GNG 5125: Assignment1 Report

Jianxiao Luo, Shuyu Liu, Zhaoyang Xie, Souroosh Memarian

February 2020

Text Classification

1 Overview

Our group took seven different samples of Gutenberg digital books, that are of seven different authors. We did several steps of data preparing and preprocessing to get it ready for our models. Three different transformations are used to transform the text to digital, followed by five machine learning algorithms to do the training and testing respectively. To evaluate each model and compare there results, we did ten-fold cross-validation to reach a high reliability and we also do some visualizations to help us learn the results intuitively. In the end, we did some error analysis to find where the program would make mistake. Our report will explain each part in detail.

2 Preprocessing

Data pre-processing step is very essential for data classification, which also takes time and effort to do data cleaning and choose suitable data structure. For this assignment, our team did a couple of data cleaning strategies and choose python dictionary as our data structure to manipulate raw text data.

2.1 Data Selection and Preparing

First of all,We made a list of 13 books from Gutenberg library,We labeled them from 0 to 12.Seven books out of 13 books should be selected.For each book, our purpose is to extract 200 documents which each of them includes 150 words and we don't want to choose them from the first 10 percent of each book(we eliminate the first 10 percent of each book because they mostly contain unrelated materials like preface and publisher notes). Therefore, the most important criterion for selecting these books is that each of them must have at least 30000/0.9 words.After choosing books, by using word tokenizing, we create the list of words of each book.Next,by dividing the last 90 percent of the sequence into 150-word parts,The documents will be obtained for each document. After that, we attached each document with its author. All of two attributes are saved in a dictionary data type. As displayed on figure 1, the data information after data-processing is implemented in a dictionary. In the dictionary, there are 1400 key-value pairs. The key elements are 150 words (aka: document) selected from seven books. The document itself is displayed by sequence in each key element. The value element is author name which the document belonging to. The final step before data transformation is to randomly shuffle the dictionary elements. The purpose of shuffling data is that ensuring that each data point creates an ”independent” change on the model, without being biased by the same points before them. Suppose data is sorted in a specified order.

2.2 Data Cleaning

Data cleaning is a significant part of data preprocessing which includes deleting irrelevant and inaccurate parts of data.Two functions RegexpTokenizer()(for removing punctuation or numbers) and lower()(for changing all the words into lowercase) was used for data cleaning purposes.

3 Feature Engineering

3.1 Word of Bag

In order to train the text file, we need to extract some useful information from the raw text. This step is also called feature engineering. A common and simple of feature extraction method is called the bag-of-words (BOW) method. The representation of the BOW includes two parts: vocabulary of known words and frequency

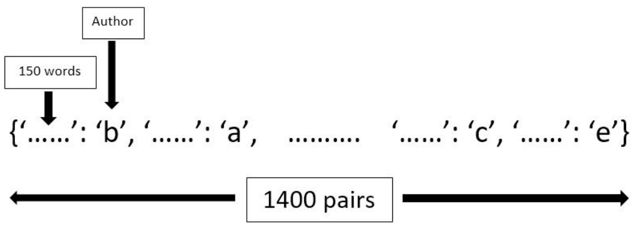
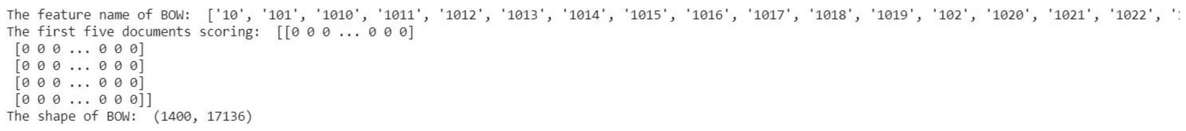


Figure 1: Data Structure after pre-processing

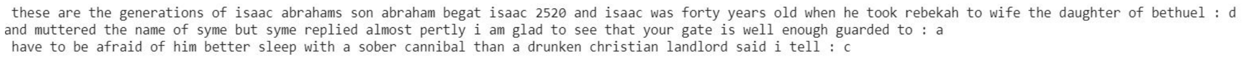
of the known words. The model only concerns about whether known words appears in a document while ignore the sequence of the words. To better illustrate the BOW, in the following we will explain what we did in this assignment.

In the previous section, we splitted seven books into 150-words documents and then saved with the author name into a single dictionary. The dictionary includes 1400 elements. In the feature engineering section, we will separate the dictionary into training data and true data, where training data is the 150-words documents and true data is the name of authors. (See the figure 2) In the next step, the training data is converted into a list of vectors that we can use as input for machine learning models. (See the figure 3) On the same figure, feature name and the first five document scoring of the BOW are displayed as well.



```
The feature name of BOW:  ['10', '101', '1010', '1011', '1012', '1013', '1014', '1015', '1016', '1017', '1018', '1019', '102', '1020', '1021', '1022', ':
The first five documents scoring:  [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
The shape of BOW:  (1400, 17136)
```

Figure 2: Important info of Bag of Words



```
 these are the generations of isaac abrahams son abraham begat isaac 2520 and isaac was forty years old when he took rebekah to wife the daughter of bethuel : d
and muttered the name of syme but syme replied almost pertly i am glad to see that your gate is well enough guarded to : a
 have to be afraid of him better sleep with a sober cannibal than a drunken christian landlord said i tell : c
```

Figure 3: Input dictionary

3.2 Two-Gram

N-gram is a contiguous sequence of n items from a given sample. The items can be phonemes, syllables, letters, words or base pairs according to the application. In our application, we use words to be the items and set n as two. In fact, BOW is a special example of N-gram, which sets n as one. Two benefits of n-gram models (and algorithms that use them) are simplicity and scalability – with larger n, a model can store more context with a well-understood space–time tradeoff, enabling small experiments to scale up efficiently.[3] For example, this sentence, ["I love data science application"], can be transfer to two-gram as ["I love", "love data", "data science", "science application"]. Unlike BOW, it can maintain some sequence and collocations of the author's writing.
After we get all the 2-gram collocations, we count the time of them and store in a list. In principle, every two words can make up a 2-gram collocation, but many collocations do not exist. We use sklearn to help us do two-gram. In our text, we get over 10,000 words in BOW, and for 2-gram, we have over 130,000 collocations, which is not too bad, although it also costs much more time to train.

3.3 TF-IDF

It's a little unfair to regard TF-IDF as a kind of transform. Unlike bow or n-gram which focus on transform words into counts, TF-IDF not just focus on the occurrences of tokens in each document, it also normalizes and weights with diminishing importance tokens that occur in the majority of samples / documents.
We use three transform methods here, they are "bow", "tf-idf" and "n-gram", but I think "tf-idf" could be used in both "bow" and "n-gram" to get a normalization values of their vectors. Actually, "tf-idf" is a kind of improvement.
To clarify the mechanism of this algorithm, we need to separate "tf-idf" into two parts - "tf" and "idf", each

```
Length of vector:  1400
Length of vector[0]:  135067
Vector[0]:  [0 0 0 ... 0 0 0]
```

Figure 4: Important info of Two-gram

one of which contains their unique but relevant formulas and logic to handle with counts.
• In tf(term-frequency) parts: $df(t)$ is the number of documents in the document set that contain term t.
• In idf(inverse document-frequency) parts:

$$idf(t) = \log \frac{1+n}{1+df(t)} + 1$$

where n is the total number of documents in the document set
• The resulting tf-idf vectors are then normalized by the Euclidean norm:

$$V_{norm} = \frac{v}{||v||_2} = \frac{v}{\sqrt{v_1^2 + v_2^2 + ... + v_n^2}}$$

4 Algorithm Experiments and Results

4.1 K-Nearest Neighbor

One of popular machine learning algorithms is K Nearest Neighbor (KNN). KNN is very simply, easy to understand and versatile. The application of the KNN is very wide in finance, healthcare and image recognition. Some advantages of the KNN: Non-parametric and lazy learning algorithm. Non-parametric means the model structure determined from the dataset. Lazy algorithm means all training data used in the testing phase, which also save time and computing memory. In KNN, K is the number of nearest neighbors. Suppose point A is the new point needs to predict. Firstly, we need to find the k closest point near the point A. Then classify it by majority vote of its K neighbors. To illustrate the concept better, a visual aid is displayed in the figure 5. For this example, we only take the 3 nearest points in considerations. Therefore, the new point is in Class B. In this assignment, we used the default K (k=5) value for training. Here is another observation we found, every algorithm in the assignment except the regression algorithm can take words as the desired/true value. The reason behind the scene is KNN, SVM, MLP and Naive Bayes belongs to classification.

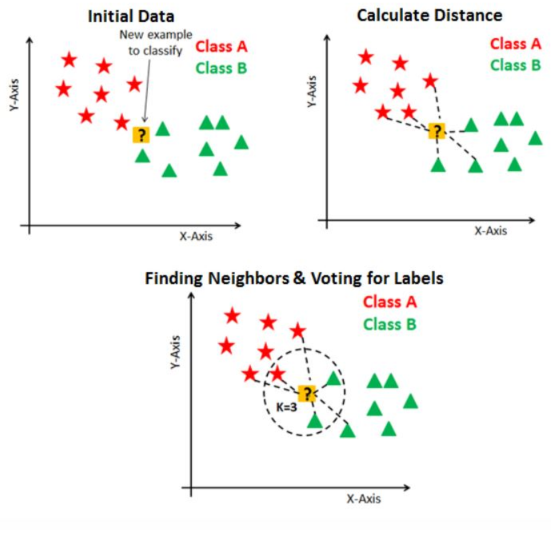


Figure 5: KNN algorithm

4.2 Decision Tree

Tree based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based methods empower predictive models with high accuracy, stability and ease of interpretation. Unlike

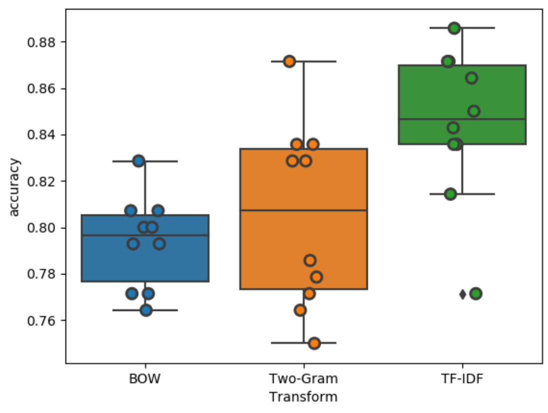


Figure 6: 10-Fold cross validation scoring for KNN

linear models, they map non-linear relationships quite well. [7] Some advantages and disadvantages of the decision tree: Advantages of decision tree:

- Easy to understand
- Useful in data exploration. One of the fastest way to identify most significant variables and relation between two or more variables.

Disadvantages of decision tree:

- May suffer from over-fitting
- Not fit for continuous variables
- Does not easily handle non-numeric data

For this assignment, the decision tree classifier takes two input variables: list of documents and authors' name. We kept all the parameters as default. The validation testing results are displayed in the figure 7

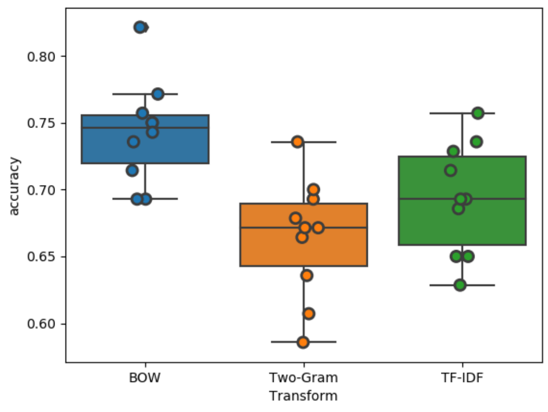


Figure 7: 10-Fold cross validation scoring for Decision Tree

4.3 Multilayer Perceptron

The advantages of Multi-layer Perceptron (MLP) are:

- Capability to learn non-linear models.
- Capability to learn models in real-time

The disadvantages of MLP includes

- MLP with hidden layers have a non-convex loss function where there exists more than one local minimum.
- MLP requires tuning a number of hyperparameters such as the number of hidden neurons, layers, and iterations.
- MLP is sensitive to feature scaling.

The architecture of the neurons in the network is often called the network topology. (See the figure 8)The bottom layer that takes input from data-set is called the visible layer, because it is the exposed part of the network. Layers after the input layer are called hidden layers because that are not directly exposed to the input. The final hidden layer is called the output layer and it is responsible for outputting a value or vector of values that correspond to the format required for the problem. Neural networks require the input to be scaled in a consistent way. The typical range is between 0 and 1, which is called normalization. Another popular technique is to standardize it so that the distribution of each column has the mean of zero and the standard deviation of 1.

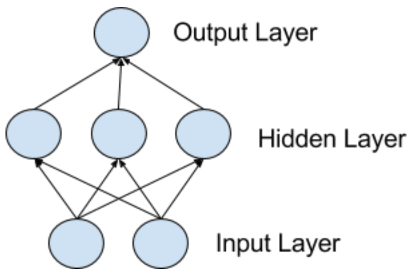


Figure 8: Multilayer Perceptron structure

We only successfully trained the Bag of Words and TF-IDF with the multilayer perceptron. We made a few attempts to train n-gram, however we ended up crashed the Colab during training process. We found out the number of input neurons for the 2-gram is more than 150,000, which might be the reason failed the training process. Compared the n-gram, the Bag of words transformation has relatively small input neurons (17,000 input neurons), which leads to a promising training results. There is another observation regarding of training MLP. We replaced alphabet characters with numeric character. In the other words, we mapped authors' name into 0-6 integer numbers. On the figure 9, the training accuracy shows fairly decent.

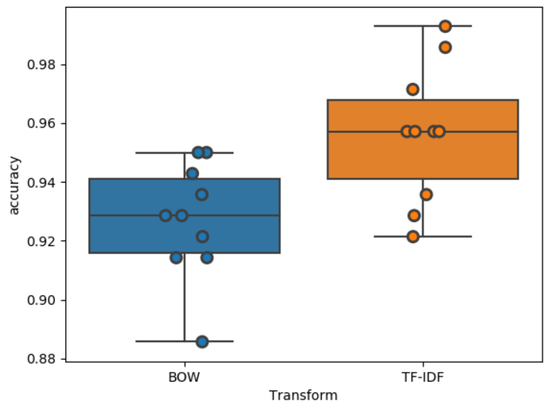


Figure 9: 10-Fold cross validation scoring for MLP

4.4 Regression

Regression is a method of modelling a target value based on independent predictors. This method is mostly used for forecasting and finding out cause and effect relationship between variables. Simple linear regression is a type of regression analysis where the number of independent variables is one and there is a linear relationship between the independent(x) and dependent(y) variable.[4]

In simplest linear regression, X and Y are both 1D. To apply this algorithm to our project, X is the vector of the transformed data and Y is the index of the author, so the X is n dimension(over 17,000 D in BOW and 135,000 in 2-Gram). It is called Multiple Linear Regression and it is hard to draw such a plot in this case.

The model is like:

$$Y = \beta_1 + \beta_2 X_2 + ... + \beta_n X_n + \varepsilon \tag{1}$$

The training of the model is to reduce the cost function and try to find a global minimize. The training process of it is somehow similar with MLP, like the key point of them is to modify the coefficients of the formula to get the least cost and regression can also use Gradient Descent to modify the parameters.

We get around 80% accuracy in all three transformations. Ten-fold cross validation result of regression see 10. When we looked at the result of regression, we found the outputs are continuous. We realized that we ignored the difference between classification and regression, and regression is not suitable for this classification task. Regression is to predictive a mapping from input variables to a continuous output variable, which is a real-value, such as an integer or floating point value. These are often quantities, such as amounts and sizes. It means that the labels should have numerical relationship with each others, like which is larger, higher, or later. In our task, the labels are seven different books, although we have transferred the word to true numbers, we cannot assign any numerical relationship to them, so it is meaningless to do comparison or calculation.

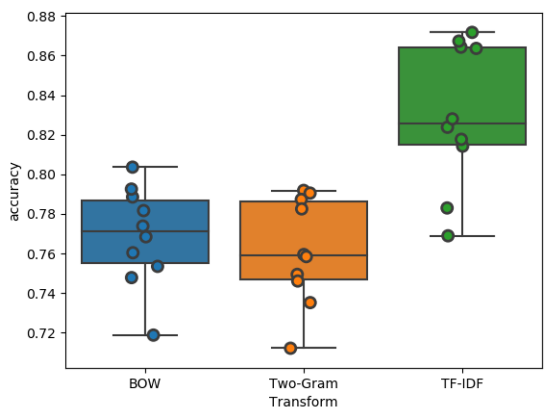


Figure 10: 10-Fold cross validation scoring for Linear Regression

4.5 SVM

A Support Vector Machine outputs an optimal hyperplane that classifies new examples based on given labelled training data. The SVM algorithm finds out a line/hyper-plane to distinguish different classes. There are several paramenters which we can tune to increase the accuracy, which includes: gamma, regularization parameter, margin, kernel and etc. For this assignment, we only tweaked the regularization parameter. In the theory, a low regularization parameter tolerates some outlier points while a high regularization achieve 0 tolerance with perfect partition. [6] Some observations are found during the training: we chose several different regularization values (1,3, 50) to test the algorithm. However, there is no major difference found among all the C values. Some pros and cons of this algorithm are listed as below.
The advantages of support vector machines are: [1]

- Effective in high dimensional spaces and cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different Kernel functions can be specified for the decision function.

The disadvantages of support vector machines include:

- If the number of features is much greater than the number of samples, avoid over-fitting in choosing Kernel functions and regularization term is crucial.
- SVM do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation

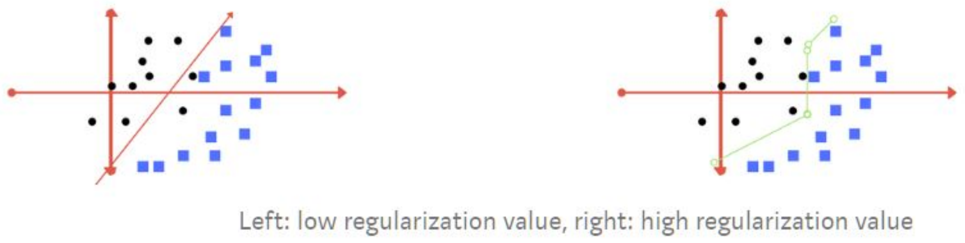

Left: low regularization value, right: high regularization value

Figure 11: SVM with different regularization

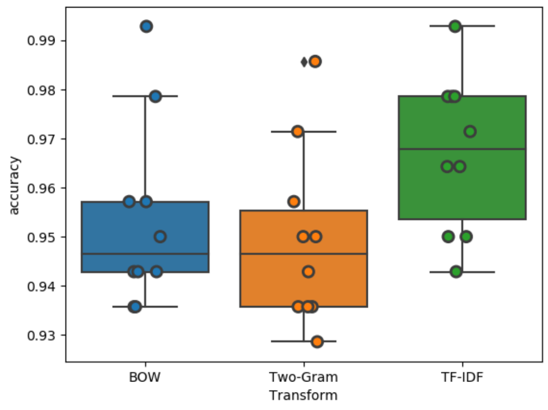


Figure 12: 10-Fold cross validation scoring for SVM

4.6 Gaussian Naive Bayes

Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models.[5] Actually, the model of naive Bayes is very simple, it is based on the Bayes' theorem.

The probabilistic model of naive Bayes:

$$p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(X)} \tag{2}$$

In practice, we always ignore the denominator because it does not depend on C and the values of the features x_i are given, so it is the same for any $p(C_k|x)$.

To estimate the parameters for a feature's distribution, one must assume a distribution or generate non-parametric models for the features from the training set. Gaussian naive Bayes is a typical assumption when dealing continuous values. Actually, we have tried 3 kinds of Bayes algorithm, including Guassian naive Bayes, Bernoulli naive Bayes and Multinomial naive Bayes. We get extremly high accuracy in Guassian naive Bayes, especially for two-gram transformation. The result of 10-fold cross validation see 13.

5 Algorithm Comparison

To compare these six algorithms clearly, we draw a line graph 14 to preform the accuracy. The order of algorithms in x axis is the order of complexity. The complexity order is based on the run time of two-gram for six algorithms 1.

The following part is a comparison from different perspectives:

- We select **Gaussian naive Bayes** as the champion algorithm, because it has the highest overall accuracy at between 95% to 99% and the shortest training time. All the three transforms preform very well by this algorithm. **SVM** should be the second good algorithm, with a good performance around 95%.
- The tendency of three transforms are the same and each of them reach the top in one or more algorithms. TF-IDF is the best transform overall, since it is the top in three algorithms, as we expected.

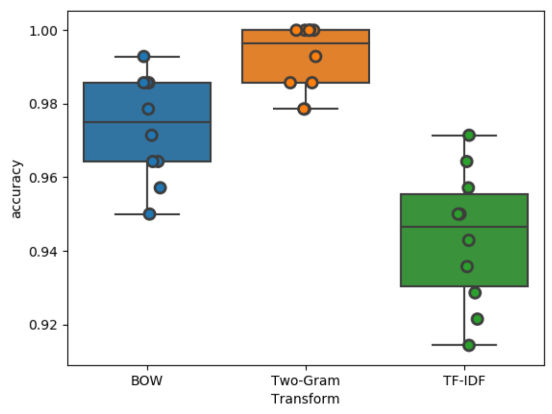


Figure 13: 10-Fold cross validation scoring for Gaussian naive Bayes

- There is also a obvious tendency that for the algorithm with overall high accuracy, the variance between three transforms is small, but for the algorithm with low accuracy, the variance is large.

Table 1: Run time of Two-Gram for six algorithms

Algorithm	Gaussian naive Bayes	Decision Tree	Linear Regression	KNN	SVM	MLP
time	42.4215	185.217	612.778	703.863	6697.08	NaN

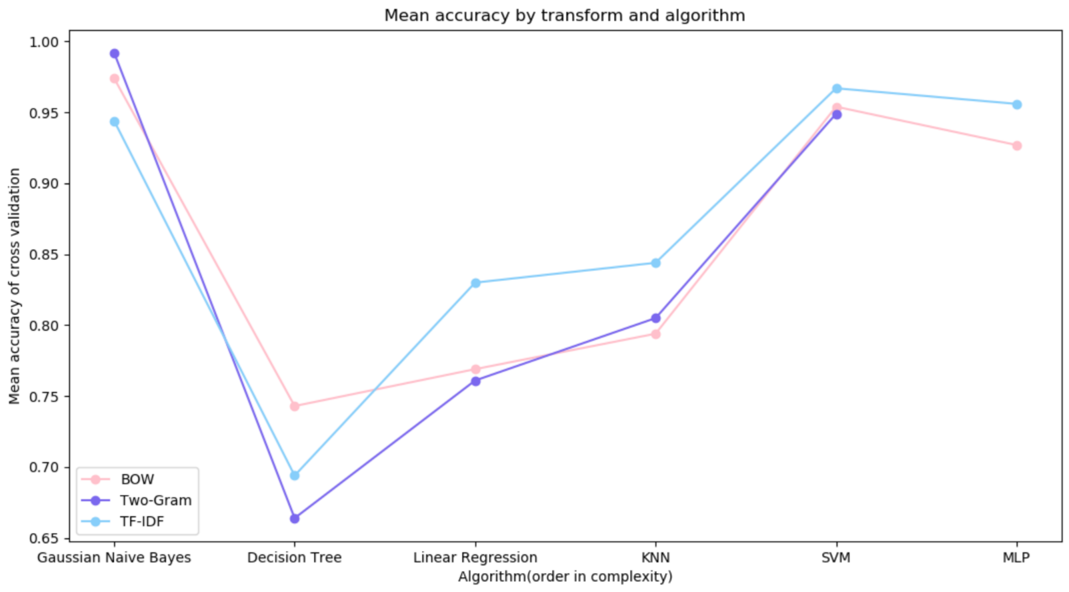


Figure 14: Mean accuracy by transform and algorithm

6 Error Analysis

Our modules is not always perfect, some of them produce a lower scores indicating that the prediction is not good. But when analysis error we decides to use the module(transform = 'bow', tf-idf = 'none', training = 'KNN'). The precision of this module is 80% roughly, which, of course is not a high values among all our results. But 80% is a "tolerable" value, meaning that this training module is not totally invalid. The first step is to generate the confusion matrix, which shows the counts of right prediction and wrong prediction according the results in a explicit way. This is confusion matrix(transform = 'bow', tf-idf = 'none', training = 'KNN') 15

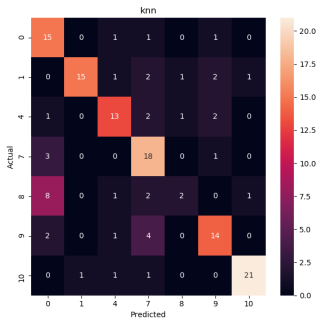
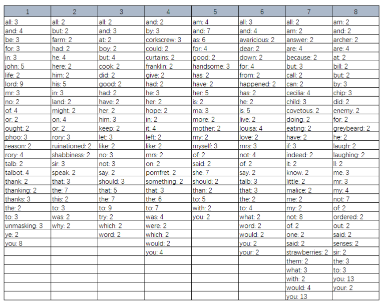


Figure 15: confusion matrix(transform = ’bow’, tf-idf = ’none’, training = ’KNN’)

We use nominal numbers to represent authors that we need to figure out, the y axis means the actual author in our test dataset and x axis means the predicted author our machine made. Numbers in the grid means counts under specified actual label and prediction label.
Here are two points that we need to focus on:
• grids alone the diagonal means the ”right predictions” while grids on other areas means ”no hits”
• The count of grid [Predicted=0, Actual=8] is 8, means 8 cases whose actual label is 8 are recognized as label 0.
The root cause of this kind of mistakes is the same, these cases are closer to label 0 than label 8. If we see words and counts in these cases, we will understand better:
16



1	2	3	4	5	6	7	8
all: 3	all: 2	all: 2	and: 2	am: 4	all: 3	all: 2	am: 2
and: 4	but: 2	and: 3	by: 3	and: 7	and: 4	am: 2	and: 2
be: 3	farm: 2	at: 2	corkscrew: 3	as: 6	avaricious: 2	answer: 2	archer: 2
for: 3	had: 2	boy: 2	could: 2	for: 4	dear: 2	are: 4	are: 4
in: 3	he: 4	but: 4	curtains: 2	good: 2	down: 2	because: 2	at: 2
john: 5	here: 2	cook: 2	franklin: 2	handsome: 3	for: 4	but: 3	bill: 2
life: 2	him: 2	did: 2	give: 2	has: 2	from: 2	call: 2	but: 2
lord: 9	his: 5	good: 2	had: 2	have: 2	happened: 2	can: 2	by: 3
mr: 3	in: 3	had: 2	he: 3	her: 5	has: 2	cecilia: 4	chip: 3
no: 2	land: 2	have: 2	her: 2	is: 2	he: 2	child: 3	did: 2
of: 4	might: 2	her: 2	hope: 2	ma: 3	is: 5	covetous: 2	enemy: 2
or: 2	on: 4	him: 3	in: 2	more: 2	live: 2	doing: 2	for: 2
ought: 2	or: 2	keep: 2	it: 4	mother: 2	louisa: 4	eating: 2	greybeard: 2
phoo: 3	rory: 3	let: 3	left: 2	my: 2	love: 2	have: 2	he: 2
reason: 2	ruinationed: 2	like: 2	like: 2	myself: 3	mrs: 3	if: 3	laugh: 2
rory: 4	shabbiness: 2	no: 3	mrs: 2	of: 2	not: 4	indeed: 2	laughing: 2
talb: 2	sir: 3	not: 3	on: 2	said: 2	of: 2	it: 2	ll: 2
talbot: 4	speak: 2	say: 2	pomfret: 2	she: 7	say: 2	know: 2	me: 3
thank: 2	that: 3	should: 3	something: 2	should: 2	talb: 3	little: 2	mr: 3
thanking: 2	the: 7	that: 5	that: 3	than: 2	that: 3	malice: 2	my: 4
thanks: 3	this: 2	the: 7	the: 6	to: 5	the: 2	me: 2	not: 7
the: 2	to: 3	to: 9	to: 7	with: 2	to: 4	my: 2	of: 2
to: 3	was: 2	try: 2	was: 4	you: 2	what: 2	not: 8	ordered: 2
unmasking: 3	why: 2	which: 2	were: 2		word: 2	of: 2	out: 2
ye: 2		word: 2	which: 2		would: 2	one: 2	said: 2
you: 8			would: 2		you: 2	said: 2	senses: 2
			you: 4		your: 2	strawberries: 2	sir: 2
						them: 2	the: 3
						what: 3	to: 3
						with: 2	you: 13
						would: 4	your: 2
						you: 13	

Figure 16: words(transform = ’bow’, tf-idf = ’none’, training = ’KNN’)

In order to improve the result, in fact we could use tf-idf, from the confusion-matrix, the result is better.

References

[1] scikit-learn

[2] NLTK

[3] Wikipedia N-Gram

[4] Introduction to Machine Learning Algorithms: Linear Regression

[5] Wikipedia Naive Bayes

[6] Support Vector Machine — Theory

[7] Decision tree