Kuwait University

Project 2: Intelligent Web Page Classification Using Machine Learning

CS 541: Advanced AI
Fall 25-26

Submitted by:
Abeer Alsafran and Faisal Alsubaiei

December 13, 2025

Contents

List of Figures

Introduction and Objectives

In project 1 we have seen how to crawl a webpage and score it based on heuristic search algorithms such as A* search and Greedy Best First Search (GBFS). But in this project we want to convert the scoring function with a more advanced machine learning model that can classify if the web page is Artificial Intelligence (AI) related or not. Our objective in this project is to build a dataset that has binary labels "1" if the web page is related to AI and "0" if it is not related to AI. Then we train at least two models, one of them must be Naive Bayes. Finally, the goal is to classify if a given link is related to AI or not.

Dataset collection and labeling process

The dataset was automatically created using a script that automatically creates a dataset for a set of AI-related web pages and non-AI-related web pages. The process starts with a list of AI-related links (i.e., web pages) in a text file and another set of non-AI-related links in another text file. Then we make the request to retrieve the HTML content and label it as "1" if it is related to AI and "0" otherwise. With that, the `dataset.csv` file is created. A total of 106 entries for related AI web pages and 142 entries for Non-AI related web pages. After cleaning all the text we have a total of 249 entry.

Feature Extraction Method

Feature extraction in text dataset requires specific tools. Since Machine Learning (ML) models can't read raw text, TF-IDF turns documents into vectors that capture which words are important. TF-IDF increases the weight of words that are: frequent within the document (TF), rare across all documents (IDF). Tokenization is a text preprocessing step where raw text is broken into smaller units called tokens. These tokens become the building blocks for downstream tasks such as classification. Stop-word removal is an important step in text pre-processing, especially for machine learning and NLP tasks such as classification (like AI vs non-AI webpage classifier).

Model explanation

Support Vector Machines

Using the text processing tools in TF_IDF, `scikit-learn` and `nltk` libraries, we pre-process the text first by removing the punctuation and the stop words. After cleaning the dataset, we vectorize the text data. Then we split the dataset randomly for train and test (80:20). By importing the model `SVC` from `sklearn.svm`, we train the model. Then

we test the model by evaluating it with the confusion matrix and generate the report of the evaluation metrics such as F1-score, recall, accuracy, and precision.

Naive Bayes

Using the same text pre-processing for the SVM model, we split the dataset and train the model from `sklearn.naive_bayes`. We choose two variants:

- GNB: Gaussian Naive Bayes
- MNB: Multinomial Naive Bayes

Decision Tree

For the Decision Tree model, we again use the same pre-processed and vectorized dataset and train a classifier from `sklearn.tree`. The tree learns decision rules based on the term features in order to separate AI-related from non-AI-related web pages.

Results and evaluation

The models have been trained and tested on the dataset that consists of a total of 249 data entries.

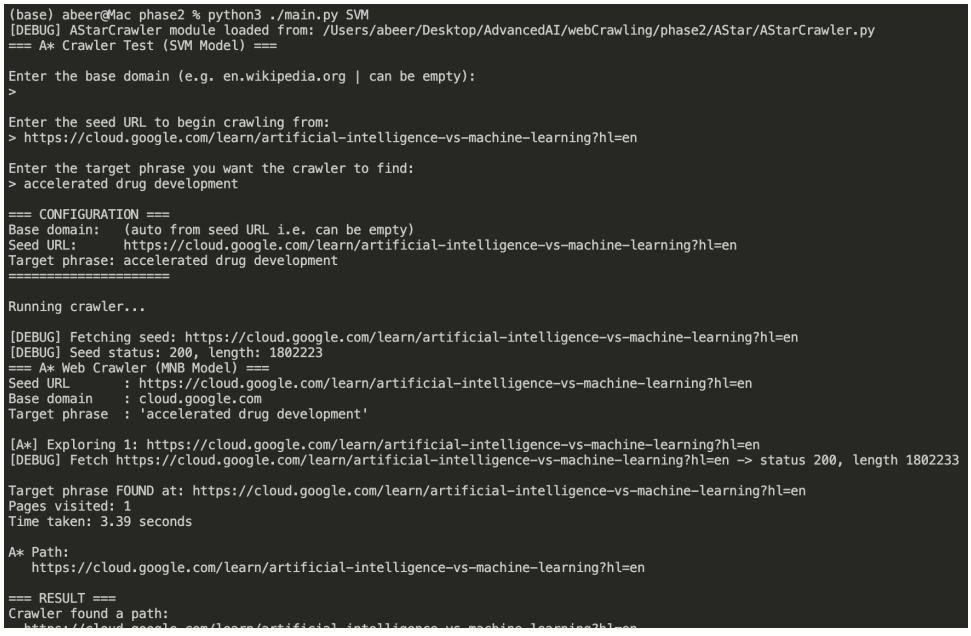

Figure 1: Integrated crawling using ML instead of heuristic search

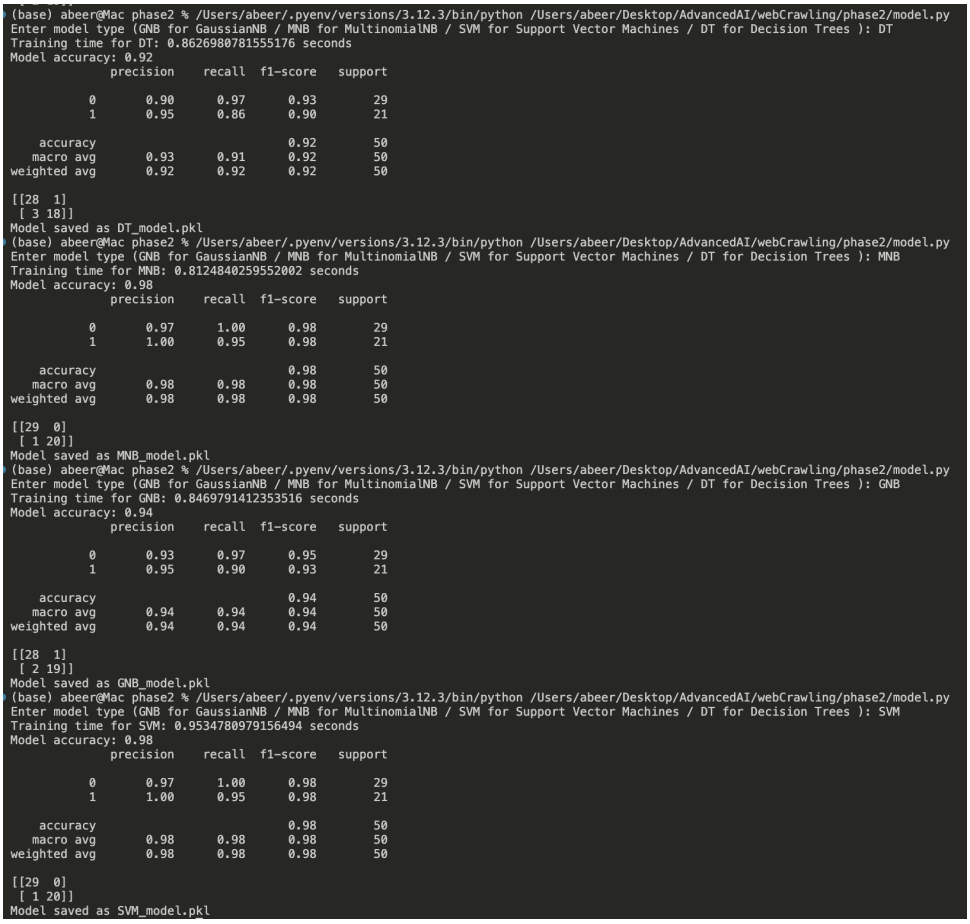

Figure 2: Training four different models (GNB, MNB, SVM, DT).

Model	Class	Precision	Recall	F1-score	Accuracy	Training Time (s)
GNB	0	0.93	0.97	0.95	0.94	0.8469
	1	0.95	0.90	0.93		
MNB	0	0.97	1.00	0.98	**0.98**	0.8125
	1	1.00	0.95	0.98		
SVM	0	0.97	1.00	0.98	**0.98**	0.9535
	1	1.00	0.95	0.98		
DT	0	0.90	0.97	0.93	0.92	0.8627
	1	0.95	0.86	0.90		

Table 1: Per-class evaluation metrics for each model (Class 0 vs Class 1), including accuracy and training time.

Comparison with heuristic method (Project 1)

Comparing the current approach with the heuristic method, we can confidently say that this method with the Machine Learning models is more efficient, faster, and reliable. One of the features that this approach is better is the accuracy; as mentioned in table 1 the

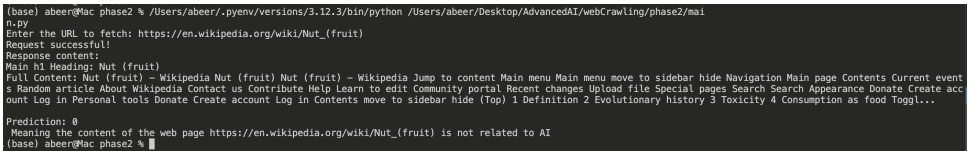

Figure 3: Sample output for model testing link entered by the user

accuracy is very high, which means that we can rely on it in distinguishing for the web pages. Another feature that counts for this approach is the time taken to train the model and make a correct decision.

Analysis and discussion

The best model is SVM and MNB if we compare on the biases accuracy, but if we consider the time factor, then MNB is better because it has a faster running time. The limitation is regarding the small size of the dataset; if we increase the size of the dataset, there is a chance that the accuracy will decrease.

Conclusion & improvement suggestions

We have built a crawler that safely crawls a web page and extracts the content of the web page (HTML), and cleans it using the text cleaning tools, and then perform the model training for SVM, GNB, MNB, and DT. The results show that the best models are MNB and SVM with MNB outperform SVM in time taken in training and testing the model.

References

[1] *Scikit-learn: Naive Bayes documentation*, `https://scikit-learn.org/stable/modules/naive_bayes.html#gaussian-naive-bayes`

[2] *Scikit-learn: Classification*, `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html#sklearn.metrics.classification_report`