCLUSION

Some classifiers did well than others in the experiment but the best classifiers for each of the different kinds of classifiers used here have almost the same accuracy. This is maybe because for this kind of dataset, this is the maximum accuracy you can get or it converges to some maximum point near the average test accuracy of our results.

The worst classifier is the KNeighborsClassifier because it is susceptible to noise and only considers what is near to it. Sometimes that is not enough. The main advantage of the KNeighborsClassifier is that it doesn't need training and there is little to no overhead before classification compared to the Random forest that takes awhile to create an ensemble of decision trees as well as the MLP Neural Network that takes a very long time training.

The best classifiers are the last three (Random forest, MLP Neural Network and the Voting Classifier) but they have one thing in common, they require a lot of processing. The last one, the voting classifier just piggy backs from the success of the last two and thus it has even the same test accuracy with the Random forest.

REFERENCES

Esposito F., Malerba D., & Semeraro G Multistrategy Learning for Document Recognition Applied Artificial Intelligence, 8, pp. 33-84, 1994

http://www.saedsayad.com/k_nearest_neighbors.htm

http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html#sklearn.neural_network.MLPClassifier https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd

https://deeplearning4j.org/multilayerperceptron

http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html