ITCS 6150/8150 Intelligent Systems

Final Project Report:

SMS Spam Detection using Support Vector Machines (SVM) for Binary Classification.

Submitted by:

Nishan Dhakal

and

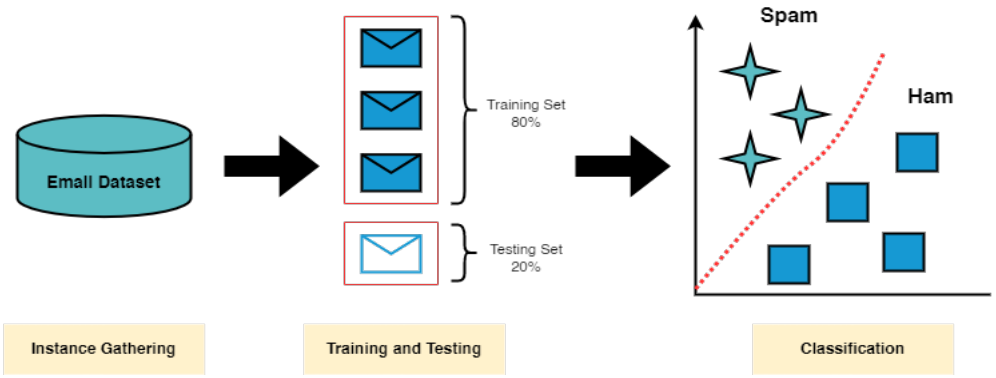
Afif Mujibur-Rahman

Abstract

Spam emails are becoming more and more common, which not only makes digital communication less efficient but also presents serious security issues. This study uses Support Vector Machine (SVM) methods to construct and assess a viable spam detection model. Using the SMS Spam Collection dataset from the UCI Machine Learning Repository, the work aims to achieve high levels of recall, accuracy, and precision in spam categorization.

Introduction

The primary objective for this project is to use Support Vector Machines (SVM) for binary classification in order to develop an SMS spam detection system. Due to its strong performance in binary classification tasks and its efficiency in managing high-dimensional data, SVM is an especially well-suited supervised machine learning technique for this task. This project is essential to improving the user experience by removing unsolicited and maybe hazardous messages from users' inboxes. The detailed report that follows explains the intricate procedure of creating a Python SMS Spam Detection model includes all of the necessary processes, including loading datasets, preprocessing, training, evaluating, and saving the model for use in future predictions.



Goals/Problem Statement

(I) Create an SMS spam detection system that can differentiate between messages that are classified as "spam" and those that are classified as "ham" from a dataset of SMS texts. **(II)** Process and prepare text data ready for modeling, which includes cleaning and preprocessing of SMS message data so that it can be used for feature extraction. **(III)** Train a machine learning model,

such as Support Vector Machine (SVM), on the preprocessed text data and evaluate its performance in correctly classifying messages as 'ham' or 'spam'. **(IV)** Accomplish high classification accuracy to effectively identify spam and non-spam SMS messages, minimizing both false positives and false negatives.

Tools and Dataset Used

Our models were trained and tested using data from the "SMS Spam Collection" dataset at the UCI Machine Learning Repository. This dataset offers a complete set of attributes derived from emails and is commonly used for benchmarking spam detection systems. The UCI spam collection dataset has two categories "ham" (non-spam) and "spam" as well as text explanations for each.

```
[ ] # Step 2: Convert text labels to numeric labels
    label_encoder = LabelEncoder()
    sms_df['label'] = label_encoder.fit_transform(sms_df[0])
```

```
sms_df.head()
```

	0	1	label
0	ham	Go until jurong point, crazy.. Available only ...	0
1	ham	Ok lar... Joking wif u oni...	0
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	ham	U dun say so early hor... U c already then say...	0
4	ham	Nah I don't think he goes to usf, he lives aro...	0

Because of its large ecosystem of data science libraries, Python was the preferred programming language used in google collab. In particular, Scikit-Learn, which offers tools for data splitting, preprocessing, model training, and assessment, was used to develop SVM.

Support Vector Machine (SVM)

SVM was selected due of its effectiveness in tasks involving binary classification. In order to function, the method builds a hyperplane in a high-dimensional space that best divides the data points of various classes. The SVM algorithm plots data items as points in n-dimensional space and differentiates the two classes by finding the hyperplane that separates them effectively. In the specific context of spam email classification, SVMs have been used to recognize patterns in email text and predict whether an email is spam or genuine. The use of the algorithm in

spam email detection is illustrated through the application of CountVectorizer, preparing the text data for classification.

Use of CountVectorizer and TF-IDF Transformer

When it comes to transforming unprocessed text data into a format that machine learning algorithms can use, **CountVectorizer** is important. CountVectorizer is significant in the area of spam detection mainly because it makes textual email data convertible to a numerical representation that machine learning models may use for categorization.

To convert count-based features obtained from CountVectorizer into normalized term frequency-inverse document frequency (TF-IDF) features for spam detection, using the **TfidfTransformer** is important.

```
# Step 5: Text vectorization
vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)

tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_vectorized)
```

Detail explanation of above code snippets:

The CountVectorizer class from the scikit-learn library is utilized to convert the raw text data (X_train) into a matrix of token counts.

The output of the CountVectorizer transformation is stored in the variable X_train_vectorized, which represents the matrix of token counts obtained from the training data. This numerical representation of the text data provides the foundation for subsequent processing and analysis.

The TfidfTransformer class is employed in the next step to normalize the count matrix obtained from the previous stage using the term frequency-inverse document frequency (TF-IDF) transformation. This transformation adjusts the raw word counts by considering the frequency of each word in the entire corpus.

The output of the TfidfTransformer transformation is stored in the variable X_train_tfidf, representing the TF-IDF weighted matrix derived from the original text data.

Limitations and Challenges

Even though the Support Vector Machine (SVM) model has a lot of potential for spam detection, there are a number of issues and restrictions that could affect how well it works and how widely it can be used. For example:

1. Spammers are always coming up with new strategies to get beyond traditional filtering techniques. If SVM models based on historical data are not updated on a regular basis, they may not perform well against newly developed spam methods.

2. Spam detection systems are anticipated to function in real-time in realistic applications. Due to its processing costs during the classification phase, SVM especially with complicated kernels may not be able to meet the latency criteria for real-time classification.

Classification Report and Confusion Matrix

```
# Step 7: Evaluate the model
# Import the necessary modules
from sklearn.metrics import classification_report
X_test_vectorized = vectorizer.transform(X_test)
X_test_tfidf = tfidf_transformer.transform(X_test_vectorized)

y_pred = model.predict(X_test_tfidf)

print(f"Accuracy: {accuracy_score(y_test, y_pred)}")
print(classification_report(y_test, y_pred))

# Creating the Confusion matrix
print(confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.9910313901345291
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       966
           1       1.00      0.93      0.97       149

    accuracy                           0.99      1115
   macro avg       0.99      0.97      0.98      1115
weighted avg       0.99      0.99      0.99      1115

[[966   0]
 [ 10 139]]
```

True Negatives (TN): 966 emails correctly identified as non-spam.

False Positives (FP): 0 emails incorrectly identified as spam (there are no false positives, which is ideal).

False Negatives (FN): 10 spam emails incorrectly identified as non-spam.

True Positives (TP): 139 emails correctly identified as spam.

Classification Report

Detail metrics explanation of above-mentioned classification report:

- **Precision:**
  - Non-Spam (0): 0.99 - This means 99% of emails predicted as non-spam were actually non-spam.
  - Spam (1): 1.00 - This means 100% of emails predicted as spam were actually spam.
- **Recall:**
  - Non-Spam (0): 1.00 - This means the model successfully identified 100% of all non-spam emails.
  - Spam (1): 0.93 - This means the model successfully identified 93% of all actual spam emails.
- **F1-Score:**
  - Non-Spam (0): 0.99 - A high F1 score for non-spam emails shows a strong balance between precision and recall.
  - Spam (1): 0.97 - Even with the high level of accuracy, the lower recall slightly reduces the F1 score, reflecting the 10 missed spam emails.
- **Support:**
  - Non-Spam (0): 966 - The number of actual non-spam emails in dataset.
  - Spam (1): 149 - The number of actual spam emails in dataset.

Overall Accuracy

99.1% - The model's overall accuracy, meaning that 99.1% of emails were accurately categorized. This impressive accuracy rate suggests that the model is quite effective for the dataset that was used.

Conclusion

With an overall accuracy of 99.1%, the SVM-based spam detection model showed exceptional performance, which is noteworthy for a binary classification problem in the context of natural language processing. For the purpose of protecting user confidence and the integrity of the email system, the model

demonstrated remarkable accuracy, with no valid emails being mistakenly classified as spam. When it came to spam emails, the precision was perfect (100%), meaning that every email that was flagged as spam was actually spam.

In summary, the research was successful in creating a very accurate and dependable SVM-based spam detection system. The knowledge acquired and the outcomes attained provide a strong basis for additional study and real-world implementation of spam detection algorithms. To guarantee that the model continues to be effective against the constantly changing landscape of email spam, future research should concentrate on improving the model's adaptability and minimizing even the smallest errors.

Sources

- SMS Spam Collection Dataset - UCI Machine Learning Repository
https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection

- Shrawan Kumar Trivedi - BML Munjal University, Gurgaon, Haryana, India
A study of machine learning classifiers for spam detection
https://ieeexplore.ieee.org/abstract/document/7743279/authors#authors

- Olubodunde Agboola, Vaidyanathan, Ramachandran, Knapp, Gerald M., Wei, Shuangqing, and Wu, Jianan. 2022.

Spam Detection Using Machine Learning and Deep Learning. Ph.D. Dissertation.

https://dl.acm.org/doi/book/10.5555/aai30276699