تصنيف الرسائل الإلكترونية للكشف عن البريد المزعج

- **مقدمة:**

مع الاستخدام الواسع النطاق للبريد الإلكتروني في كل من التواصل الشخصي والمهني، أصبحت مشكلة رسائل البريد المزعج مصدر قلق كبير. فالرسائل الغير مرغوب فيها لا تعبّر فقط عن البريد العشوائي بل تشكل أيضًا مخاطر أمنية وتهديدات للخصوصية للمستخدمين. تعاني الطرق التقليدية لفرز البريد المزعج، من قيود في القابلية للتكيّف والفعالية. لذلك، اكتساب تقنيات التعلم الآلي لتصنيف البريد الإلكتروني قد اكتسب أهمية بسبب قدرتها الكبيرة على تحسين الدقة والكفاءة في اكتشاف البريد المزعج.

في هذا التقرير، سنقوم بدراسة استخدام النماذج الإحصائية Naive Bayes وLogistic Regression لتصنيف البريد الإلكتروني إلى فئات "spam" و "ham" (غير مؤذي). سنقوم بتحليل أداء النماذج وفقًا لمعايير مثل الدقة ومصفوفة الارتباك.

- **الطريقة:**

في هذا المشروع، نهدف إلى تطوير برنامج لتصنيف الرسائل الإلكترونية إلى فئتين: البريد المزعج والشرعي (البريد الحقيقي). تتضمن الطريقة المقترحة الخطوات التالية:

1. استيراد المكتبات اللازمة.
2. يتم تعريف نموذجين للتصنيف:
   - الأول يستخدم Naive Bayes.
   - الثاني يستخدم Logistic Regression.
3. تعريف الكلاس الرئيسي :EmailClassifier يقوم بقراءة البيانات، تنظيفها، تدريب النماذج، وطباعة النتائج.
4. قراءة وتنظيف البيانات: يتم قراءة البيانات من ملف CSV ومعالجتها لإزالة القيم المفقودة.
5. تدريب النماذج: يتم تدريب النماذج باستخدام تقنية.K-Fold Cross Validation
6. تقييم الأداء: يتم قياس الدقة ومصفوفة الارتباك لكل نموذج.

- **التجربة:**

في نصف المشروع، قمنا بإكمال تجهيز البيانات وتدريب النموذج بنجاح. تم تنظيف المجموعة البيانات، واستخراج الميزات من النصوص البريدية. أظهرت التجارب الأولية مع خوارزمية Naive Bayes نتائج واعدة من حيث الدقة والكفاءة. ومع ذلك، يتطلب الأمر إجراء تجارب وتقييم إضافي لضبط النموذج وتحسين أدائه.

- **الاستنتاج:**

في الختام، فإن تطوير برنامج لتصنيف الرسائل الإلكترونية إلى فئتي البريد المزعج والشرعي باستخدام تقنيات التعلم الآلي له القدرة على تعزيز أمان البريد الإلكتروني وتجربة المستخدم. من خلال تجهيز البيانات بشكل منهجي، واختيار الميزات المناسبة، وتدريب النماذج، نهدف إلى إنشاء نظام تصنيف فعال ودقيق. ستسهم التجارب المستمرة والتعديلات في تحسين فعالية الطريقة المقترحة في مكافحة رسائل البريد المزعج.

- **المراجع:**

[1] مجموعة بيانات .Email Spam Classification Dataset متوفر على Kaggle:

https://www.kaggle.com/datasets/purusinghvi/email-spam-classification-dataset

البرنامج:

```python
import pandas as pd
from sklearn.model_selection import KFold
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Naive Bayes تعريف النموذج المتعلق بـ
class NaiveBayesModel:
    def __init__(self):
        def __init__(self):
         self.model = MultinomialNB()  # استخدام نموذج Naive Bayes
         self.vectorizer = CountVectorizer()  # استخدام تقنية Bag-of-Words

    def train(self, X_train, y_train):{}
        # تحويل النصوص إلى تمثيل رقمي
         # تدريب النموذج

    def predict(self, X_test):{}
        # تحويل النصوص إلى تمثيل رقمي
        # التنبؤ بالتصنيف

# Logistic Regression تعريف النموذج المتعلق بـ
class LogisticRegressionModel:
    def __init__(self):
        self.model = LogisticRegression(max_iter=1000)  # استخدام نموذج Logistic Regression مع زيادة عدد الدورات
        self.vectorizer = CountVectorizer()  # استخدام تقنية Bag-of-Words

    def train(self, X_train, y_train):{}
        # تدريب


    def predict(self, X_test):{}
        # تبؤ

```

```python


# تعريف الكلاس الرئيسي لتصنيف البريد الإلكتروني
class EmailClassifier:
    def __init__(self, data_path):
        self.data_path = data_path
        self.data = None
        self.models = {'Naive Bayes': NaiveBayesModel(), 'Logistic Regression': LogisticRegressionModel()}

    def read_data(self):
        self.data = pd.read_csv(self.data_path)  # قراءة البيانات من المسار المحدد

    def clean_data(self):
        self.data = self.data.fillna('')  # ملء القيم المفقودة بقيم فارغة
        self.data = self.data[self.data['text'].notna()]  # إزالة الصفوف التي تحتوي على قيمة مفقودة في العمود 'text'

    def train_models(self):{}
        #class تقوم بتدريب النماذج الموجودة في ال
    def print_results(self, results):{}
         # طباعة متوسط مصفوفات الارتباك لكل نموذج

if __name__ == "__main__":
    data_path = "C:\\Users\\sulim\\Downloads\\DataSets_for_Spam-Emails\\combined_data.csv"  # مسار ملف البيانات
    classifier = EmailClassifier(data_path)
    classifier.read_data()  # قراءة البيانات
    classifier.clean_data()  # تنظيف البيانات
    results = classifier.train_models()  # تدريب النماذج
    classifier.print_results(results)  # طباعة النتائج
```