SMS Spam Detection Model

Students

- Hussein AbdElkader
- Ahmed Hesham
- Elsherif Shaban

Overview

This repository houses a machine learning model designed to detect spam messages in SMS (Short Message Service) data. The model is built using the ID3 algorithm and implemented using the scikit-learn library.

Spam SMS Dataset

- **Description:** Classifies SMS messages as spam or ham.
- **Records:** 5573
- **Target variable:** Spam classification (spam or ham)
- **Python libraries:** `pandas`, `scikit-learn`

Getting Started

Prerequisites

- Python 3
- Libraries: pandas, scikit-learn

Install the required libraries using:

```
pip install pandas scikit-learn
```

Usage

1. Clone this repository:

```
git clone [repository_url]
cd spam-detection-model
```

2. Download the SMS Spam Collection Dataset (e.g., 'spam.csv').

3. Run the model:

```
python main.py
```

4. Explore the results in the console. The accuracy and classification report will be displayed.

Files and Directory Structure

- `main.py` : Main script containing the implementation of the ID3 algorithm and model evaluation.
- `spam.csv` : SMS Spam Collection Dataset (not included, download and place in the same directory).

- `README.md` : Documentation file.

Model Details

- **ID3 Algorithm**: The model uses the Iterative Dichotomiser 3 (ID3) algorithm for decision tree-based classification.
- **Feature Extraction**: Text data is transformed using the CountVectorizer to convert messages into a format suitable for machine learning.
- **Training and Evaluation**: The model is trained on a subset of the dataset, and its performance is evaluated on another subset.

Results

- **Accuracy**: The accuracy of the model on the test set.
- **Classification Report**:

Class	Precision	Recall	F1-Score	Support
ham	0.98	0.99	0.98	965
spam	0.91	0.87	0.89	150
Accuracy			0.97	1115
Macro Avg	0.95	0.93	0.94	1115
Weighted Avg	0.97	0.97	0.97	1115

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier, plot_tree
import graphviz
from sklearn.tree import export_graphviz
from sklearn.tree import export_text

# Step 1: Load the dataset
df = pd.read_csv('spam.csv', encoding='latin-1')

# Step 2: Check the Loaded dataset
print(df.head())
print("Columns:", df.columns)
print("Missing values:\n", df.isnull().sum())
print("Class distribution:\n", df['v1'].value_counts())

# Step 3: Preprocess the Data
df = df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis=1)
```

```
# Step 4: Feature Extraction
# Using CountVectorizer to convert text data to a format suitable for machine
learning
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(df['v2'])

# Step 5: Split the Data
X_train, X_test, y_train, y_test = train_test_split(X, df['v1'], test_size=0.2,
random_state=42)

# Step 6: Implement the ID3 Algorithm
# Step 7: Train the Model
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)

# Step 8: Evaluate the Model
y_pred = clf.predict(X_test)

# Corrected part: Use vectorizer.get_feature_names_out() for feature names
plt.figure(figsize=(18, 12))
plot_tree(clf, filled=True, feature_names=vectorizer.get_feature_names_out(),
class_names=['non-spam', 'spam'], rounded=True)

# Export the decision tree to a Graphviz file
dot_data = export_graphviz(clf, out_file=None,
                           feature_names=vectorizer.get_feature_names_out(),
                           class_names=['non-spam', 'spam'],
                           filled=True, rounded=True, special_characters=True)

# Visualize the Graphviz file using the graphviz library
graph = graphviz.Source(dot_data)
graph.render("spam_decision_tree", format="png")
graph.view("spam_decision_tree")

# Metrics
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```