Due Date October 18, 2021
Late Submissions 30% per day
Teams You can do the mini-project in teams of at most 3.
Teams must submit only 1 copy of the mini-project via the team leader's account.
Purpose The purpose of this mini-project is to make you experiment with machine learning.

Experiments with Machine Learning

For this mini-project, you will experiment with different machine learning algorithms and different data sets. As you will use built-in functions from the `scikit-learn` Library, the focus of this mini-project lies more on the experimentations and analysis than on the implementation.

1 Your Tasks

You will perform 2 tasks: a text classification task to better understand the Multinomial Naive Bayes classifier, and another classification with a variety of types of features to better appreciate how to work with other types of data and machine learning models. You must use:

1. Python 3.8 and the `scikit-learn` library. `Scikit-learn` (see http://scikit-learn.org/stable/) provides an interface to program with a variety of different algorithms and built-in datasets. There are plenty of tutorials and examples of code online.

2. GitHub (make sure your project is private while developing).

2 Task 1: Text Classification

1. Download the BBC dataset provided on Moodle. The dataset, created by [Greene and Cunningham, 2006], is a collection of 2225 documents from the BBC news website already categorized into 5 classes: business, entertainment, politics, sport, and tech.
2. Plot the distribution of the instances in each class and save the graphic in a file called `BBC-distribution.pdf`. You may want to use `matplotlib.pyplot` and `savefig` to do this. This pre-analysis of the data set will allow you to determine if the classes are balanced, and which metric is more appropriate to use to evaluate the performance of your classifier.
3. Load the corpus using `load_files` and make sure you set the encoding to `latin1`. This will read the file structure and assign the category name to each file from their parent directory name.
4. Pre-process the dataset to have the features ready to be used by a multinomial Naive Bayes classifier. This means that the frequency of each word in each class must be computed and stored in a term-document matrix. For this, you can use `feature_extraction.text.CountVectorizer`.
5. Split the dataset into 80% for training and 20% for testing. For this, you must use `train_test_split` with the parameter `random_state` set to `None`.
6. Train a multinomial Naive Bayes Classifier (`naive_bayes.MultinomialNB`) on the training set using the default parameters and evaluate it on the test set.
7. In a file called `bbc-performance.txt`, save the following information: (to make it easier for the TAs, make sure that your output for each sub-question below is clearly marked in your output file, using the headings *(a), (b) ...*)
   (a) a clear separator (a sequence of hyphens or stars) and string clearly describing the model (e.g. "MultinomialNB default values, try 1")
   (b) the confusion matrix (you can use `confusion_matrix`)
   (c) the precision, recall, and F1-measure for each class (you can use `classification_report`)
   (d) the accuracy, macro-average F1 and weighted-average F1 of the model (you can use `accuracy_score` and `f1_score`)
   (e) the prior probability of each class
   (f) the size of the vocabulary (i.e. the number of different words[1])
   (g) the number of word-tokens in each class (i.e. the number of words in total[2])
   (h) the number of word-tokens in the entire corpus
   (i) the number and percentage of words with a frequency of zero in each class
   (j) the number and percentage of words with a frequency of ~~zero~~ one in the entire corpus
   (k) your 2 favorite words (that are present in the vocabulary) and their log-prob
8. Redo steps 6 and 7 without changing anything (do not redo step 5, the dataset split). Change the model name to something like "MultinomialNB default values, try 2" and append the results to the file `bbc-performance.txt`.
9. Redo steps 6 and 7 again, but this time, change the smoothing value to `0.0001`. Append the results at the end of `bbc-performance.txt`.
10. Redo steps 6 and 7, but this time, change the smoothing value to `0.9`. Append the results at the end of `bbc-performance.txt`.
11. In a separate plain text file called `bbc-discussion.txt`, explain in 1 to 2 paragraphs:
    (a) what metric is best suited to this dataset/task and why (see step (2))
    (b) why the performance of steps (8-10) are the same or are different than those of step (7) above.

In total, you should have 3 output files for task 1: `bbc-distribution.pdf`, `bbc-performance.txt`, and `bbc-discussion.txt`.

[1]for example, if the word potato appears 3 times, you only count it once.
[2]for example, if the word potato appears 3 times, you count it 3 times.

3 Task 2: Drug Classification

1. Download the Drug dataset on Moodle. This dataset, in `csv` format, contains features that are numerical, categorical and ordinal as well as one of 5 classes to predict: `DrugA`, `DrugB`, `DrugC`, `DrugX`, or `DrugY`.
2. Load the dataset in Python (you can use `pandas.read_csv`).
3. Plot the distribution of the instances in each class and store the graphic in a file called `drug-distribution.pdf`. You can use `matplotlib.pyplot`. This pre-analysis will allow you to determine if the classes are balanced, and which metric is more appropriate to use to evaluate the performance of your classifier.
4. Convert all ordinal and nominal features in numerical format. Make sure that your converted format respects the ordering of ordinal features, and does not introduce any ordering for nominal features. You may want to take a look at `pandas.get_dummies` and `pandas.Categorical` to do this.
5. Split the dataset using `train_test_split` using the default parameter values.
6. Run 6 different classifiers:

   (a) **NB:** a Gaussian Naive Bayes Classifier (`naive_bayes.GaussianNB`) with the default parameters.

   (b) **Base-DT:** a Decision Tree (`tree.DecisionTreeClassifier`) with the default parameters.

   (c) **Top-DT**: a better performing Decision Tree found using (`GridSearchCV`). The gridsearch will allow you to find the best combination of hyper-parameters, as determined by the evaluation function that you have determined in step (3) above. The hyper-parameters that you will experiment with are:
   - criterion: gini or entropy
   - max_depth : 2 different values of your choice
   - min_samples_split: 3 different values of your choice

   (d) **PER:** a Perceptron (`linear_model.Perceptron`), with default parameter values.

   (e) **Base-MLP:** a Multi-Layered Perceptron (`neural_network.MLPClassifier`) with 1 hidden layer of 100 neurons, sigmoid/logistic as activation function, stochastic gradient descent, and default values for the rest of the parameters.

   (f) **Top-MLP:** a better performing Multi-Layered Perceptron found using grid search. For this, you need to experiment with the following parameter values:
   - activation function: sigmoid, tanh, relu and identity
   - 2 network architectures of your choice: for eg 2 hidden layers with $30 + 50$ nodes, 3 hidden layers with $10 + 10 + 10$
   - solver: Adam and stochastic gradient descent

7. For each of the 6 classifier above, append the following information in a file called `drugs-performance.txt`: (to make it easier for the TAs, make sure that your output for each sub-question below is clearly marked in your output file, using the headings *(a), (b) . . .*)

   (a) a clear separator (a sequence of hyphens or stars) and a string clearly describing the model (e.g. the model name + hyper-parameter values that you changed). In the case of Top-DT and Top-MLP, display the best hyperparameters found by the gridsearch.

   (b) the confusion matrix

   (c) the precision, recall, and F1-measure for each class

   (d) the accuracy, macro-average F1 and weighted-average F1 of the model

8. Redo steps 6, 10 times for each model and append the average accuracy, average macro-average F1, average weighted-average F1 as well as the standard deviation for the accuracy, the standard deviation of the macro-average F1, and the standard deviation of the weighted-average F1 at the end of the file `drugs-performance.txt`. Does the same model give you the same performance every time? Explain in a plain text file called `drugs-discussion.txt`. A 1 or 2 paragraph discussion is expected.

In total, you should have 3 output files for task 2: `drug-distribution.pdf`, `drug-performance.txt`, and `drug-discussion.txt`.

4 Deliverables

The submission of the mini-project will consist of 2 deliverables:

1. The code & the output files:

   ☐ Submit all files necessary to run your code. If you used a Jupyter notebook, submit the `.ipynb` files; otherwise submit the `.py` files.

   ☐ Submit a `readme.md` which will contain specific and complete instructions on how to run your experiments. You do not need to submit the datasets. If the instructions in your readme file do not work, are incomplete or a file is missing, you will not be given the benefit of the doubt.

   ☐ Submit the 6 output files: `bbc-distribution.pdf`, `bbc-performance.txt`, `bbc-discussion.txt` and `drug-distribution.pdf`, `drug-performance.txt`, `drug-discussion.txt`.

2. The demo (8 min presentation & Q/A)

   You will have to demo your mini-project for ≈ 12 minutes. Regardless of the demo time, you will demo the program that was uploaded as the official submission on or before the due date. The schedule of the demos will be posted on Moodle. The demos will consist of 2 parts: a presentation ≈ 8 minutes and a Q/A part (≈ 4 minutes). Note that the demos will be done via Zoom and will be recorded.

   Prepare an 8-minute presentation to analyse and compare the performance of your models. The intended audience of your presentation is your TAs. Hence there is no need to explain the theory behind the models. Your presentation should focus on **your** work and the comparison of the performance of the models when the hyper-parameters are modified.

   Your presentation should contain at least the following:

   ☐ An analysis of the initial dataset given on Moodle. If there is anything particular about these datasets that might have an impact on the performance of some models, explain it.

   ☐ An analysis of the results of all the models with the data sets. In particular, compare and contrast the performance of each model with one another, and with the datasets. Please note that your presentation must be analytical. This means that in addition to stating the facts (e.g. the macro-F1 has this value), you should also analyse them (i.e. explain why some metric seems more appropriate than another, or why your model did not do as well as expected. Tables, graphs and contingency tables to back up your claims would be very welcome here.

   ☐ In the case of team work, a description of the responsibilities and contributions of each team member.

   Any material used for the presentation (slides, ...) must be uploaded on EAS before the due date.

   After your presentation, your TA will proceed with a ≈ 4 minute question period. Each student will be asked questions on the code/mini-project, and he/she will be required to answer the TA satisfactorily. In particular, each member should know what each parameters that you experimented with represent and their effect on the performance. Hence every member of team is expected to attend the demo.

   In addition, your TA may give you a new dataset and ask you to train or run your models on this dataset. The output file generated by your program will have to be uploaded on EAS during your demo.

5 Evaluation Scheme

Students in teams can be assigned different grades based on their individual contribution to project.

Individual grades will count for 15% and will be based on:

1. a peer-evaluation done after the submission.
2. the contribution of each student as indicated on GitHub.
3. the Q/A of each student during the demo (correct and clear answers to questions, knowledge of the program, . . .).

The team grade will count for 85% and will be based on:

Code	functionality, proper use of the datasets, design, programming style, . . .	45
Output files with given dataset	Format, Correctness and Depth of discussion (`x-distribution.pdf`, `x-performance.txt`, and `x-discussion.txt`)	15
Demo – Presentation	depth of the analysis, clarity and conciseness, presentation, time-management, . . .	15
Output with demo-dataset	correctness and format	10
Total		85

6 Submission

If you work in a team, identify one member as the team leader. The only additional responsibility of the team leader is to upload all required files (including the files at the demo) from her/his account and book the demo on the Moodle scheduler. If you work individually, by definition, you are the team leader of your one-person team.

6.1 Submission Schedule

Each deliverable is due on the date indicated below.

Deliverable	**Due Date**	**Upload as**
Submit your code, output files, presentation material	October 18, 2021, 11:59pm	Assignment 1
Submit the output files generated at demo time	during your demo	Assignment 10 (yes! 10)

6.2 Submission Checklist

In your GitHub project, include a `README.md` file that contains:

1. on its first line: the URL of your GitHub repository,
2. specific and complete instructions on how to run your program.

Code & Output files

- ☐ Create one zip file containing all your code, the output files for the data set on Moodle and the `README.md` file.
- ☐ Name your zip file: `472_Assignment1_ID1_ID2_ID3.zip` where `ID1` is the ID of the team leader.
- ☐ Have the team leader upload the zip file at: https://fis.encs.concordia.ca/eas/ as `Assignment1`.

Demo & Output files During your actual demo with the TA:

- ☐ Prior to your demo, make your GitHub repository public.
- ☐ Generate the output files for the data set that the TA will give you.
- ☐ Create a zip file called: `472_Demo1_ID1_ID2_ID3` where `ID1` is the ID of the team leader.
- ☐ Have the team leader upload the zip file at: https://fis.encs.concordia.ca/eas/ as `Assignment10`.

Have fun!

References

[Greene and Cunningham, 2006] Greene, D. and Cunningham, P. (2006). Practical solutions to the problem of diagonal dominance in kernel document clustering. In Proc. 23rd International Conference on Machine learning (ICML'06), pages 377–384. ACM Press.