Extended Essay May 2021

Comparison of NB and k-NN using time and accuracy measurements

To what extent are Naïve Bayes classifier and k-NN algorithms efficient in text classification?

Computer Science

Word Count: 3781

Machine Learning : k-NN vs NB

Introduction

Text classification is the set of machine learning algorithms which classify a set of characters, phrases and sentences into categories through various methods of classification. Digital marketing to spam email separation, text classification is used in various industries to improve efficiency and productivity. Furthermore, there are various processes by which information is classified: using Euclidean distances(K-NN), probabilistic theory (Naïve Bayes) and a large system of Neural networks making complex decisions based on different mathematical ideas such as vector geometry, function mapping etc.

Since, performance of ML algorithms on large datasets are a big factor in decision making, important information about efficiencies would need to be researched to avoid misuse of time and effort(accuracy). Based on the findings from secondary research about the algorithms and their applications, the RQ is as follows: "To what extent are k-NN and Naïve Bayes algorithms efficient in classifying text" will be used as the main focus to conduct the study".

The two text classification algorithms explored are *Naïve Bayes* Classifier and K-Nearest-Neighbor. Both algorithms return names of the pre-defined categories when test data is input. Firstly, Naïve Bayes classifier works on the principles of the Bayes theorem of condition probability. It takes different parameters as inputs and compares every single parameter in the algorithm to return a complete account of the probability in a certain phrase/sentence belonging to a certain category even though it may look similar to text/sentences from a different category.

Secondly, K-Nearest-Neighbor works on the principles of data clustering (grouping the training dataset into different groups depending on similarities). Hence, if 4 phrases are similar, then in the first few iterations the computer will attempt at group the respective phrases and as the number of iterations increase the differences between multiple phrases in the same group will become smaller and smaller and eventually the computer outputs a point in the cost function where there is no or negligible difference in phrases.

Efficiencies of both algorithms will be compared with respect to the number of pre-defined categories by measuring the time take for each algorithm to complete classification and the accuracy after each iteration of the whole algorithm. The tool used to measure the time and accuracy of each trial is an in-built accuracy and time metric of Python so that the data collected is standard and the environment in which it is collected remains unchanged.

Background Information

Classification is a one of the most widely used methods of identifying trends in a certain dataset. Classification algorithms are of a range of types and mathematical foundation. However, all work towards a similar goal of classifying the given dataset in a quick and reliable method without employing any negligence or other data manipulation techniques to provide an unfair result. Furthermore, classification algorithms are of two types: Supervised and Unsupervised; The difference being, in supervised learning, the machine learning algorithm uses labelled data to make decisions whereas in unsupervised learning, unlabeled data is provided. For E.g. Supervised learning algorithms may contend with linear relationships whereas unsupervised learning algorithms will have to contend with unknown relationships (usually clustering).Therefore, depending on the type of dataset, the scientist/engineer will have to take a decision on the exact algorithm to employ for a certain problem. However, for this study, supervised learning algorithms have been chosen as unsupervised learning algorithms are usually used for the types of datasets which have very frequent data updates which may change the perception of scientist labelling the data for supervised classification.

Naïve Bayes Algorithm

Classification algorithm which works on the concept of the *Bayes* theorem[1] of conditional probability. The Bayes theorem is an extension of the theory of conditional probability (given below)

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \text{......eqn(1)}$$

[1] Aiyappan, Jaya. "Naive Bayes Classifier for Text Classification." *Medium*, Analytics Vidhya, 8 Sept. 2019, medium.com/analytics-vidhya/naive-bayes-classifier-for-text-classification-556fabaf252b.

The Bayes theorem is derived from the conditional probability theory in mathematics.

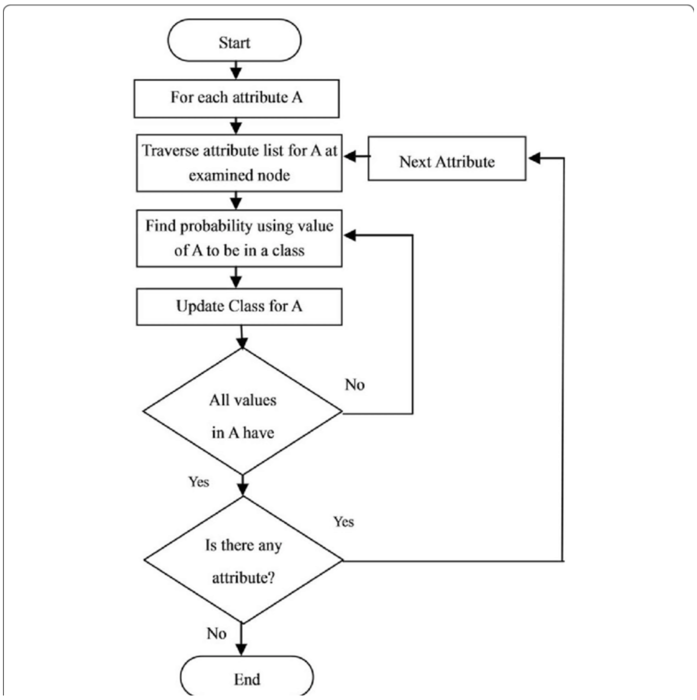
$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \text{.......eqn(2)}$$

The above theorem states that, the probability of an event A occurring given another event occurring B is product of probabilities of B occurring given A occurred and probability of A occurring divided by the probability of B.

In the context of the Naïve Bayes classifier, the probabilities indicate different values in order to accurately classify text. Since all events in the case of classification are connected as they impact the decisions following a certain selection, So, the algorithm uses probability values to predict which word/phrase would belong to a certain category based on the previous choices and or the training provided to the algorithm on a existing dataset.

In $P(A|B)$, as a result of the number of classes/categories being larger than 1, B is a feature vector containing n elements. A feature vector, simply, is a list of aspects that the algorithm needs to identify in data of a dataset and make any decisions based on insights drawn from those comparisons. These n elements are compared with A individually and based on conditional probability of obtaining A in B_n(n is the nth element of set B) is the product of individual probabilities of B_1 given A, B_2 given A and so on until Bn given A. Furthermore, this product is multiplied by probability of A and then subsequently divided by the product of parameter vector B: $P(B_1) \times P(B_2) \ldots \times P(B_n)$. This shows that naïve bayes classifier takes into account all possible combinations by measuring probability and returning the category/class which has the highest probability of having a given word, phrase or sentence.

The flowchart below illustrates working of naïve bayes algorithm:



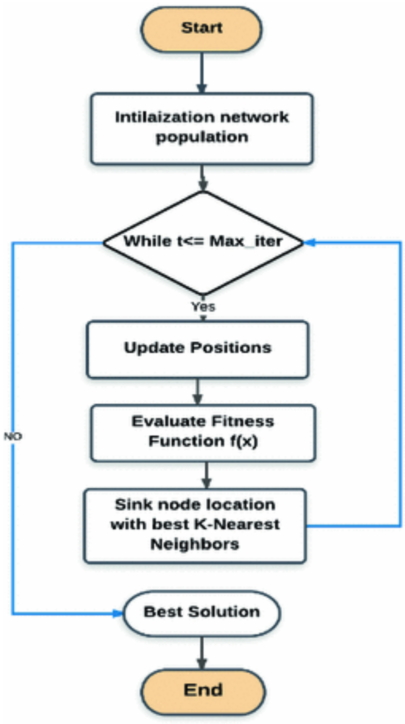
(Fig 1: Naïve Bayes Flowchart[2])

K-Nearest-Neighbor

k-NN[3] classification algorithm works on the concept of clustering similar data together. All data is plotted on a 3D plot as a part of a parameter vector. For every data point on the 3D plot, there is a decision boundary created which separated dissimilar data. Since, this is a supervised learning algorithm this algorithm is already to trained to classify text into different categories. The decision boundary is generated taking into account all factors such as distance between data points, outliers etc. After every iteration, kNN plots a decision boundary which is a line, or a curve separates the classes from each other on the 3D plot. It repeats this process until no further separation is required.

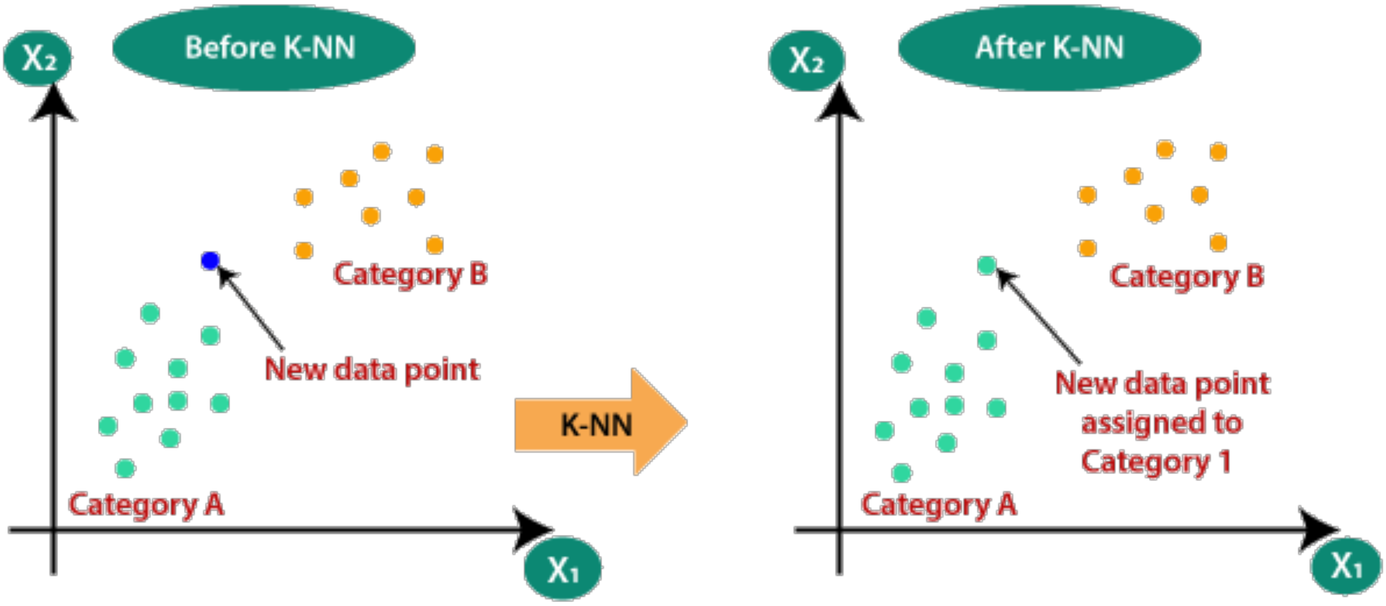
[2] Javed, Aisha. "Naïve Bayes from Scratch Using Python Only- No Fancy Frameworks." *Medium*, Towards Data Science, 12 Jan. 2020, towardsdatascience.com/na%C3%AFve-bayes-from-scratch-using-python-only-no-fancy-frameworks-a1904b37222d.

[3] José, Italo. "KNN (K-Nearest Neighbors) #1." *Medium*, Towards Data Science, 26 June 2019, towardsdatascience.com/knn-k-nearest-neighbors-1-a4707b24bd1d.


Start
Intilaization network population
While t<= Max_iter
Yes
NO
Update Positions
Evaluate Fitness Function f(x)
Sink node location with best K-Nearest Neighbors
Best Solution
End

(Fig2: K-Nearest-Neighbors Flowchart[4])

Even though the flowchart above is able to represent the logical processes which occur in the k-NN algorithm, below is a diagram demonstrating the mathematical perspective of k-NN classification.



(Fig3 : k-NN – Processes[5])

[4] José, Italo. "KNN (K-Nearest Neighbors) #1." *Medium*, Towards Data Science, 26 June 2019, towardsdatascience.com/knn-k-nearest-neighbors-1-a4707b24bd1d.

[5] "K-Nearest-Neighbour." *Java T Point*, www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

It is evident in the diagram above that k-NN uses coordinate geometry as a basis to identify which data point belongs to which category. As the name suggests, this is carried out by determining the smallest Euclidean distances between a cluster of coordinates on a certain plane. Referring back to Fig3, it can be seen that the new data point given on the left is closer to the cluster of category A than category B, Hence, making the algorithm decide that the new data point belongs to category A.

Comparing Algorithms: Theoretically

Proposition that Naïve Bayes will be more efficient than k-NN due to the mathematical processes involves in each one. Firstly, NB is a set of probability formula whereas, k-NN has to compute numerous Euclidean distances to classify the same dataset. This could mean that the time taken for k-NN is significantly higher than NB. Secondly, with respect to the accuracy measurements; it is not definitive, however, NB might still be accurate as identifying Euclidean distances might result in a wrong prediction as a given word may classify into multiple categories which won't be identified by k-NN but will be by NB as its decisions are made based on previous observations.

Methodology

The dataset: "20newsGroups"; part of the software Scikit-learn's Sklearn library used for machine learning and data science. Below is a variable table which contains the choice of variables to be compared in order to ensure the fairness of this study.

(Table 1)

Independent Variable Table	
Algorithm	K-NN vs Naïve Bayes
Number of Categories	2,4,6,8,10,12,14,16,18,20
Dataset Size (No of words)	50,100,150

(Table 2)

Dependent Variable Table	
Time(s)	Time taken to fully classify a given dataset
Accuracy (%)	Success rate of classifying a given dataset

Potential Trends

The first trend to be investigated is "Time against number of Categories". This will show efficiency of the algorithms as run-time is an excellent indicator of whether an algorithm is efficient or not. Also, a lower run-time indicates a higher efficiency as the algorithm is able to classify the given dataset quicker. Secondly, "Accuracy against Number of Categories". This will also indicate the algorithm's level of efficiency as it has to correctly classify the text into numerous categories otherwise it doesn't fulfill its purpose. In this case, a high accuracy shows high efficiency. Lastly, "Time against Accuracy". This will show the pattern in the trade-off the algorithm incurs during the process. For a highly efficient algorithm, there has to be an inverse proportionality between time and accuracy because the least accurate trial should ideally cost the largest time. The trends described above are the core analysis. To extend this study further than the 3 trends, the dataset size will be changed. This will theoretically result in a larger run-time and accuracy may or may not change. Lastly, analytical mathematics will also be used so that the study takes into account most of the factors which may lead to a biased result.

Data Collection

All data are one word phrases to be tested against both the KNN and Naïve Bayes classifier.

(Table 3)

Dataset Size : 50				
	k-NN		Naïve Bayes	
Categories	Accuracy(%)	Time(s)	Accuracy(%)	Time(s)
2	87.5706	142.8707	97.0339	6.4200
4	69.8718	205.7925	85.2075	8.1517
6	66.9745	386.2229	82.3219	9.6814
8	65.0654	686.8287	81.9608	10.2861

10	67.8080	588.5778	84.5914	11.4632
12	70.3441	878.6213	83.8925	12.7629
14	66.6667	1006.0287	82.0371	14.3832
16	67.2605	970.1837	80.0514	15.5271
18	67.6660	1740.8852	81.2796	17.3927
20	65.5868	1327.5150	77.3898	30.1577

(Table 4)

Dataset Size : 100				
	k-NN		Naïve Bayes	
Categories	Accuracy(%)	Time(s)	Accuracy(%)	Time(s)
2	87.5706	194.0879	97.0339	5.8107
4	73.4940	393.7712	85.2075	6.9804
6	66.9745	552.7707	82.3219	8.7534
8	65.0654	621.3629	81.9608	8.8410
10	67.8080	988.4942	84.5914	10.7189
12	70.3441	1287.8618	83.8925	12.0600
14	66.6667	1699.9935	82.0371	13.7593
16	67.2605	2256.4186	80.0514	13.9571
18	67.6660	2020.3238	81.2796	15.8407
20	65.5868	2661.6908	77.3898	16.4511

(Table 5)

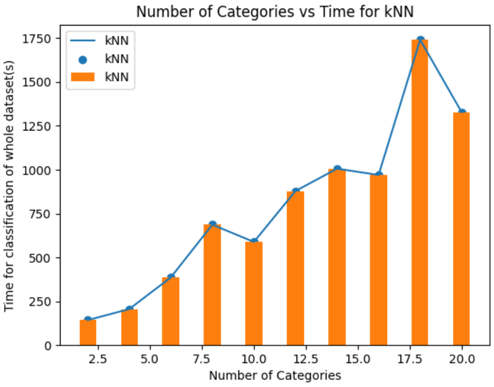
Dataset Size : 150				
	kNN		Naïve Bayes	
Categories	Accuracy(%)	Time(s)	Accuracy(%)	Time(s)
2	87.5706	318.8799	97.0339	8.6758

4	73.4940	546.5272	85.2075	9.2927
6	66.9745	805.0976	82.3219	8.4206
8	65.0654	947.6434	81.9608	10.7661
10	67.8080	1446.4994	84.5914	10.4958
12	70.3441	1769.9204	83.8925	11.6834
14	66.6667	3941.3443	82.0371	12.7341
16	67.2605	2807.6569	80.0514	22.1514
18	67.6660	3032.2038	81.2796	16.1151
20	65.5868	3483.1926	77.3898	19.5172

Comparative Analysis

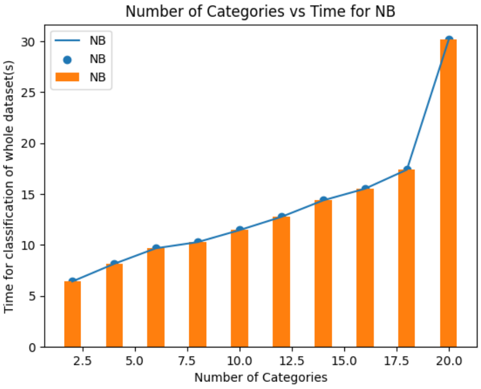
Number of Categories against Time (Dataset size = 50)

k-NN



(Fig4)

NB

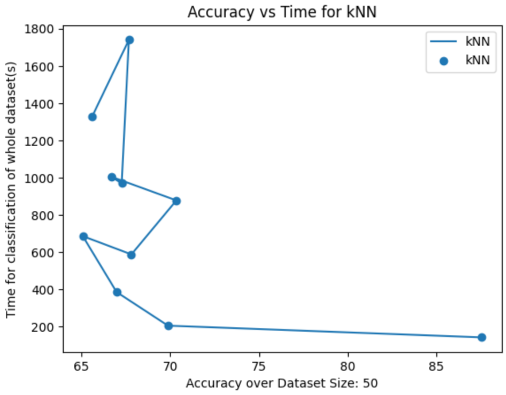


(Fig5)

Fig4 and Fig5 are the visualizations of k-Nearest Neighbour and Naïve Bayes respectively. As the data suggests the Naïve Bayes algorithm is able to perform the classification of the data on 50 words very quickly. Hence, solely using the numbers it is possible to conclude that Naïve Bayes is more efficient. However, this is not the case. Even though, time is a crucial factor to determine the efficiency of a particular algorithm, without taking into account accuracy it is highly difficult to be certain about the algorithm's efficiency because, in the event that the algorithm is not able to classify the data into the categories in which they belong, time would not matter and the use of the algorithm for classification problems will prove ineffective. Therefore, to confirm the claim that Naïve Bayes is more efficient than the k-NN algorithm. Below are two visualizations representing the Time vs Accuracy of k-NN and Naïve Bayes respectively. Both diagrams show the correlation for the same dataset(size=50).

<table>
<tr><th colspan="2">Time against Accuracy (Dataset Size = 50)</th></tr>
<tr>
<td>

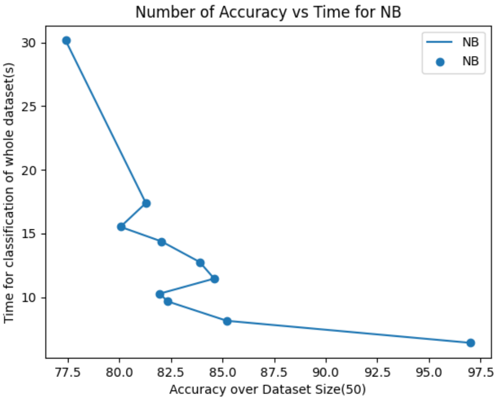
k-NN



(Fig6)

</td>
<td>

NB



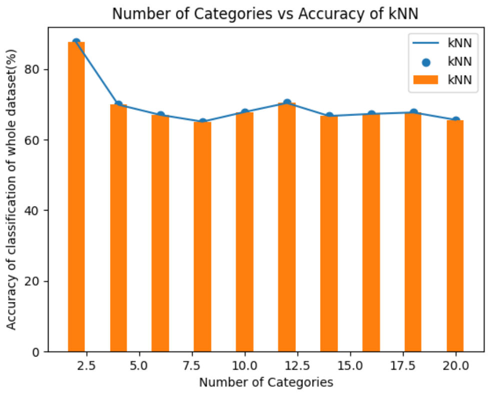
(Fig7)

</td>
</tr>
</table>

The above diagrams are more accurate representation of the real efficiency of both algorithms because both time and accuracy have been taken into account in these diagrams. Firstly, Fig6 and Fig7 are graphs which are extremely misleading due to the unorganized and random lines connecting the points on the graph. The overall trends in both graphs are similar because, as the accuracy of the algorithm increases the time taken for the algorithm decreases. This occurs due to the nature of the problem which is classification. Classification of words into different categories involves complex decision making as there are a definite set of categories in which the algorithm has to classify the word. Hence, in given scenario, the compared to Naïve Bayes , k-NN has performed poorly in both accuracy and time measurements. The graphs below confirm that k-NN has performed a lot worse in terms of the accuracy as even calculating the accuracy of the algorithm was of prime importance to this study.

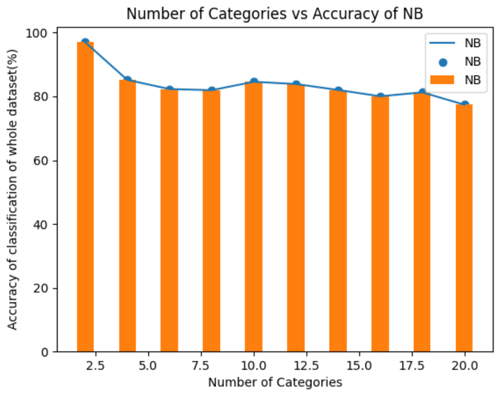
Number of Categories against Accuracy (Dataset Size = 50)

k-NN



(Fig8)

NB



(Fig9)

As seen in Fig8 and Fig9, even though the both graphs are visually identical, the absolute accuracy results are much higher for Naïve Bayes in every interval of categories. These results shown that as the number of categories increases the accuracy of classification reduces but doesn't reach 0. A possibility is that the dataset was a relatively small dataset ; used as the system's computation power was limited. However, these results are still credible because the library used to classify these words is python's own Sckit-Learn set of libraries. In extension, the accuracy metric used was also python's inbuilt accuracy metric which shows that the errors in the data are minimized as the environment in which the data was collected was highly controlled. However, this data could have slight errors.

Extended Analysis

A major deciding factor in comparing 2 algorithms apart from their absolute efficiencies are their relative rates of change. In theory, a more stable rate of change constitutes a well versed progression which doesn't have sudden changes which may ever be because external factors.

In order to compare efficiency from a purely mathematical perspective, calculus and statistical concepts : rates of change and exponential regression will be used. Calculus is the mathematical study which investigates changing quantities whether it may be objects, bodies or event intangible concepts such as time and productivity. Secondly, statistical mathematics is the approach to study practical phenomena using large amounts of data so that the derived insights can be carried towards crucial decisions.

With reference to Fig 4,5,8 and 9, there are assertions to be made whether the algorithms are really efficient in relative to their respective recorded measurements of time with respect to the number of categories[6]. Firstly, as made evident, the figures chosen will be reproduced as a scatter plot and an exponential regression line will be applied to each of them so that the rate of change can be calculate efficiently and accurately. In order to avoid confusion, the table below houses common notation which will be used to indicate various quantities in the study.

(Table 6)

Calculus Notation	
Notation	Meaning
$T(C) = e^{aC+b}$	Time as an exponential function with respect to Number of Categories ; a, b ∈ R (a and b are real numbers).
$T'(C) = \frac{d}{dC}T(C)$	Derivative of Time with respect to Number of Categories (This calculation provides the gradient function). – Shows the gradient at an instant value of C
$T''^{(C)} = \frac{d}{dC}T'(C)$	2nd Derivative of Time with respect to Number of Categories(This provides a method to study the behaviour of the gradient function)

(Table 7)

Statistical Notation	
Notation	Meaning
r^2	Coefficient of Determination : Square of the Pearson's coefficient of correlation to indicate the relative strength of a set of bivariate data on a scale of 0 to 1. This takes into account any residual errors the dataset may contain post regression.

[6] dataset size of 50; will be analyzed.

(Table 8)

Algorithm	Dataset Size	$T(C)$	r^2 value[7]
k-NN	50	$158.5e^{0.1247C}$	0.8838
NB	50	$5.8403e^{0.0688C}$	0.9246

In order to ensure the fairness of this study, the mean gradient will be calculated by using the values of the gradients of the functions at $C = \{2,4,6,8,10,12,14,16,18,20\}$. Furthermore, the 2nd derivative of each function will be used to analyse the way in which the function is behaving.

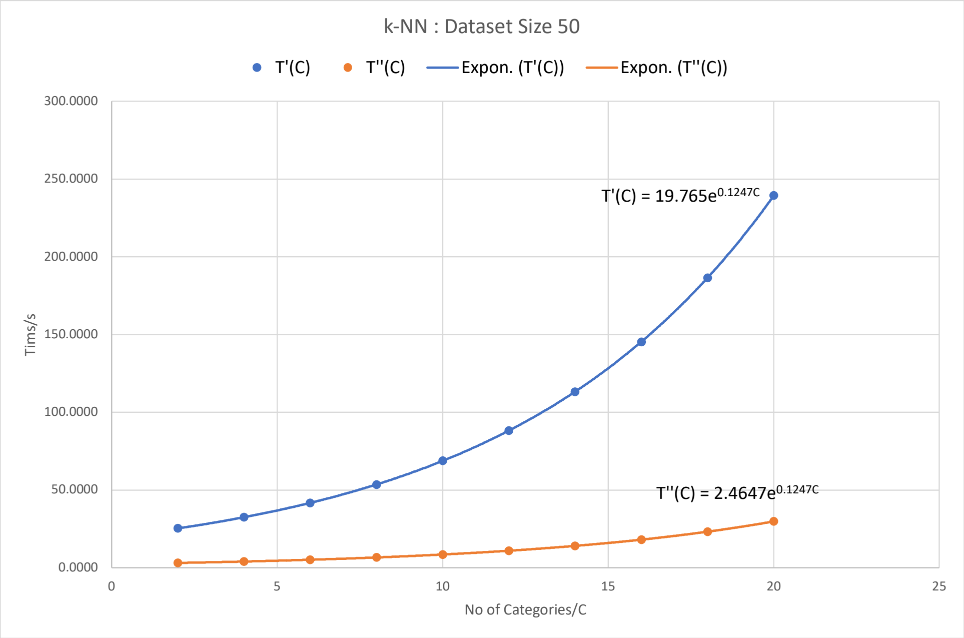
(Table 9)

<table>
<tr><th colspan="3">Processed Data Table For Further Analysis</th></tr>
<tr><th>Algorithm</th><th>$T(C)$</th><th>Calculations</th></tr>
<tr><td>k-NN
Dataset Size : 50</td><td>$158.5e^{0.1247C}$</td><td>$T'(C) = 19.76495e^{0.1247C}$
$T''(C) = 2.46469e^{0.1247C}$
<table>
<tr><th>C</th><th>$T'(C)$</th><th>$T''(C)$</th></tr>
<tr><td>2</td><td>25.3635</td><td>3.1628</td></tr>
<tr><td>4</td><td>32.5478</td><td>4.0587</td></tr>
<tr><td>6</td><td>41.7672</td><td>5.2084</td></tr>
<tr><td>8</td><td>53.5979</td><td>6.6837</td></tr>
<tr><td>10</td><td>68.7798</td><td>8.5768</td></tr>
<tr><td>12</td><td>88.2620</td><td>11.0063</td></tr>
<tr><td>14</td><td>113.2627</td><td>14.1239</td></tr>
<tr><td>16</td><td>145.3450</td><td>18.1245</td></tr>
<tr><td>18</td><td>186.5147</td><td>23.2584</td></tr>
<tr><td>20</td><td>239.3460</td><td>29.8464</td></tr>
</table>
</td></tr>
<tr><td></td><td></td><td>$T'(C) = 0.4018e^{0.0688C}$</td></tr>
</table>

[7] All the regression curves have a r^2 above 0.85 to ensure that the error in computation is minimal.

<table>
<tr>
<td>Naïve Bayes
Dataset Size : 50</td>
<td>$5.8403e^{0.0688C}$</td>
<td>$T''(C) = 0.02764e^{0.0688C}$
<table>
<tr><th>C</th><th>$T'(C)$</th><th>$T''(C)$</th></tr>
<tr><td>2</td><td>0.4611</td><td>0.0317</td></tr>
<tr><td>4</td><td>0.5291</td><td>0.0364</td></tr>
<tr><td>6</td><td>0.6071</td><td>0.0418</td></tr>
<tr><td>8</td><td>0.6967</td><td>0.0479</td></tr>
<tr><td>10</td><td>0.7995</td><td>0.0550</td></tr>
<tr><td>12</td><td>0.9174</td><td>0.0631</td></tr>
<tr><td>14</td><td>1.0527</td><td>0.0724</td></tr>
<tr><td>16</td><td>1.2080</td><td>0.0831</td></tr>
<tr><td>18</td><td>1.3862</td><td>0.0954</td></tr>
<tr><td>20</td><td>1.5907</td><td>0.1094</td></tr>
</table>
</td>
</tr>
</table>

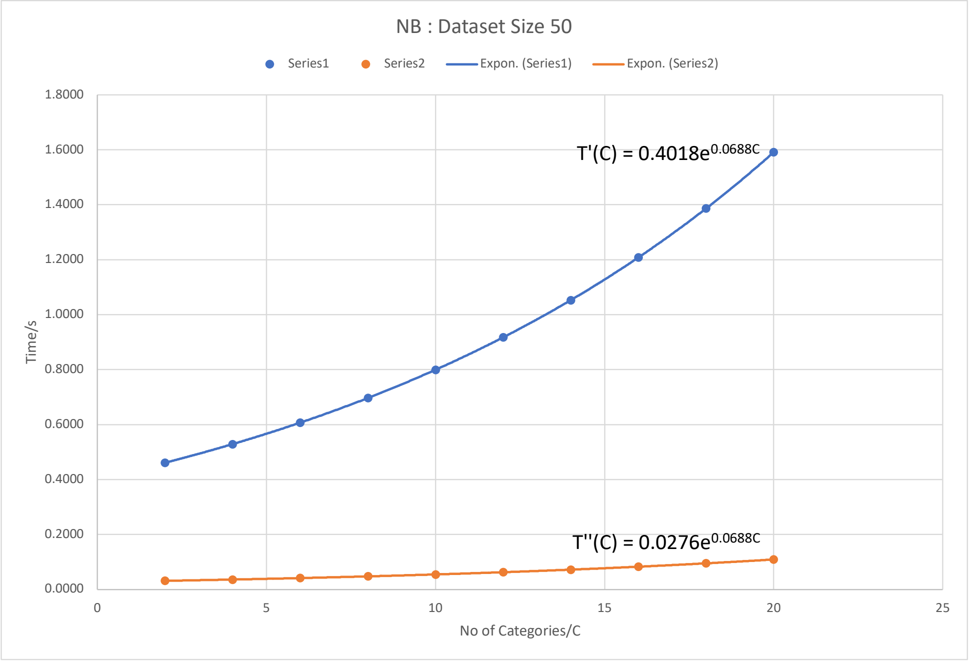
The above calculations reveal vital insights about the two algorithms. Below are 2 graphs[8] which contain, plotted: $T'(C)$ *and* $T''(C)$, for both algorithms[9].



(Fig 10)

[8] The legend is formatted as per : "Expon. (T[C])" where "Expon" refers to the type of best fit : exponential and the T'[C] and T''[C] refers to the function

[9] Visualizations for Dataset Size 100 and 150 are not in the document as they represent a similar outcome.



(Fig 11)

Figure 10 and 11 : visualizations of the 1st derivative (blue trend line) and 2nd derivative(orange trend line) for both algorithms. The major deciding factor is the rate at which the time taken varies with the increase in categories. Therefore, the comparison between the orange trend lines is to be made. The orange trend line represents the second derivative ; the rate of increase with respect to the gradual increases in the time taken to classify the number of categories. In order to be most efficient, the orange trend line will need to have a gradient which tends to 0[10]. This is because this shows a close to constant change in the absolute time taken by a given algorithm. The Naïve-Bayes algorithm has produced a 2nd derivative function which has a lower gradient than k-NN ; indicating that it is also a more efficient algorithm from a relativistic point of view.

The Analysis makes it highly evident that the Naïve Bayes algorithm is far more efficient than k-Nearest Neighbour. However as conclusive this might seem, all measurements have uncertainties ; they might not be significant and probably won't change the result of the study but still have an impact on the absolute values of the measurements.

[10] Needs to be as horizontally oriented as possible.

Evaluation

The method of dataset preparation wasn't perfect because, a large portion of the dataset may belong to a fewer number of categories. The problem is that Naïve Bayes works on the principle of probabilities. Hence, it is very likely that a large portion of text may belong to a certain category making the performance of the algorithm increase. For k-NN, Since, the algorithm measures Euclidean distances on a plot, a single cluster with a large number of data could indicate that it can classify the given text faster which would create bias in the performance measure.

Another very crucial factor to effective data collection is system used to collect it which is a MacBook Air(Intel Core i5, 8GB ram). This device is not manufactured for heavy data processing. This explains the observation that during the data collection the laptop's load and CPU temperature were >85% and 70º respectively. This could impact performance of both algorithms because as the load on the processor increases, the efficiency of the same tends to decrease by a small amount, which, could be a reason for the sudden spikes in the graphs of both NB and k-NN. Also, this justifies the sudden decrease from the maximum point on the k-NN and NB graphs as the processor may have artificially limited the performance levels as in the same time period there was an increase in the time. However, since the system used for data collection was not changed at any point in time during the data collection, it can be concluded that the system did not impact the relative performance and time of the algorithms by a significant margin.

An analytical measure of efficiency is the BigO complexity. The BigO complexity is a measure of efficiency as the parameters of a certain algorithm keep on increasing. By Definition, the BigO complexity of Naïve Bayes is O(Classes* D(features)). In a worst case scenario, value of C = 20 and D = 1 because there is only 1 feature as the algorithm classifies on the basis of probabilities instead of direct comparison. Hence, the BigO complexity of Naïve Bayes is O(20) Secondly, the BigO complexity of k-NN is O(k*n*D). In a worst case scenario, k = 20, d = 2 and n=20. Therefore making the BigO complexity of k-NN O(800) which is very high thus indicating a high level of inefficiency.

The above findings regarding BigO efficiency have been researched thouroughly in the past. However, the perspective of the researcher also matters. Hence, conducting a small study from the perspective of an ill-experienced individual also needs to be addressed because it would introduce possibility of new findings with regard to the algorithms themselves.

Conclusion

To conclude, it can be seen as though, Naïve Bayes is more efficient than k-Nearest-Neighbours. However, this conclusion is very relativistic as in this study different types of data weren't accounted for such as Images, audio etc. Hence, with regards to text classification NB performs better than the k-NN but it cannot be said that k-NN is not efficient as the algorithms usually behave differently with different types of datasets. K-NN is less efficient compared to Naïve Bayes based on the datasets, graphs and also the BigO time complexity. However, it is not to say that on very large and complex datasets K-NN and NB will have such significant discrepancies in the values of time and accuracy. Also, text classification is only 1 application of clustering so there are numerous ways to classify text which may or may not be more efficient than both of these algorithms. Lastly, to conclude the Research Question : To what extent are k-NN and Naïve Bayes algorithms efficient in classifying text; Naïve Bayes is more efficient that k-NN. However, there are numerous other factors affects the performance as listed above which are outside the scope of this study which could hinder these findings.

Bibliography

Aiyappan, Jaya. "Naive Bayes Classifier for Text Classification." *Medium*, Analytics Vidhya, 8 Sept. 2019, medium.com/analytics-vidhya/naive-bayes-classifier-for-text-classification-556fabaf252b.

Brownlee, Jason. "Naive Bayes Classifier From Scratch in Python." *Machine Learning Mastery*, 24 Oct. 2019, machinelearningmastery.com/naive-bayes-classifier-scratch-python/.

Javed, Aisha. "Naïve Bayes from Scratch Using Python Only- No Fancy Frameworks." *Medium*, Towards Data Science, 12 Jan. 2020, towardsdatascience.com/na%C3%AFve-bayes-from-scratch-using-python-only-no-fancy-frameworks-a1904b37222d.

José, Italo. "KNN (K-Nearest Neighbors) #1." *Medium*, Towards Data Science, 26 June 2019, towardsdatascience.com/knn-k-nearest-neighbors-1-a4707b24bd1d.

Klein, Bernd. "Python Machine Learning Tutorial." *Machine Learning with Python: k-Nearest Neighbor Classifier in Python*, www.python-course.eu/k_nearest_neighbor_classifier.php.

Patel, Harshiv. "Text Classification Using K Nearest Neighbors (KNN)." *OpenGenus IQ: Learn Computer Science*, OpenGenus IQ: Learn Computer Science, 18 Dec. 2019, iq.opengenus.org/text-classification-using-k-nearest-neighbors/.

Sneha, N. & Gangil, Tarun. (2019). Analysis of diabetes mellitus for early prediction using optimal features selection. Journal of Big Data. 6. 10.1186/s40537-019-0175-6.

VanderPlas, Jake. "In Depth: Naive Bayes Classification." *In Depth: Naive Bayes Classification | Python Data Science Handbook*, jakevdp.github.io/PythonDataScienceHandbook/05.05-naive-bayes.html.

"K-Nearest-Neighbour." *Java T Point*, www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

Appendices

A : Naïve Bayes Classifier in Python

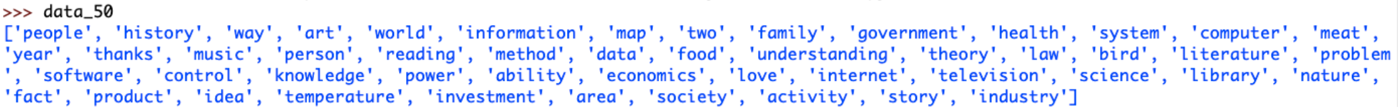
```
import time
start = time.time() # Start time

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import sklearn.datasets
from sklearn.metrics import accuracy_score
from sklearn.datasets import make_blobs
X, y = make_blobs(50, 2, centers=2, random_state=2, cluster_std=1.5)
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu');

from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X, y);

rng = np.random.RandomState(0)
Xnew = [-6, -14] + [14, 18] * rng.rand(50, 2)
ynew = model.predict(Xnew)

plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu')
lim = plt.axis()
plt.scatter(Xnew[:, 0], Xnew[:, 1], c=ynew, s=20, cmap='RdBu', alpha=0.1)
plt.axis(lim);

yprob = model.predict_proba(Xnew)
yprob[-8:].round(2)

from sklearn.datasets import fetch_20newsgroups

data = fetch_20newsgroups()
data.target_names

categories = ['alt.atheism',
             'comp.graphics',
 'comp.os.ms-windows.misc',
 'comp.sys.ibm.pc.hardware',
 'comp.sys.mac.hardware',
 'comp.windows.x',
 'misc.forsale',
 'rec.autos',
 'rec.motorcycles',
 'rec.sport.baseball',
 'rec.sport.hockey',
 'sci.crypt',
 'sci.electronics',
 'sci.med',
 'sci.space',
 'soc.religion.christian',
 'talk.politics.guns',
 'talk.politics.mideast',
 'talk.politics.misc',
 'talk.religion.misc']


train = fetch_20newsgroups(subset='train', categories=categories) # Training Data
test = fetch_20newsgroups(subset='test', categories=categories) # Testing Data
```

```
# print(train.data[5])

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline

model = make_pipeline(TfidfVectorizer(), MultinomialNB())

model.fit(train.data, train.target)
labels = model.predict(test.data)

from sklearn.metrics import confusion_matrix
mat = confusion_matrix(test.target, labels)
sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False,
            xticklabels=train.target_names, yticklabels=train.target_names)
plt.xlabel('true label')
plt.ylabel('predicted label');
plt.show()

def predict_category(s, train=train, model=model):
    pred = model.predict([s])
    return train.target_names[pred[0]]

docs_new =["Dataset"]

for i in docs_new :
    print(i,'=>',predict_category(i))

from sklearn.metrics import accuracy_score
accuracy = accuracy_score(test.target,labels) * 100
print('The accuracy score is :',accuracy)

end = time.time()
print( ' Total Time to run with', len(categories) , 'categories is: ', end-start)
```

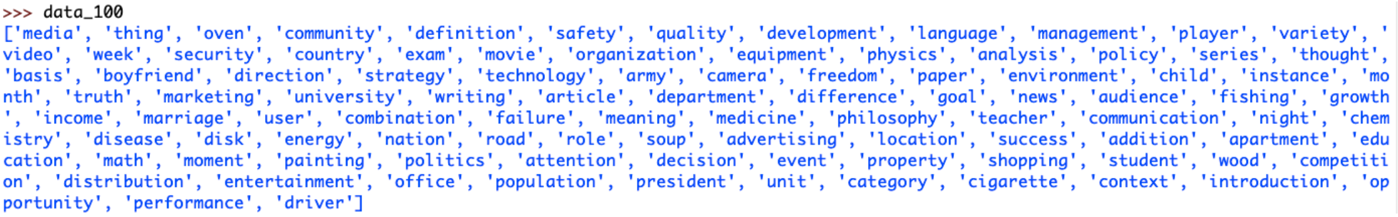
B : k-Nearest-Neighbor Classifier in Python

```
import time
start = time.time()
import numpy as np
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline

from sklearn.datasets import fetch_20newsgroups

categories = ['alt.atheism',
             'comp.graphics',
 'comp.os.ms-windows.misc',
 'comp.sys.ibm.pc.hardware',
 'comp.sys.mac.hardware',
 'comp.windows.x',
 'misc.forsale',
 'rec.autos',
 'rec.motorcycles',
 'rec.sport.baseball'
 'rec.sport.hockey',
 'sci.crypt',
 'sci.electronics',
 'sci.med',
 'sci.space',
 'soc.religion.christian',
 'talk.politics.guns',
 'talk.politics.mideast',
 'talk.politics.misc',
 'talk.religion.misc']

# sklearn provides us with subset data for training and testing
train_data = fetch_20newsgroups(subset='train',
                                categories=categories, shuffle=True, random_state=42)

# print(train_data.target_names)

# print("\n".join(train_data.data[0].split("\n")[:3]))
# print(train_data.target_names[train_data.target[0]])

# Let's look at categories of our first ten training data
for t in train_data.target[:10]:
    # print(train_data.target_names[t])

    count_vect = CountVectorizer()
    X_train_counts = count_vect.fit_transform(train_data.data)

    # transform a count matrix to a normalized tf-idf representation (tf-idf transformer)
    tfidf_transformer = TfidfTransformer()
    X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

    knn = KNeighborsClassifier(n_neighbors=7)

    # training our classifier ; train_data.target will be having numbers assigned for each category in train data
    clf = knn.fit(X_train_tfidf, train_data.target)

    # Input Data to predict their classes of the given categories
    docs_new =[“Dataset”]

    # building up feature vector of our input
    X_new_counts = count_vect.transform(docs_new)
    # We call transform instead of fit_transform because it's already been fit
    X_new_tfidf = tfidf_transformer.transform(X_new_counts)

predicted = clf.predict(X_new_tfidf)

for doc, category in zip(docs_new, predicted):
    print('%r => %s' % (doc, train_data.target_names[category]))

    text_clf = Pipeline([
        ('vect', CountVectorizer()),
        ('tfidf', TfidfTransformer()),
        ('clf', knn),
    ])
    # Fitting our train data to the pipeline
    text_clf.fit(train_data.data, train_data.target)

    # Test data
    test_data = fetch_20newsgroups(subset='test',
                                   categories=categories, shuffle=True, random_state=42)
    docs_test = test_data.data
    # Predicting our test data
    predicted = text_clf.predict(docs_test)
print('We got an accuracy of',np.mean(predicted == test_data.target)*100, '% over the test data.')

end = time.time()
print( ' Total Time to run with', len(categories) , 'categories is: ', end-start)
```

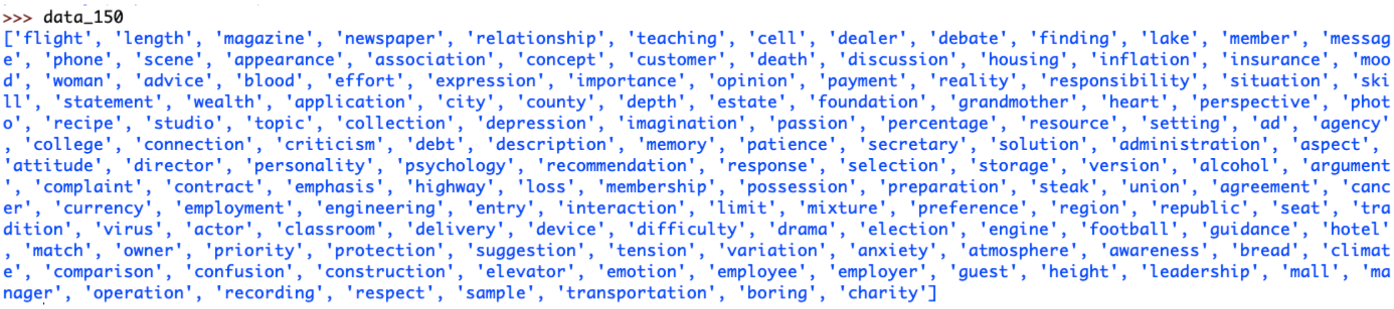
C : Dataset



```
>>> data_50
['people', 'history', 'way', 'art', 'world', 'information', 'map', 'two', 'family', 'government', 'health', 'system', 'computer', 'meat',
'year', 'thanks', 'music', 'person', 'reading', 'method', 'data', 'food', 'understanding', 'theory', 'law', 'bird', 'literature', 'problem
', 'software', 'control', 'knowledge', 'power', 'ability', 'economics', 'love', 'internet', 'television', 'science', 'library', 'nature',
'fact', 'product', 'idea', 'temperature', 'investment', 'area', 'society', 'activity', 'story', 'industry']
```



```
>>> data_100
['media', 'thing', 'oven', 'community', 'definition', 'safety', 'quality', 'development', 'language', 'management', 'player', 'variety', '
video', 'week', 'security', 'country', 'exam', 'movie', 'organization', 'equipment', 'physics', 'analysis', 'policy', 'series', 'thought',
'basis', 'boyfriend', 'direction', 'strategy', 'technology', 'army', 'camera', 'freedom', 'paper', 'environment', 'child', 'instance', 'mo
nth', 'truth', 'marketing', 'university', 'writing', 'article', 'department', 'difference', 'goal', 'news', 'audience', 'fishing', 'growth
', 'income', 'marriage', 'user', 'combination', 'failure', 'meaning', 'medicine', 'philosophy', 'teacher', 'communication', 'night', 'chem
istry', 'disease', 'disk', 'energy', 'nation', 'road', 'role', 'soup', 'advertising', 'location', 'success', 'addition', 'apartment', 'edu
cation', 'math', 'moment', 'painting', 'politics', 'attention', 'decision', 'event', 'property', 'shopping', 'student', 'wood', 'competiti
on', 'distribution', 'entertainment', 'office', 'population', 'president', 'unit', 'category', 'cigarette', 'context', 'introduction', 'op
portunity', 'performance', 'driver']
```



```
>>> data_150
['flight', 'length', 'magazine', 'newspaper', 'relationship', 'teaching', 'cell', 'dealer', 'debate', 'finding', 'lake', 'member', 'messag
e', 'phone', 'scene', 'appearance', 'association', 'concept', 'customer', 'death', 'discussion', 'housing', 'inflation', 'insurance', 'moo
d', 'woman', 'advice', 'blood', 'effort', 'expression', 'importance', 'opinion', 'payment', 'reality', 'responsibility', 'situation', 'ski
ll', 'statement', 'wealth', 'application', 'city', 'county', 'depth', 'estate', 'foundation', 'grandmother', 'heart', 'perspective', 'phot
o', 'recipe', 'studio', 'topic', 'collection', 'depression', 'imagination', 'passion', 'percentage', 'resource', 'setting', 'ad', 'agency'
, 'college', 'connection', 'criticism', 'debt', 'description', 'memory', 'patience', 'secretary', 'solution', 'administration', 'aspect',
'attitude', 'director', 'personality', 'psychology', 'recommendation', 'response', 'selection', 'storage', 'version', 'alcohol', 'argument
', 'complaint', 'contract', 'emphasis', 'highway', 'loss', 'membership', 'possession', 'preparation', 'steak', 'union', 'agreement', 'canc
er', 'currency', 'employment', 'engineering', 'entry', 'interaction', 'limit', 'mixture', 'preference', 'region', 'republic', 'seat', 'tra
dition', 'virus', 'actor', 'classroom', 'delivery', 'device', 'difficulty', 'drama', 'election', 'engine', 'football', 'guidance', 'hotel'
, 'match', 'owner', 'priority', 'protection', 'suggestion', 'tension', 'variation', 'anxiety', 'atmosphere', 'awareness', 'bread', 'climat
e', 'comparison', 'confusion', 'construction', 'elevator', 'emotion', 'employee', 'employer', 'guest', 'height', 'leadership', 'mall', 'ma
nager', 'operation', 'recording', 'respect', 'sample', 'transportation', 'boring', 'charity']
```