



Fake Review Recognition using an SVM Model

Alexander Iliev^{1,2}(✉), Anandreddy Nimmala¹, Rashid Abdul Rahiman¹, Sonia Raju¹,
Swetha Chilakalanerpu¹

¹SRH University of Applied Sciences, Berlin, Germany

²Bulgarian Academy of Sciences, Sofia, Bulgaria

ailiev@berkeley.edu

Anandreddy.Nimmala@stud.srh-campus-berlin.de;

RashidAbdulRahiman@stud.srh-campus-berlin.de;

Sonia.Raju@stud.srh-campus-berlin.de;

Swetha.Chilakalanerpu@stud.srh-campus-berlin.de

Abstract. This paper aims to develop a fake product review recognition system on Amazon using Python code. E-commerce websites form an ever-expanding portion of the global business market, and reviews play a vital role in deciding the authenticity and quality of a product. Reviews and comments play a vital role in the e-commerce domain contributing to a product's and in turn a seller's good or bad reputation. There are often fake reviews posted on these platforms which ultimately result in either raising or dropping the ratings of these products. To promote products and disparage rivals, immoral actors hire bot farms, purchase legitimate accounts, and use Sybil accounts. Many businesses employ people to post fictitious favourable reviews about their goods or services or unfairly critical reviews about the goods or services of their rivals. This procedure provides inaccurate information to potential buyers of these goods; hence we need a way to identify and eliminate bogus reviews. Moreover, it directly affects the reliability of both the seller as well as the platform. Through this project, we aim to utilize SVM techniques in Python script to differentiate fake reviews from authentic ones, which ultimately decide the quality of the product and reliability of the seller.

Keywords: Natural Language Processing, Fake Review Recognition, SVM model

1 Introduction

Product reviews on e-commerce websites play a vital role in determining the quality and authenticity of both a product and its seller. For this reason, there are often fake reviews posted on these platforms which ultimately result in a hike in the ratings of the products and in turn, the sales revenue of the business. Alternatively, there also exists fake negative reviews which can be used to miscredit the opposition organisations in a specific business sector. For these reasons, it is crucial to understand which reviews are fake and which real and filter them accordingly. This project uses an SVM model to determine whether a review posted on the Amazon website is fake or real.

The upcoming section talks about the machine learning algorithm as well as the python packages used, in order to better understand what is happening with the computer code for fake review detection.

1.1 SVM Model

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. It is mostly used in classification problems [1]. In this project, we use an SVM Model in Python code which classifies the reviews as being real or fake. Using an existing review text dataset, we build a model that predicts the authenticity of reviews without the help of any deep learning techniques. The concept of support vector machine (SVM) methodology is transferring the vectors to a higher dimension [2]. The optimal linear hyperplane could be obtained from the largest distances between the 2 categories [3]. A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labelled training data for each category, they're able to categorise new text [4].

Computing the SVM Classifier amounts to minimizing an expression of the form

$$\left[\frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i(w^T x_i - b)) \right] + \lambda ||w||^2 \quad (1)$$

Where n is the number of points, y_i is the i -th target, $w^T x_i - b$ is the i -th output. The parameter λ determines the trade-off between increasing the margin size and ensuring that the x_i lie on the correct side of the margin.

1.2 SVM Model Advantages and Disadvantages

SVM models are effective in high dimensional spaces. The structure of an SVM model is such that they remain effective in cases where the number of dimensions is greater than the number of samples [5]. They are highly memory efficient, being a result of using a subset of training points in the decision function called support vectors [6].

SVM models do not directly provide probability estimates, as these are calculated using an expensive five-fold cross-validation [6].

1.3 Natural Language Toolkit (NLTK)

The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP) [4]. It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning. It helps the computer to analysis, pre-process, and understand the written text [5]. It is one of the most powerful NLP libraries, which contains packages to make machines understand human language and reply to it with an appropriate response [6]. NLTK is also used for sentiment analysis to check if a product review is positive or negative [6].

1.4 NLTK Advantages and Disadvantages

NLTK is the most well-known and full NLP library with many 3rd extensions [2]. It supports the largest number of languages compared to other libraries. NLTK however, is slow and difficult to learn and use. It has no neural network models and only splits text by sentences without analysing the semantic structure [6].

1.5 Scikit Learn (Sklearn)

Sklearn is a library in Python that provides many unsupervised and supervised learning algorithms [4]. It's built upon some of the technology which is familiar with like NumPy and Matplotlib. It is the most useful and robust library for machine learning in Python [6]. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction via a consistency interface in Python [2].

1.6 Scikit Learn Advantages and Disadvantages

The library is distributed under the BSD license, making it free with minimum legal and licensing restrictions.

It is easy to use. The scikit-learn library is very versatile and handy and serves real-world purposes like the prediction of consumer behaviour, the creation of neuroimages, etc. Scikit-learn is backed and updated by numerous authors, contributors, and a vast international online community. The scikit-learn website provides elaborate API documentation for users who want to integrate the algorithms with their platforms.

1.7 Data Flow Diagram

The first phase of our fake review recognition is the initial research and survey, which is the primary phase of the data acquisition process. Our reviews are collected and prepared for the pre-processing process.

Data acquisition is where we collect the unformatted raw data from our specific source or sources. This is where we start building the dataset upon which our model is completely based upon. The acquisition is a raw process which is then further filtered down and structured in the following procedures starting with data pre-processing.

Data Integration and Spam Identification Labeling are where we both combine multiple review sources into one, as well as look for various key identifiers in this integrated dataset.

Data pre-processing is where the data collected is converted into a clean data set. This is where we fix punctuation, cases, letters, and typographic errors, in order to keep all the reviews on a level playing field. Preprocessing also includes tokenization, parsing, semantic reasoning and stemming, and these are carried out with the Natural Language Toolkit.

The model then moves into the processing stage where we use the Sklearn package to successfully label between genuine and fake reviews. Training the dataset is done with

tuning done on the parameters using the validation set. Finally, the model moves into the evaluation phase, where we determine whether the model can successfully determine if a review is genuine or fake. The model is graphically represented in figure 1:

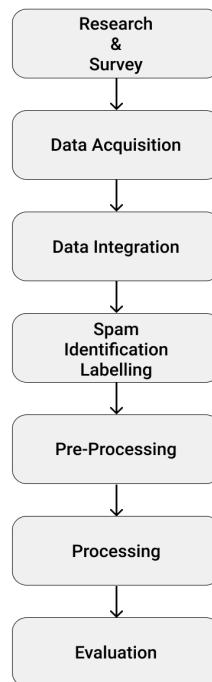


Figure 1: Data Flow Diagram of Product Review Recognition

1.8 Use Case Diagram

The integrity of a review is often crucial to determine not just the quality of the product, but also the seller as well. There exists a delicate balance between a good review and a fake review determining whether both the seller and the product is trustworthy for the customer.

Review extraction is where we can use the model to extract specific reviews (that are not fake) focusing on a feature, rather than giving an overall opinion. This use case is helpful in cases where we can also move into a filtering option to categorize reviews based on a specific feature or product.

Spam filtering is the primary use case in this model, where we differentiate fake reviews from authentic ones, which ultimately decide the quality of the product and reliability of the seller.

The product page is the key landing position on the website of any seller. There is an incredibly increased chance of a customer going through with the purchase whenever there is a reliable positive review on the product page.

All of these steps are graphically depicted in figure 2:

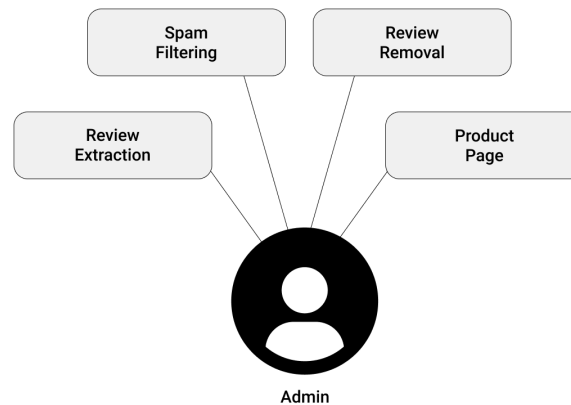


Figure 2: Use Case Diagram of Product Review Recognition

1.9 Dataset Used

A recently released corpus of Amazon reviews containing 21000 text reviews of products. This corpus was used to pre-process the data and implement cross-validation.

2 RELATED WORK

Fake review detection problem is experimentally being worked on since 2007 [6]. Textual and behavioral elements have been extensively used in the research on the detection of fake reviews. Textual features refer to a review activity's language characteristics. In other words, textual features mostly depend on the reviews' content. The nonverbal qualities of the reviews are referred to as their behavioral aspects. They largely depend on the reviewers' actions, including their writing style, facial expressions, and frequency of review writing. While addressing textual features is difficult and necessary, behavioral features are also highly significant and cannot be disregarded because they have a significant impact on how well the fake review detection process works. For the purpose of detecting fake reviews, Elmurngi and Gherbi [7] employed supervised machine learning techniques. Five classifiers—SVM, Naive-bayes, KNN, k-star, and decision tree—are employed. SVM and Naive base classifiers were both employed by Shojaee et al. [8]. The authors used the yield dataset, which has 1600 reviews gathered from 20 well-known Chicago hotels. To identify false opinion spam, Ren and Ji [9] employed neural and discrete models with average, CNN, RNN, GRNN, average GRNN, and bi-directional average GRNN classifiers. These were research work focused on the textual features and not the behavioral features, which we aim to emulate in this study.

3 COMPUTER CODE

The primary libraries used in this program are Sklearn and NLTK in addition to the usual necessities in NumPy. SVM Model is used to classify the reviews as authentic.

```

import csv
from sklearn.svm import LinearSVC
from nltk.classify import SklearnClassifier
from random import shuffle
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import precision_recall_fscore_support
from sklearn.metrics import accuracy_score
import numpy as np
import nltk
from nltk.tokenize import word_tokenize
nltk.download('punkt')

rawData = []
preprocessedData = []
trainData = []
testData = []
fakeLabel = 'fake'
realLabel = 'real'
reviewPath = 'amazon_reviews.txt'

```

Here, 'rawData' is the filtered data from the dataset file that is 21000 samples. 'preprocessedData' is the preprocessed reviews.

```

print("Now %d rawData, %d trainData, %d testData" % (len(raw-
Data), len(trainData), len(testData)),
      "Preparing the dataset...",sep='\n')
loadData(reviewPath)

```

This is to parse the dataset and put it in a raw data list. This is done to ensure that the data we are working with is all in the same format.

```

print("Now %d rawData, %d trainData, %d testData" % (len(raw-
Data), len(trainData), len(testData)),
      "Preparing training and test data...",sep='\n')
splitData(0.8)

```

This is to split the raw dataset into a set of training data and a set of test data in a 80-20 percentage division.

```

print("Now %d rawData, %d trainData, %d testData" % (len(raw-
Data), len(trainData), len(testData)),
      "Training Samples: ", len(trainData), "Fea-
tures: ", len(featureDict),sep='\n')
print("Mean of cross-validations (Accuracy, Precision, Re-
call, Fscore): ", crossValidate(trainData, 10))

```

This is to print the number of training samples and the number of features, providing us with more information on what we are working with on the execution of this model.

```

table = str.maketrans({key: None for key in string.punctuation})
def preprocess(text):
    # Should return a list of tokens
    lemmatizer = WordNetLemmatizer()
    filtered_tokens=[]
    lemmatized_tokens = []
    stop_words = set(stopwords.words('english'))
    text = text.translate(table)
    for w in text.split(" "):
        if w not in stop_words:
            lemmatized_tokens.append(lemmatizer.lemmatize(w.lower()))
    filtered_tokens = [' '.join(l) for l in nltk.bi-
grams(lemmatized_tokens)] + lemmatized_tokens
    return filtered_tokens

```

In order to preprocess the text and vectorise their features, we use the above code.

```

featureDict = {}
def toFeatureVector(Rating, verified_Purchase, product_Cate-
gory, tokens):
    localDict = {}
    featureDict["R"] = 1
    localDict["R"] = Rating
    featureDict["VP"] = 1
    if verified_Purchase == "N":
        localDict["VP"] = 0
    else:
        localDict["VP"] = 1
    if product_Category not in featureDict:
        featureDict[product_Category] = 1
    else:
        featureDict[product_Category] = +1
    if product_Category not in localDict:
        localDict[product_Category] = 1
    else:
        localDict[product_Category] = +1

    for token in tokens:
        if token not in featureDict:
            featureDict[token] = 1
        else:
            featureDict[token] = +1
        if token not in localDict:
            localDict[token] = 1
        else:
            localDict[token] = +1
    return localDict

```

‘featureDict’ here enables us to have a global dictionary of features that help us categorise a review’s rating, whether a purchase is verified or not, the category of the product and whether the review is a text.

```
classifier = trainClassifier(trainData)
predictions = predictLabels(testData, classifier)
true_labels = list(map(lambda d: d, testData))
a = accuracy_score(true_labels, predictions)
p, r, f1, _ = precision_recall_fscore_support(true_labels, predictions, average='macro')
print("accuracy: ", a)
print("Precision: ", p)
print("Recall: ", a)
print("f1-score: ", f1)
```

Finally, we can understand the accuracy and prediction of our model by using the test data of 20% against the trained data of 80% to recognise whether a review is fake or real. With the execution of our code, we achieved an accuracy of 80.42% with a precision of 80.80%.

4 Conclusions

Detection of fake reviews on an e-seller website is a time consuming and labouring task when done individually. It is extremely important to have an efficient machine learning system that automatically does the classification and filtering of reviews in a largely electronic shopping adaptive world today, especially with the influx of these numbers due to the COVID-19 pandemic. Using the SVM model in python, we achieved an accuracy of 80.42%, with a precision of 80.80%. The model was able to successfully differentiate fake reviews from authentic ones using the cross-validation techniques learnt from the corpus of reviews. This paves the way for this model to be implemented on a large scale with e-commerce websites, with multiple use cases on the agenda. Fake review recognition enables the user and customer to buy genuine and quality products from the online market, and solves the problem of misdirection, misinformation, and misconduct on online platforms.

References

- [1] Jain P., Chheda K., Jain M., Lade P. (2019) Fake Product Review Monitoring System, International Journal of Trend in Scientific Research and Development (IJTSRD), ISSN:2456-6470, Volume-3|Issue-3, pp.105-107
- [2] Anil A. (2020) Interface for Fake Product Review Monitoring and Removal, International Research Journal of Engineering and Technology (IRJET), ISSN:2395-0056, Volume-7|Issue-7
- [3] Sinha A., Arora N., Cheema M. (2018) Fake product review monitoring using opinion mining, International Journal of Pure and Applied Mathematics

- [4] Narayan R., Rout J., Jena S. (2018) Review Spam Detection using Unsupervised Technique, *Progress in Intelligent Computing Techniques: Theory, Practice and Applications*, pp.281-286
- [5] Etaowo W., Naymat G. (2017) The Impact of Applying Preprocessing steps on Review Spam Detection, *The 8th International Conference on Emerging Ubiquitous System and Pervasion Networks*, pp.273-279
- [6] Jindal N., Liu B. (2008) Opinion Spam and Analysis, *Proceedings of the International Conference on Web Search and Web Data Mining – WSDM 08 ACM*, pp. 219-230
- [7] E. Elmurngi and A. Gherbi, *Detecting Fake Reviews through Sentiment Analysis Using Machine Learning Techniques. IARIA/DATA ANALYTICS*, 2017
- [8] S. Shojaee et al., “Detecting deceptive reviews using lexical and syntactic features.” 2013.
- [9] Y. Ren and D. Ji, “Neural networks for deceptive opinion spam detection: An empirical study,” *Information Sciences*, vol. 385, pp. 213224, 2017.