

AI & MACHINE LEARNING

Problem-solving with ML: automatic document classification

Ahmed Kachkach

Find an article...

[Latest stories](#)[Products](#)[Topics](#)[About](#)[RSS Feed](#)**Help improve Google Cloud Website**

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

them.

We'll use a public dataset from the BBC comprised of 2225 articles, each labeled under one of 5 categories: business, entertainment, politics, sport or tech. Our goal will be to build a system that can accurately classify previously unseen news articles into the right category.

The Python code used in this article and some accompanying text and plots are available as a [Colab notebook](#).

Data extraction and exploration

Loading data

Data is the essential resource for any ML project. Fortunately, there are plenty of datasets freely available in [Google BigQuery Public Datasets](#).

For this blog post, we'll use the BBC News dataset. It can either be extracted from BigQuery and exported to CSV, or directly [downloaded from its original source](#).

Here is a sample from this dataset:

category	filename	title	content
----------	----------	-------	---------

[Latest stories](#)[Products](#)[Topics](#)[About](#)[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

tech 368.txt Net fingerprints combat attacks Net fingerprints combat attacks Eighty larg...

We use [pandas](#) to load this as a [DataFrame](#), and add an integer `category_id` (which will be handy later on):

Language: Python

```
1 import pandas as pd
2 df = pd.load_csv("bbc-news.csv")
3 df['category_id'] = df.category.factorize()[0]
```

The `DataFrame` is a useful data structure, first popularized by the [R](#) language, that allows us to easily transform and navigate our dataset in an efficient manner.

Data analysis

Before diving head-first into training machine learning models, we should become familiar with the structure and characteristics of our dataset: these properties might inform our problem-solving approach. First, it's always useful to look at the number of documents per class:

🔍 Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

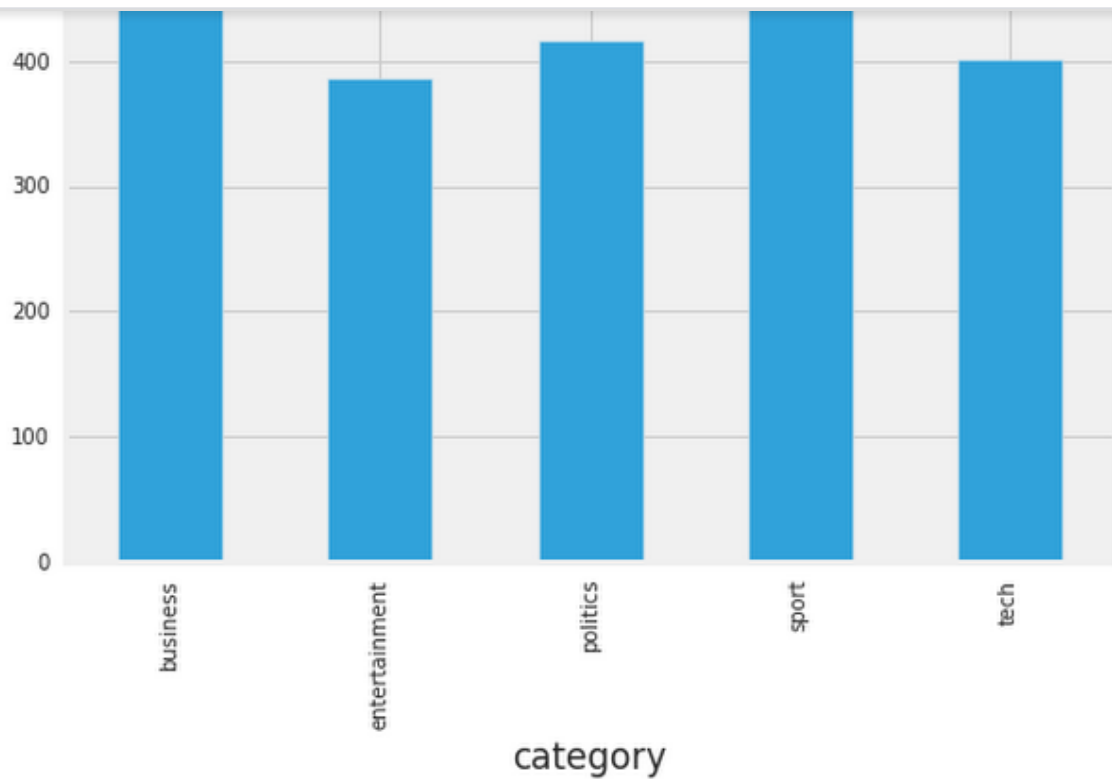
Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied



Here, we see that the number of articles per class is roughly balanced, which is helpful! If our dataset were imbalanced, we would need to carefully configure our model or artificially balance the dataset, for example by [undersampling](#) or [oversampling](#) each

[Latest stories](#)[Products](#)[Topics](#)[About](#)[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

present in most documents).

We'll use `sklearn` (also known as `scikit-learn`), a machine learning library that is particularly accessible to beginners. Within `sklearn`, we'll use the `TfidfVectorizer` class to calculate a `tf-idf` vector for each of our documents:

```
1 from sklearn.feature_extraction.text import TfidfVectorizer
2 tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', encoding='latin1')
3 features = tfidf.fit_transform(df.content).toarray()
4 labels = df.category_id
```

Note that we pass some additional parameters to the `tfidf` class: setting `sublinear_tf` means that we will use the log of the frequency, as word frequencies follow an exponential distribution (see [Zipf law](#)), we normalize our vectors to `l2 norm` so that the length of a document does not bias its representation, and we also consider bigrams (pairs of words) as these might carry a different meaning than each of their components separately (e.g "box office" vs "box" and "office").

The resulting `features` variable contains one row of numerical features (each representing the `tf-idf` for a word or pair of words) for each of our documents. This representation is not only useful for solving our classification task, but also to familiarize ourselves with the dataset. For example, we can use the [chi-squared test](#) to find the terms are the most correlated with each of the categories:

🔍 Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

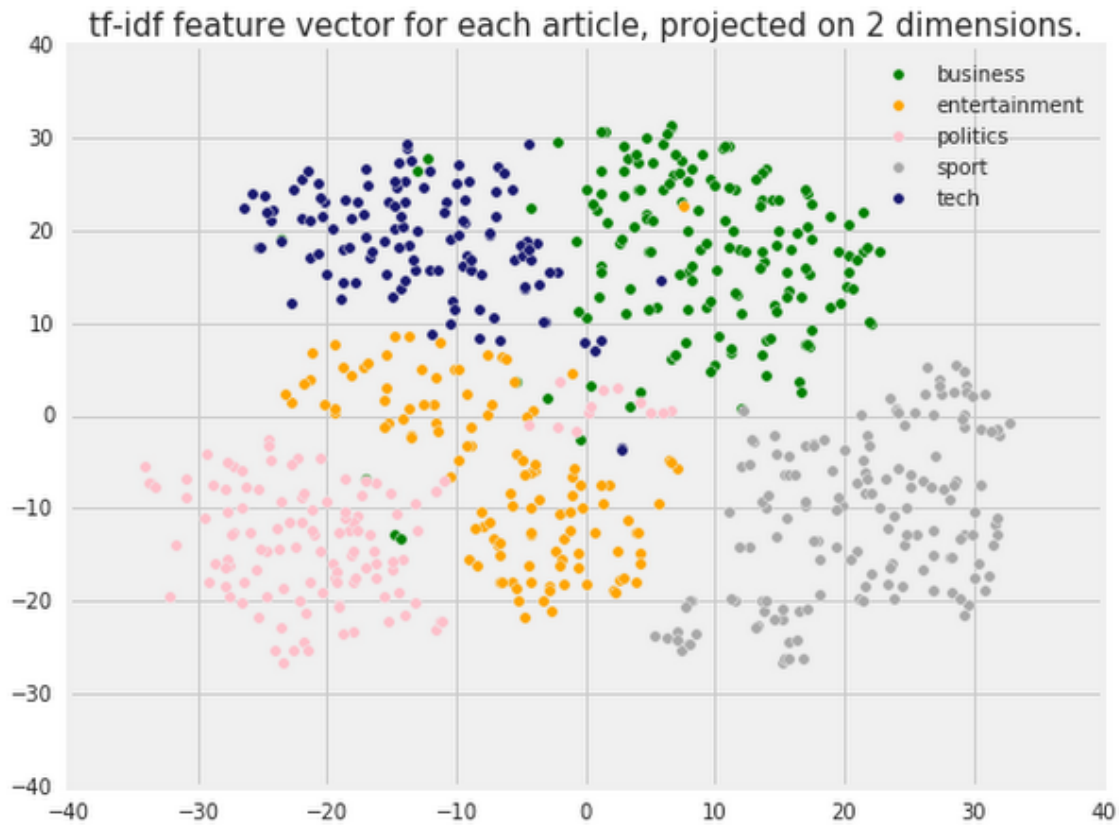
Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

points in the high-dimensional space close to each other in the projected space (t-SNE). Below is the result of clustering a third of the articles in our dataset using t-SNE:



🔍 Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

...), it is vital to choose which metric to optimize for. Accuracy (the percentage of correct classifications) is the most commonly used metric, but it is not always the right choice: if we are trying to detect fraudulent monetary transactions, which might constitute less than 0.01% of the total transactions, a model that would classify all users as being non-fraudulent would have 99.99% accuracy! Because of this, we need to carefully choose our metrics.

Here, we are dealing with a multi-class classification task (assigning a label out of multiple possible values). Given the relative balance of our dataset, accuracy would be an appropriate metric. If one of the labels was more important than the others, we could look at [Precision and Recall](#) for each class, or use the [ROC curve](#), and optimize the Area Under the Curve (ROC AUC).

Choice of model

scikit-learn provides implementations for a large number of machine learning models, spanning a few different families:

- **Linear models:** Linear Regression, Logistic Regression, ...
- **Ensemble models:** Random Forest, Gradient Boosting Trees, Adaboost, ...
- **Bayesian models:** (Multinomial/Gaussian/...) Naive Bayes, Gaussian Processes, ...
- Support Vector Machines, k-Nearest Neighbors, and various other models.

🔍 Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

- **Random Forest:** Random Forest (as the name might suggest) is the [ensembling](#) of a large number of decision trees, each trained on a random subset of the input features. They work well when complex feature-relations are involved and are relatively robust to overfitting.


We usually also perform a hyperparameter search for each model: tuning each of its "knobs" (number of trees for Random Forest, penalty for Logistic Regression) until we find the optimal ones. For simplicity, we skip this step below and directly provide reasonably good parameters for each model.

Model evaluation

One common mistake when evaluating a model is to train and test it on the same dataset: this is problematic because this will not evaluate how well the model works in realistic conditions, on unseen data, and models that overfit to the data will seem to perform better.

It is common practice to split the data in three parts:

- A training set that the model will be trained on.
- A validation set used for finding the optimal parameters (as discussed previously).
- A test set to evaluate the model's performance.

 Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

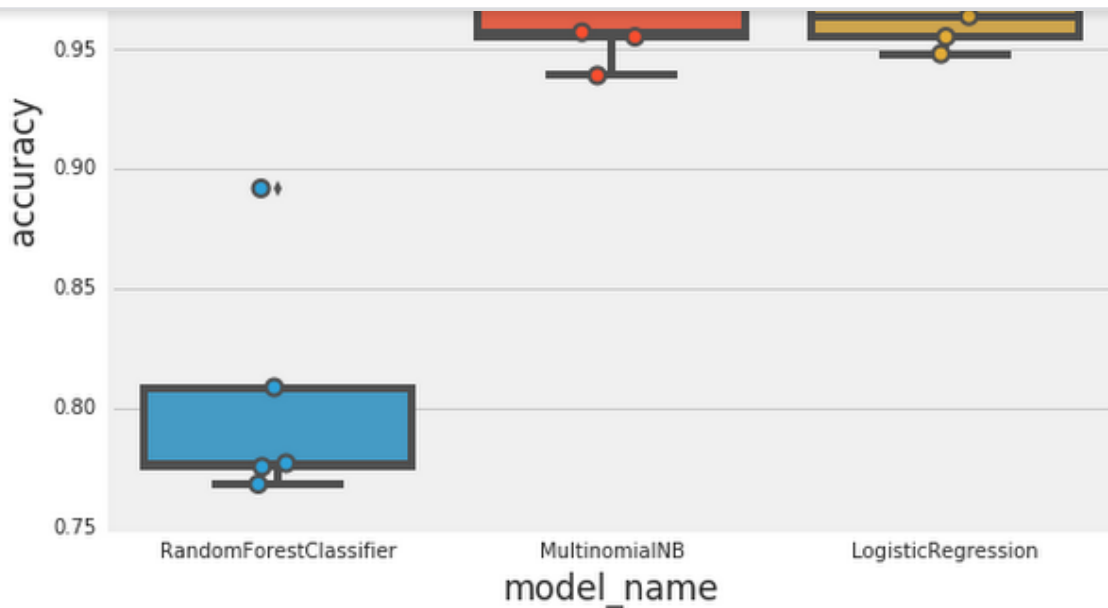
Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied



The results for the Random Forest model show a large variance, the sign of a model that is overfitting to its training data. Running cross-validation is vital, because results from a single train/test split might be misleading. We also notice that both [Multinomial Naive Bayes](#) and [Logistic Regression](#) perform extremely well, with Logistic Regression having a slight advantage with a median accuracy of around 97%! With such results, we could just pack our things and consider the task complete, but it's good to apply a healthy dose of skepticism, especially when results are this good.

 Find an article...

[Latest stories](#)
[Products](#)
[Topics](#)
[About](#)
[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

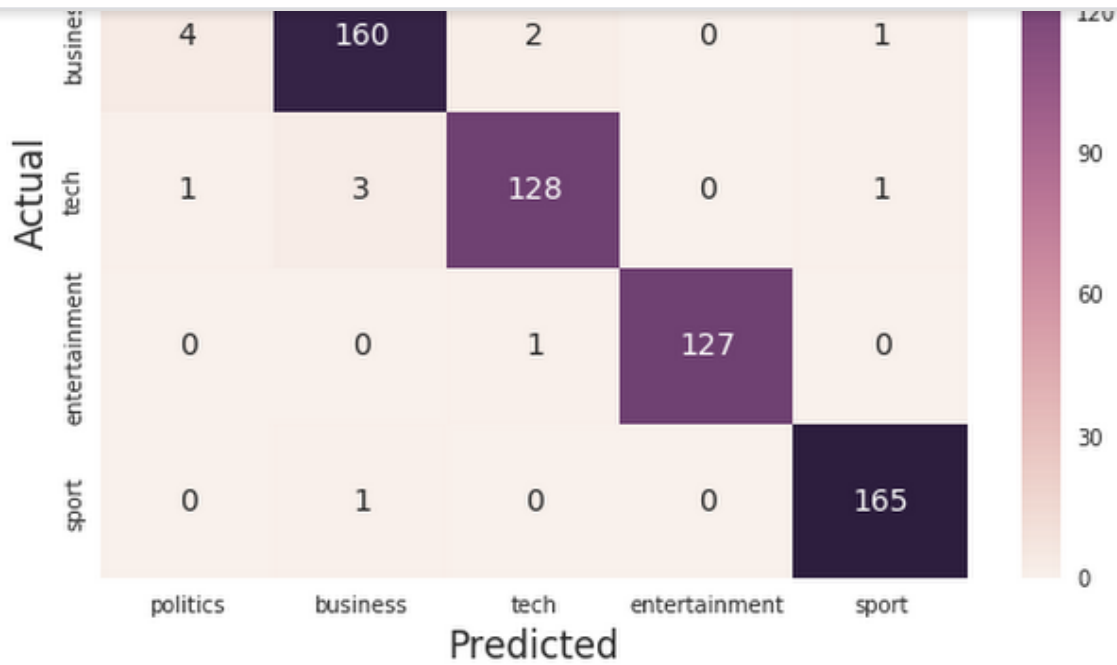
Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied



Given the very high accuracy of our model, almost all the predictions end up on the diagonal (predicted label = actual label), right where we want them to be!

Let's examine the misclassified examples:

'business' predicted as 'politics' : 2 examples.

 Find an article...

[Latest stories](#)
[Products](#)
[Topics](#)
[About](#)
[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

PC ownership to 'double by 2010'

PC ownership to 'double by 2010' The number...

Argonaut founder rebuilds empire

Argonaut founder rebuilds empire Jez San, t...

US duo in first spam conviction

US duo in first spam conviction A brother a...

Games maker fights for survival

Games maker fights for survival One of Brit...

'politics' predicted as 'tech' : 2 examples.

title

content

MPs issued with Blackberry threat

MPs issued with Blackberry threat MPs will ...



Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

```
11 . 'entertainment':
12     . Top unigrams:
13         . film
14         . singer
15         . music
16         . band
17     . Top bigrams:
18         . box office
19         . new york
20         . los angeles
21         . big brother
22 . 'politics':
23     . Top unigrams:
24         . labour
25         . party
26         . mr
27         . election
28     . Top bigrams:
29         . mr blair
30         . tony blair
31         . general election
32         . prime minister
33 . 'sport':
34     . Top unigrams:
35         . cup
36         . match
37         . coach
38         . season
39     . Top bigrams:
40         . year old
41
```



Find an article...

[Latest stories](#)[Products](#)[Topics](#)[About](#)[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied


containing the bigram "news website" are 'tech' articles. This is called a [data leakage](#), which occurs when information about the label we want to predict is accidentally added to the training set. In this specific case, our model still works equally well when excluding the leaky terms, but it's always good to be skeptical when a model performs much better than expected. Accordingly, a robust approach requires an interpretable model instead of black-boxes that might work very well on the test set, but fail on unseen data.

Conclusion

Using off-the-shelf tools and simple models, we solved a complex task, that of document classification, which might have seemed daunting at first! To do so, we followed steps common to solving any task with machine learning:

1. Load and pre-process data.
2. Analyze patterns in the data, to gain insights.
3. Train different models, and rigorously evaluate each of them.
4. Interpret the trained model.

Each of these steps comes with a number of pitfalls to avoid, and although we tried to cover as many of those, different tasks (such as those involving regression, sequential data, images, etc) come with their own challenges.

 Find an article...

[Latest stories](#)

[Products](#)

[Topics](#)

[About](#)

[RSS Feed](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied

Follow Us



Google

[Privacy](#)[Terms](#)[About Google](#)[Google Cloud Products](#) [Help](#)

Help improve Google Cloud Website

Question 1 of 5 or fewer:

Overall, how satisfied are you with this website?

Very satisfied

Somewhat satisfied

Neither satisfied nor dissatisfied

Somewhat dissatisfied

Very dissatisfied