**Background:**

في العالم الحديث، أصبح التصنيف التلقائي للنصوص من الأنشطة الهامة في مجال استخدامات التعلم الآلي. يستخدم تصنيف النصوص لتحديد فئة أو تصنيف لنص معين استنادًا إلى محتواه. من بين التطبيقات الشائعة لتصنيف النصوص تصنيف البريد الإلكتروني إلى الرسائل الإلكترونية المزعجة (السبام) والرسائل الإلكترونية العادية.

**Method:**

تم استخدام مجموعة من النماذج لتنفيذ تصنيف البريد الإلكتروني في هذا الكود. تضمن ذلك نموذجين رئيسيين: نموذج Naive Bayes ونموذج Logistic Regression. تم استخدام تقنية Bag-of-Words مع تحويل النصوص إلى تمثيل رقمي باستخدام CountVectorizer.

**Experiment:**

قام البرنامج بقراءة بيانات البريد الإلكتروني من ملف CSV وتنظيفها من القيم المفقودة. ثم قام بتدريب كلٍ من النموذجين باستخدام البيانات النصية المتاحة. تم قياس دقة النماذج باستخدام مصفوفة الارتباك (Confusion Matrix) وحساب الدقة (Accuracy Score).

**Conclusion:**

توصل البرنامج إلى أن نموذج Naive Bayes حقق متوسط دقة يبلغ حوالي 98.22%، بينما حقق نموذج Logistic Regression متوسط دقة يبلغ حوالي 99.84%. يمكن استخدام هذه النتائج لتصنيف البريد الإلكتروني بشكل فعال وتحديد ما إذا كان البريد الإلكتروني مزعجًا (سبام) أم لا.

**References:**

* Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
* A Comparison of Event Models for Naive Bayes Text Classification", Mccallum, A., Nigam, K. in AAAI-98 Workshop on Learning for Text Categorization, 1998.
* Logistic Regression", Wikipedia, [Online]. Available: <https://en.wikipedia.org/wiki/Logistic_regression> . Accessed: Mar. 20, 2024.

# Documenting the Email Classifier Code

## Introduction:

The Email Classifier code is a Python program designed to classify emails into spam and non-spam categories using two machine learning models: Naive Bayes and Logistic Regression. This document aims to provide a detailed overview and explanation of the code.

## Dependencies:

The code relies on the following Python libraries:

- pandas: for data manipulation and reading CSV files.

- scikit-learn: for implementing machine learning algorithms and evaluating their performance.

## Code Structure:

The code consists of the following components:

1. \*\*NaiveBayesModel Class\*\*:

- This class defines the Naive Bayes model used for email classification.

- It initializes a Multinomial Naive Bayes model and a CountVectorizer object.

- Methods:

- `train(X\_train, y\_train)`: Trains the Naive Bayes model on the provided training data.

- `predict(X\_test)`: Makes predictions on the test data using the trained model.

2. \*\*LogisticRegressionModel Class\*\*:

- This class defines the Logistic Regression model used for email classification.

- It initializes a Logistic Regression model with a maximum iteration of 1100 and a CountVectorizer object.

- Methods:

- `train(X\_train, y\_train)`: Trains the Logistic Regression model on the provided training data.

- `predict(X\_test)`: Makes predictions on the test data using the trained model.

3. \*\*EmailClassifier Class\*\*:

- This class orchestrates the entire email classification process.

- It initializes with the path to the data file and creates instances of NaiveBayesModel and LogisticRegressionModel.

- Methods:

- `read\_data()`: Reads email data from a CSV file specified by the data path.

- `clean\_data()`: Cleans the data by filling missing values and removing rows with missing text values.

- `train\_models()`: Trains both Naive Bayes and Logistic Regression models on the cleaned data and prints confusion matrices for each model.

- `predict\_single\_text(text)`: Predicts the class labels (spam or non-spam) for a single input text using both models.

4. \*\*Main Execution Block\*\*:

- In the main block, the EmailClassifier is instantiated, data is read and cleaned, models are trained, and predictions are made for a user-provided text input.

## Execution:

To execute the code:

1. Ensure that the required dependencies (pandas and scikit-learn) are installed.

2. Specify the path to the email data CSV file.

3. Run the script, and it will train the models, print their performance metrics, and prompt the user to enter a text for prediction.

## Conclusion:

The Email Classifier code demonstrates the implementation of two popular machine learning models for email classification. By leveraging Naive Bayes and Logistic Regression algorithms, the code effectively categorizes emails into spam and non-spam categories, showcasing the potential of machine learning in email filtering applications.

## توثيق الكود الخاص بتصنيف البريد الإلكتروني

### المقدمة:

يعتبر كود تصنيف البريد الإلكتروني برنامجًا مكتوبًا بلغة Python يهدف إلى تصنيف الرسائل الإلكترونية إلى فئتين: الرسائل المزعجة والرسائل غير المزعجة باستخدام نموذجين في تعلم الآلة: نموذج نايف بايز ونموذج الانحدار اللوجستي. يهدف هذا الوثائق إلى توثيق الكود بشكل مفصل لفهم كيفية عمله والاعتمادات المستخدمة.

### الاعتمادات:

- pandas: لتلاعب البيانات وقراءة ملفات CSV.

- scikit-learn: لتنفيذ خوارزميات تعلم الآلة وتقييم أدائها.

### بنية الكود:

يتكون الكود من الأقسام التالية:

1. \*\*نموذج NaiveBayesModel\*\*:

- يعرف هذا النموذج نموذج نايف بايز المستخدم لتصنيف البريد الإلكتروني.

- يبدأ بتهيئة نموذج Multinomial Naive Bayes وكائن CountVectorizer.

- الطرق:

- `train(X\_train, y\_train)`: يدرب النموذج نايف بايز على البيانات التدريبية المقدمة.

- `predict(X\_test)`: يقوم بعمل توقعات على بيانات الاختبار باستخدام النموذج المدرب.

2. \*\*نموذج LogisticRegressionModel\*\*:

- يعرف هذا النموذج نموذج الانحدار اللوجستي المستخدم لتصنيف البريد الإلكتروني.

- يبدأ بتهيئة نموذج الانحدار اللوجستي مع حد أقصى لعدد الدورات يساوي 1100 وكائن CountVectorizer.

- الطرق:

- `train(X\_train, y\_train)`: يدرب نموذج الانحدار اللوجستي على البيانات التدريبية المقدمة.

- `predict(X\_test)`: يقوم بعمل توقعات على بيانات الاختبار باستخدام النموذج المدرب.

3. \*\*فئة EmailClassifier\*\*:

- تدير هذه الفئة عملية تصنيف البريد الإلكتروني بأكملها.

- يتم تهيئتها مع مسار الملف الذي تحتوي عليه البيانات وإنشاء مثيلات لنموذج NaiveBayesModel و LogisticRegressionModel.

- الطرق:

- `read\_data()`: يقرأ البيانات البريد الإلكتروني من ملف CSV محدد بواسطة مسار البيانات.

- `clean\_data()`: يقوم بتنظيف البيانات عن طريق ملء القيم المفقودة وإزالة الصفوف ذات قيم نصية مفقودة.

- `train\_models()`: يقوم بتدريب نموذجي النايف بايز والانحدار اللوجستي على البيانات المنظفة وطباعة مصفوفة الارتباك لكل نموذج.

- `predict\_single\_text(text)`: يتنبأ بتسميات الفئة (مزعجة أو غير مزعجة) لنص واحد مدخل من قبل المستخدم باستخدام النموذجين.

4. \*\*الكتلة الرئيسية للتنفيذ\*\*:

- في الكتلة الرئيسية، يتم إنشاء مثيل لفئة EmailClassifier، ثم يتم قراءة البيانات وتنظيفها، وتدريب النماذج، وأخيرًا تقديم التنبؤات لنص مدخل من قبل المستخدم.

## التنفيذ:

لتنفيذ الكود:

1. تأكد من تثبيت الاعتماديات المطلوبة (pandas وscikit-learn).

2. حدد مسار ملف البيانات الخاص بالبريد الإلكتروني.

3. قم بتشغيل البرنامج، وسيتم تدريب النماذج، وطباعة مقاييس أدائها، ومن ثم يطلب من المستخدم إدخال نص للتنبؤ به.

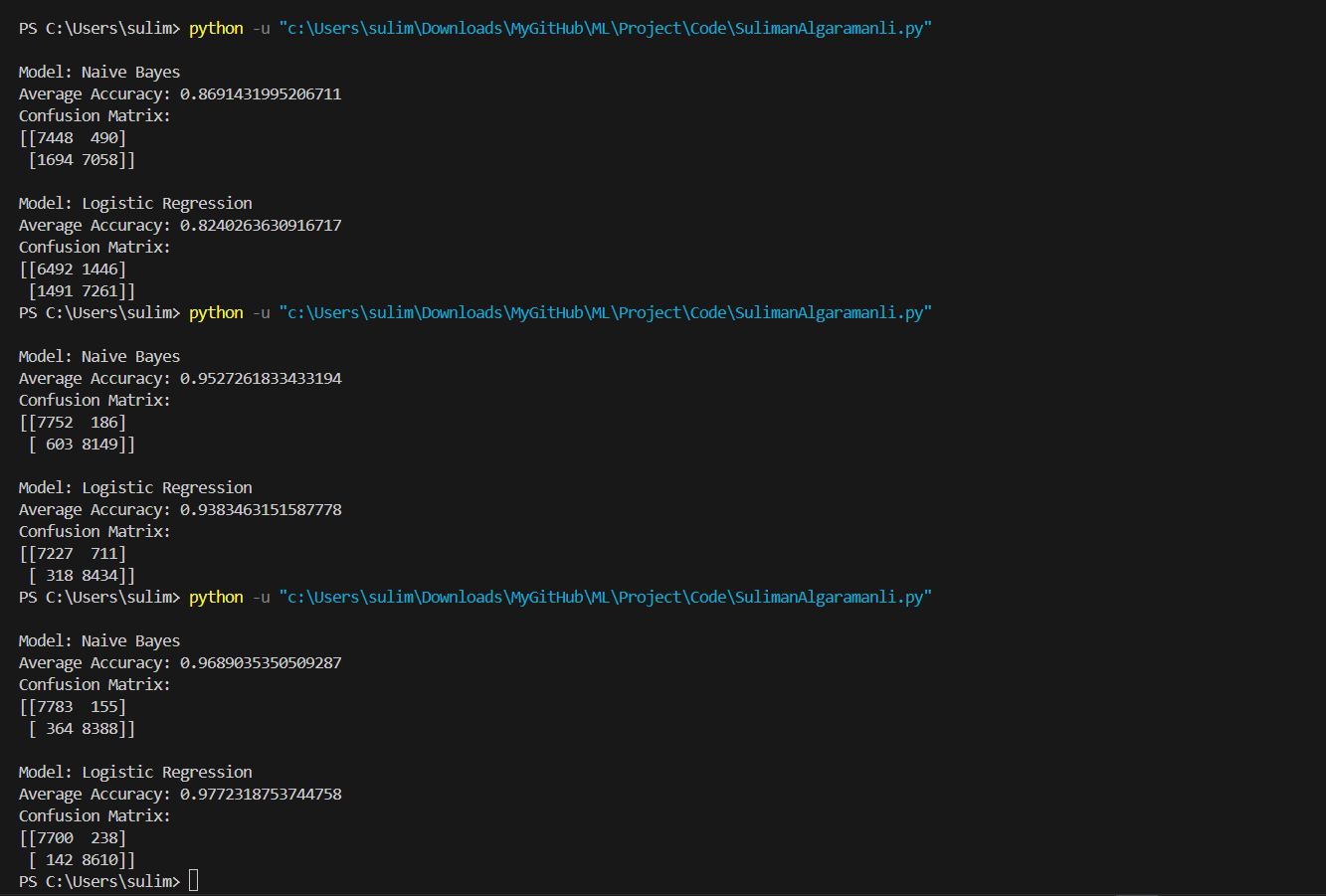
## الاستنتاج:

يوضح كود تصنيف البريد الإلكتروني تنفيذ نموذجين شهيرين في تعلم الآلة لتصنيف الرسائل الإلكترونية. من خلال استخدام خوارزميات نايف بايز والانحدار اللوجستي، يقوم الكود بفعالية بتصنيف الرسائل الإلكترونية إلى فئتين: الرسائل المزعجة والرسائل غير المزعجة، مما يبرز إمكانيات تعلم الآلة في تطبيقات تصفية البريد الإلكتروني.

ملاحظات:

عند استعمال جزء قليل من الداتا ست تقوم Naïve Bayes بالتعلم بدقة اكتر من Logistic Regression.

وهذا بسبب ان Naïve Bayes سريعة التعلم.

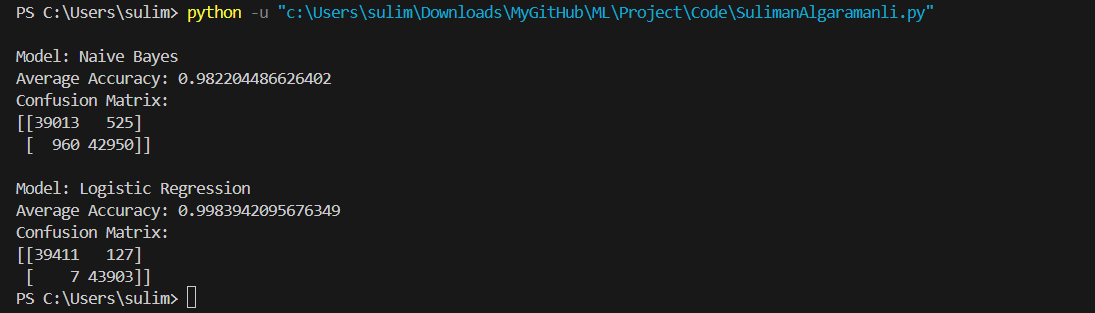


صف فقط. 10,000عند استعمال

عند استعمال 100 صف فقط.

صف فقط. 1,000عند استعمال

ولكن عند استعمال الداتا كاملة,000 80 سطر صف تقريبا. تكون نتيجة الدقة متقاربة وغالبا ما تتفوق Logistic Regression.



صف فقط. 80,000عند استعمال