A Comparative Study of Oversampling Techniques on Binary Classification for Detecting Duplicate Advertisement

MSc Research Project
Data Analytics

Smriti Verma
x17101603

School of Computing
National College of Ireland

Supervisor: Dympna O'Sullivan, Paul Styles, Pramod Pathak

National College of Ireland
Project Submission Sheet – 2017/2018
School of Computing

Student Name:	Smriti Verma
Student ID:	x17101603
Programme:	Data Analytics
Year:	2017
Module:	MSc Research Project
Lecturer:	Dympna O'Sullivan, Paul Styles, Pramod Pathak
Submission Due Date:	13/08/2018
Project Title:	A Comparative Study of Oversampling Techniques on Binary Classification for Detecting Duplicate Advertisement
Word Count:	7916

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	
Date:	5th September 2018

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
3. Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Comparative Study of Oversampling Techniques on Binary Classification for Detecting Duplicate Advertisement

Smriti Verma
x17101603
MSc Research Project in Data Analytics

5th September 2018

Abstract

The online marketplace has become a great platform for conducting business. Not only does it allow the users to find and buy desirable items easily, but also stages an area where the user can upload their refurbished products in search of a potential buyer. Due to ever increasing competition within the market, competitive sellers go to great lengths to ensure that their products are noticed. This results in sellers posting the same advertisement several times, using near-duplicate titles or using slightly altered descriptions.

This study proposes to build a dichotomous classifier that would spot such duplicate commercial advertisements that feature the same product. A Russian dataset of 3 million records was translated into English, for the better understanding of the results. The dataset was imbalanced with data samples for duplicate class less than the non-duplicate class.

This study compares the six oversampling techniques, Random oversampling, SMOTE, SMOTE-Borderline 1, SMOTE-Borderline 2, SVM SMOTE and ADASYN, used to achieve class balance in the dataset. Four classification models, Gradient Boosting Tree, Logistic Regression, Naive Bayes and SVM, are built, on top of the oversampling techniques, to identify the duplicate advertisements.

This study finds that the performance of classifiers improves with an increase in the sample size of the training data. The best performing model was SVM when paired with Borderline-SMOTE 2, with an F1 score of 0.9151

The proposed model will prevent the buyers from sifting through the dozens of deceptively identical advertisements, thereby expediting the search process. With more accurate duplicate ad detection, the model will enable the buyers to easily find a desirable product.

1 Introduction

The introduction of online shopping has redefined the way of conducting business. Shoen et al. (2012) describes online marketplace as a platform that provides services in the e-commerce marketplace to facilitate a transaction between a customer and a host. Although online marketplace is gaining popularity in the market as it allows a user to sell their product hassle-free, this dynamic medium of conducting a business is accompanied by a set of problems. The concerns about internet security and online fraud have a huge impact on the online business transactions (Smith et al.; 1999).

The increase in the competition of the online market place coerces the e-commerce websites to host high volumes of listings. Often, the competitive sellers go to great lengths to ensure that their products are noticed. These sellers post the same advertisement several times under different categories, using near-duplicate titles or using slightly altered descriptions. In the scenario of big data, the chances of presence of duplicate records becomes significant. A study by Wang et al. (2009) shows that more than 80 percentage of the news articles present on the web are not original and are just a near-duplicate of the original articles. These repetitive records hinder the user experience and results in an increase in the cost of data storage and maintenance. Hence, the duplicity of data hosted on the online marketplace is a concern shared by both, the consumers as well as the corporate business.

The aim of this research is to investigate to what extent duplicate advertisements in the online marketplace can be identified by using Machine Learning algorithms. The classification of advertisements into duplicates or non-duplicates can be considered as a text classification task, where the text-based fields such as title, description, category and numeric values of price, latitude, and longitude can be compared. According to the survey conducted by Allahyari et al. (2017) on text mining algorithms, Naive Bayes classifiers, decision tree classifiers, support vector machines (SVM) and Nearest Neighbour Classifier are some notable classifiers which are widely implemented for text classification.

Many classifiers rely on the class distribution for making predictions on data. If the data is not balanced across the different classes, it can interfere with the performance of the classification model. Class-imbalance is also closely related to cost-sensitive learning (Liu et al.; 2009). One of the major concerns when detecting duplicates is the imbalanced nature of the dataset. The likelihood of a pair of advertisements being duplicates is much less than the likelihood of them being non-duplicates. For a classification model, it is necessary for a classifier to learn from the data samples of both classes, if not the results of majority class will result in high accuracy. Hence, it is essential to ensure that there is a class balance in the data so that the classifiers do not give biased results.

The present study compares the various oversampling methods and classification algorithms on the duplicate ad dataset. The aim of this study is:

- To identify if balanced data provides better results than imbalanced data
- To observe how the classifiers behave for different oversampling techniques and examine which method provides better results
- To observe the impact of different samples of the dataset on the classifier.

The process flow of this study is threefolds:

- in the first step, similarity measures are used to identify duplicates when two records are compared;
- in the second step, six oversampling strategies are used on an imbalanced data set to attain balanced class distribution;
- in the third step, four classification models are implemented upon each of the re-distributed data sets

The rest of this paper is organized as follows: Section 2 reviews related methods. Section 3 discusses the methodology used for this study. Section 4 describes the implementation. Section 5 is the discussion on the results of the experiment. Finally, Section 6 concludes this paper.

2 Related Work

Naumann and Herschel (2010) define duplicate detection as the identification of more than one representation of same real-world objects. The duplicate records can be identical or non-identical. Identical records are the records which are exactly similar to each other whereas non-identical duplicates, also known as Fuzzy Duplicates, are records which have slight different representation of the same object.

Duplicate pairs can be identified by using a similarity measure function sim(c, c') that takes two parameters, compares them and returns a similarity score (Naumann and Herschel; 2010). A high similarity score returned by the function indicates that two compared parameters are highly similar. The two candidates are classified as duplicates if their similarity score is above a given threshold, and hence they form a duplicate pair.

2.1 Applications of duplicate detection

Numerous approaches have been suggested for identifying duplicates in news articles, literature works, patents, or finding repetitive parameters in databases or recognizing fake claims. In domains like, digital libraries, plagiarism detection, web search, spam email detection, it is important to identify near-duplicate or similar documents.

Kovacevic et al. (2010) proposes the use of the Jaro-Winkler algorithm to detect duplicate listings of titles of books in a digital library. Kondrak (2005) compares the performance of n-grams (bi and tri-) to the existing methods like LCSR and NED, concluding that a better precision of 0.841 can be achieved using bi-grams against the 0.798 in LCSR and 0.823 in NED.

Vaughan (2014) discusses the impact on the performance of a business when near-duplicate web pages are displayed on a search engine. Urvoy et al. (2008) discusses the application of similarity detection algorithms for recognizing a web scam, by identifying fake web pages that have a common look and feel to that of the original webpage, by comparing the similarity in the HTML source codes. Henzinger (2006) compares the shingling algorithm and the Charikar's hashing algorithm for the detection of duplicates in a set of 1.6 billion web pages. The results show that the hashing algorithm outperforms the shingling algorithm with a precision of 0.50 against 0.38.

MinHash is one of the widely used hashing algorithm for duplicate detection. Weissman et al. (2015) proposed the MapReduce implementation of Minhash for an XML dump of 10.2 million Wikipedia articles for identifying clusters of sentences with high Jaccard similarity. The set-up identified that nearly 45.3% articles present across the 16 Hadoop nodes were identical whereas 13.5% were near duplicates.

However, Hassanian-esfahani and Kargar (2018) argue that MinHash is not the appropriate technique to detect NDD articles, as it predicts the similarity of the two documents based on the attributes they possess, whereas for near-duplicate documents, it is essential to consider the position of the attributes. Instead of MinHash, they proposed using Sectional MinHash for the detection of the near-duplicate documents, which gave an F-measure of 87.05%.

A novel technique, SuperMinHash, has been introduced by Ertl (2017), a hashing algorithm that succeeds MinHash. SuperMinHash significantly improves the runtime for calculating signatures for large datasets. For instance, if the signature consists of m values, the complexity of Minhash is O(mn) whereas for SuperMinHash, the complexity follows as O(n +m log2 m). SuperMinHash considers that the signature values are not independent, and hence it generates random permutations for each of the input data elements.

Bilke and Naumann (2005) implemented an approach for identifying similar attributes in a database formed by the merging of various schemas. The DUMAS(Duplicate-based Matching of Schemas) approach compares a common attribute using extended tuple similarity measure etupsim(), in order to determine if a pair of tuples or rows, originating from different schemas, represents the same object. The results show that 95% of the records were matched correctly (recall) whereas the number of incorrect matches(precision) was 10%.

Phankokkruad (2017) suggests the use of longest common subsequence, Smith-Waterman, Euclidean distance and Damalau-Levenshtein distance for detecting plagiarism in the 6700 pairs of assignments submitted by 135 students. The Levenshtein Distance Algorithm has also been used by Gaikwad and Bogiri (2015) for the detection of duplicate XML documents, giving a better recall of 0.48 against that of NED algorithm (recall 0.37).

2.2 Duplicate ad detection

Burk et al. (2017) discusses a model, called Apollo, used to detect duplicate job advertisements in the online recruitment domain. The ad content from multiple sources was collected and reduced into concatenated keys consisting of a high-level occupation classification, company name and geographical location, using MapReduce jobs. These keys were then split into hashed shingle sets. Jaccard Similarity was used to compare the similarity between shingles from two job ads. After calculating the Jaccard similarity between two job ads, heuristic thresholds were used to determine if they are similar. While the traditional SimHash and Shingling techniques gave a precision of 0.27 and 0.33 respectively, the proposed model gave a precision of 0.36.

2.3 Oversampling techniques

In a real-world scenario, the identification of duplicates in large datasets is a challenge because datasets might not have many duplicate records. This results in class imbalance. An imbalanced class is where the data points of one class is way higher than the data points of the other classes. In cases such as identifying non-identical duplicates, the dataset is expected to have more records that hold information about the advertisements which are not duplicates than the duplicates itself. As suggested by He and Garcia (2009), the input that gets provided for the model needs to be altered or sampled in such a way that it provides a balanced distribution or near balanced distribution.

Weiss and Provost (2003) compared the performance of classifiers on data with balanced and natural class distribution. Using the C4.5 algorithm on 26 diverse data sets, it was concluded that a classifier that is trained on a balanced dataset performs better, even if the testing dataset is imbalanced. Thus, it was shown that class balance is important for the training stage of probabilistic machine learning algorithms.

Feng et al. (2018) categorized the approaches used for balancing datasets into two divisions: 1) the sampling approach 2) the algorithmic approach. Sampling approaches include undersampling and oversampling techniques. Undersampling is defined as a non-heuristic method which balances the class distribution by removing the samples from the majority class. Hence, a subset of the majority class is used to train the classifier (Liu et al.; 2009). Since many data points of the majority class are ignored, the training data set becomes balanced and the overall training process becomes faster. However, a major drawback of this technique is the loss of information caused due to the neglected data points (Liu et al.; 2009). Oversampling is defined as the systematically generated synthetic instances of the minority data points. These samples are synthesized by considering the class ratios of the surrounding nearest neighbours of the minority data point (Chen et al.; 2010). However, the replication of samples can result in overfitting of the classification model.

The algorithmic approach leverages a machine learning algorithm to sample the data. Lin et al. (2017) uses two undersampling techniques that employ clustering during the data preprocessing step. Forty-four small-scale and two large-scale data sets were tested against a C4.5 decision tree classifier and single multilayer perceptron classifier ensembles to deliver optimal performance. Lu et al. (2017) proposes a novel ensemble framework that combines the Ensemble of Undersampling (EUS) technique, Real Adaboost, and an adaptive boundary decision strategy to build a hybrid algorithm.

Random oversampling is a technique in which the samples from the minority class are selected randomly and replicated in the feature space until the number of minority class samples is approximatly equal to that of the majority class samples. This technique could lead to the overfitting of the model. To avoid overfitting, Synthetic Minority Oversampling TEchnique (SMOTE) is used to synthesize the samples of the minority class.

SMOTE algorithm (Chawla et al.; 2002) is one of the most well-known oversampling algorithms. It aims at attaining a class balance by fabricating data points randomly between the minority data point and its K-nearest neighbour.

Han et al. (2005) introduced some extension, such SMOTE-Borderline 1 and SMOTE-Borderline 2, which creates the minority samples on the decision boundaries among the different classes. Borderline-SMOTE 1 generates synthetic minority samples only for those data points that are endangered to be classified as majority class. Borderline-SMOTE 2 works similar to Borderline-SMOTE 1, however it oversamples the majority class samples along with performing the borderline-SMOTE. The oversampling of majority and minority samples is carried out until a desired balance is achieved between the classes. SVM SMOTE finds a hyperplane that differentiates the classes with the maximum margin, with the support vectors acting as an anchor in the separating plane. New minority samples are synthesized around the support vectors (Nguyen et al.; 2011).

Zhang et al. (2017) uses SMOTE SVM on six UCI datasets for the binary classification of high dimensional data. The study concludes that the SVM-BRFE(SVM Border-Resampling Feature Elimination) algorithm (precision of 91.3%) performs better than the original SVM-RFE algorithm (precision of 87.7%).

ADASYN is an improvement of SMOTE. This method is designed to create class balance and to adjust the classification limits adaptively with difficult samples. Aditsania et al. (2017) proposes the use of ADASYN with a back propagation classifier to predict customer churn in the telecom industry. ADASYN uses density distribution as a criterion to automatically determine the number of synthetic samples to be generated for each minority datapoint (He et al.; 2008).

Wu et al. (2017) states that oversampling can result in some noisy data which has to be removed in order to build a proper data learning model. For noise removal from insurance data, the SMOTE-RSB algorithm was used, as it filters out the sampling records when they have a similarity which is greater than a particular threshold. The SMOTE-IPF algorithm resamples synthetic sampled data to multi noise filter. SMOTE-FRST removes synthetic samples which are lesser than a given distance threshold.

3 Methodology

The research question for this study is to establish the best performing machine learning algorithm for identifying duplicate advertisements, thereby providing better customer service by reducing the number of repeat advertisements directed at the customers.

The objective of this study is to fabricate a sufficiently balanced dataset and to develop a classification model that can identify duplicate ads efficiently.

The methodology for this study is developed on the lines with CRoss Industry Standard Process for Data Mining (CRISP-DM).

3.1 Business understanding

The duplicity of data hosted on the online marketplace is a concern shared by both, consumers as well as the corporate organisation who owns the e-commerce website. If the data is not appropriate, the e-commerce websites stand to lose sizable amounts of potential profit due to customer dissatisfaction. Due to the presence of duplicate ads, it becomes difficult for a consumer to find the desired product, as manual intervention is required to filter out the advertisements of the unwanted products. These repetitive advertisements hinder the user experience and results in an increase in the cost of data storage and maintenance.

The definition of the initial problem was simple, the objective is to leverage previous researches performed in this area and to build upon it to find a unique way to form a model that identifies duplicate advertisements. The two primary approaches considered to solve this problem were:

1. to analyse the information for each seller in order to determine whether they were in the business of duplicate ad posting by tracking their activity; or,

2. to analyse the advertisement content and frame it as a text analytics problem.

The first approach identifies if the seller is highly likely to post a duplicate advertisement by studying the seller's style of writing, reviews given by the customers, and personal information, such as name, location, contact number, profile picture and so on. The second approach analyses the content of the advertisements, such as title, description, price and so on, and identifies if they are duplicates or not. The limitations in the availability of data and the issues around handling of personal information of the seller discouraged the use of the first approach. On the other hand, a sufficient amount of advertisement content related data was available from free sources. Hence, it was viable to analyze the content of advertisements to predict duplicity.

3.2 Data acquisition

An attempt was made to contact various e-commerce websites to request for datasets but unfortunately this effort did not yield any results. The next step was to build a web scraper that would collect the advertisements hosted on Olx for all Indian locations. India was chosen as it is densely populated with a high probability of sellers posting duplicate ads. Olx was selected, primarily, because it is one of the largest marketplaces for India, as it provides the ease of seller account creation and ad posting.

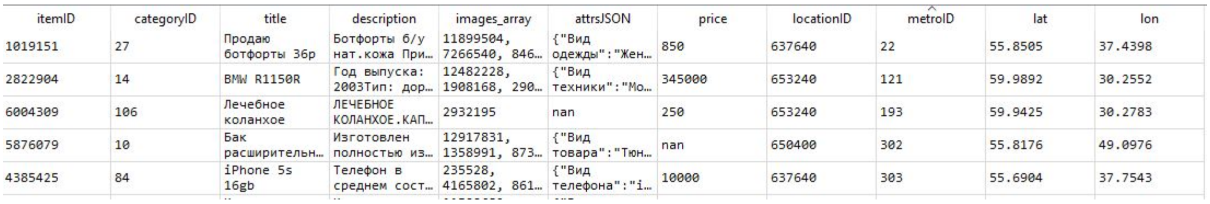
The problem with the above mentioned method was the manual identification of the duplicate and non-duplicate ad pairs, to prepare the training and testing datasets (Henzinger; 2006). Manual fabrication of data is prone to human errors, which could compromise the results of the research. Hence, an alternative solution was implemented that was to collect the data from the Kaggle website. Considerations were taken to ensure that the data was not pre-processed. The scope of this project is limited to the commercial advertisements, as the results are targeted to improve the customer experience. The concept of this project is universal and can be applied to any of the e-commerce websites.

3.3 Data preparation

The research data consists of two datasets:

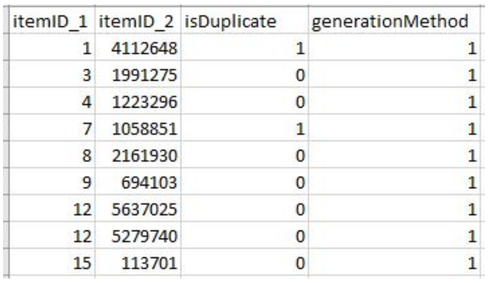
1. advertisement content dataset, ItemInfo_train.csv, having 11 columns, namely: itemID, categoryID, title, description, images_array, price, locationID, metroID, latitude, longitude (Figure 1)

2. labelled dataset, ItemPairs_train.csv, having 4 attributes, namely: itemID1, itemID2, generationMethod and isDuplicate (Figure 2).



itemID	categoryID	title	description	images_array	attrsJSON	price	locationID	metroID	lat	lon
1019151	27	Продаю ботфорты 36р	Ботфорты б/у нат.кожа При…	11899504, 7266540, 846…	{"Вид одежды":"Жен…	850	637640	22	55.8505	37.4398
2822904	14	BMW R1150R	Год выпуска: 2003Тип: дор…	12482228, 1908168, 290…	{"Вид техники":"Мо…	345000	653240	121	59.9892	30.2552
6004309	106	Лечебное коланхое	ЛЕЧЕБНОЕ КОЛАНХОЕ.КАП…	2932195	nan	250	653240	193	59.9425	30.2783
5876079	10	Бак расширительн…	Изготовлен полностью из…	12917831, 1358991, 873…	{"Вид товара":"Тюн…	nan	650400	302	55.8176	49.0976
4385425	84	iPhone 5s 16gb	Телефон в среднем сост…	235528, 4165802, 861…	{"Вид телефона":"i…	10000	637640	303	55.6904	37.7543

Figure 1: Ad content dataset

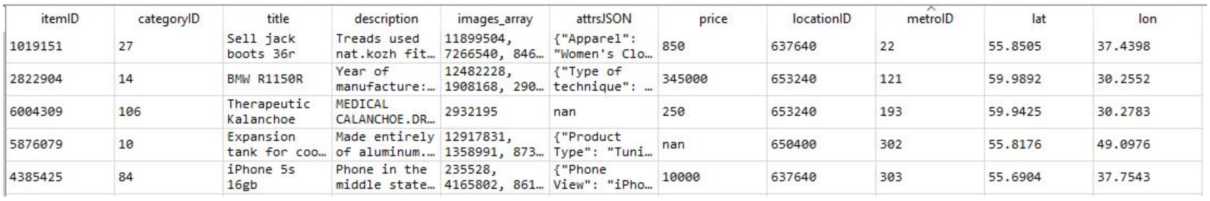


itemID_1	itemID_2	isDuplicate	generationMethod
1	4112648	1	1
3	1991275	0	1
4	1223296	0	1
7	1058851	1	1
8	2161930	0	1
9	694103	0	1
12	5637025	0	1
12	5279740	0	1
15	113701	0	1

Figure 2: Labelled dataset

The **isDuplicate** column, relates to (Figure 2), of the ItemPairs_train.csv dataset is the dependent variable. It is valued as 1 if the pair of data points is duplicate and 0 if they are not duplicate.

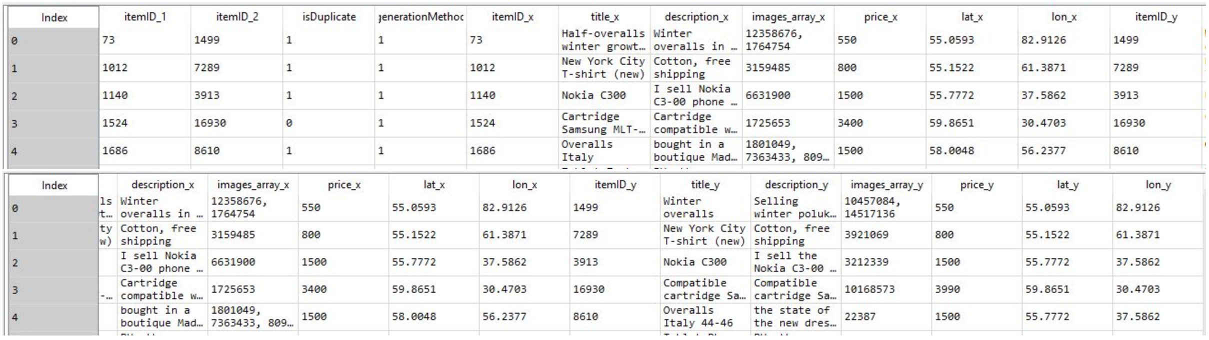
As shown is (Figure 1), the text in the ad content dataset was in the Russian language. Although the python libraries that are used in the project are universal and can run on any language, it was taken into consideration that the content should be translated into English, refer (Figure 3), for the benefit of readers who cannot understand the language. Also, the author is a non-Russian speaker, so it was difficult to understand the data and identify if and what preprocessing steps are required. Hence, in order to have a better understanding of the data and to verify the results, it was beneficial to have the dataset translated to a familiar language. (Wan; 2012)



itemID	categoryID	title	description	images_array	attrsJSON	price	locationID	metroID	lat	lon
1019151	27	Sell jack boots 36r	Treads used nat.kozh fit…	11899504, 7266540, 846…	{"Apparel": "Women's Clo…	850	637640	22	55.8505	37.4398
2822904	14	BMW R1150R	Year of manufacture:…	12482228, 1908168, 290…	{"Type of technique": …	345000	653240	121	59.9892	30.2552
6004309	106	Therapeutic Kalanchoe	MEDICAL CALANCHOE.DR…	2932195	nan	250	653240	193	59.9425	30.2783
5876079	10	Expansion tank for coo…	Made entirely of aluminum.…	12917831, 1358991, 873…	{"Product Type": "Tuni…	nan	650400	302	55.8176	49.0976
4385425	84	iPhone 5s 16gb	Phone in the middle state…	235528, 4165802, 861…	{"Phone View": "iPho…	10000	637640	303	55.6904	37.7543

Figure 3: English translation of Ad content dataset

On manual verification of the translation, it was found a few Russian characters were converted into meaningless dummy variables. Thus, the translated dataset was cleaned of unwanted characters. The resulting dataset had 3.3 million records. This dataset was then merged with the second dataset, relates to (Figure 4), taking ItemID as the primary key, resulting in 2.9 million records. Missing values were checked and removed from the merged dataset. After data cleaning and transformation, the resulting dataset had 2.4 million records.



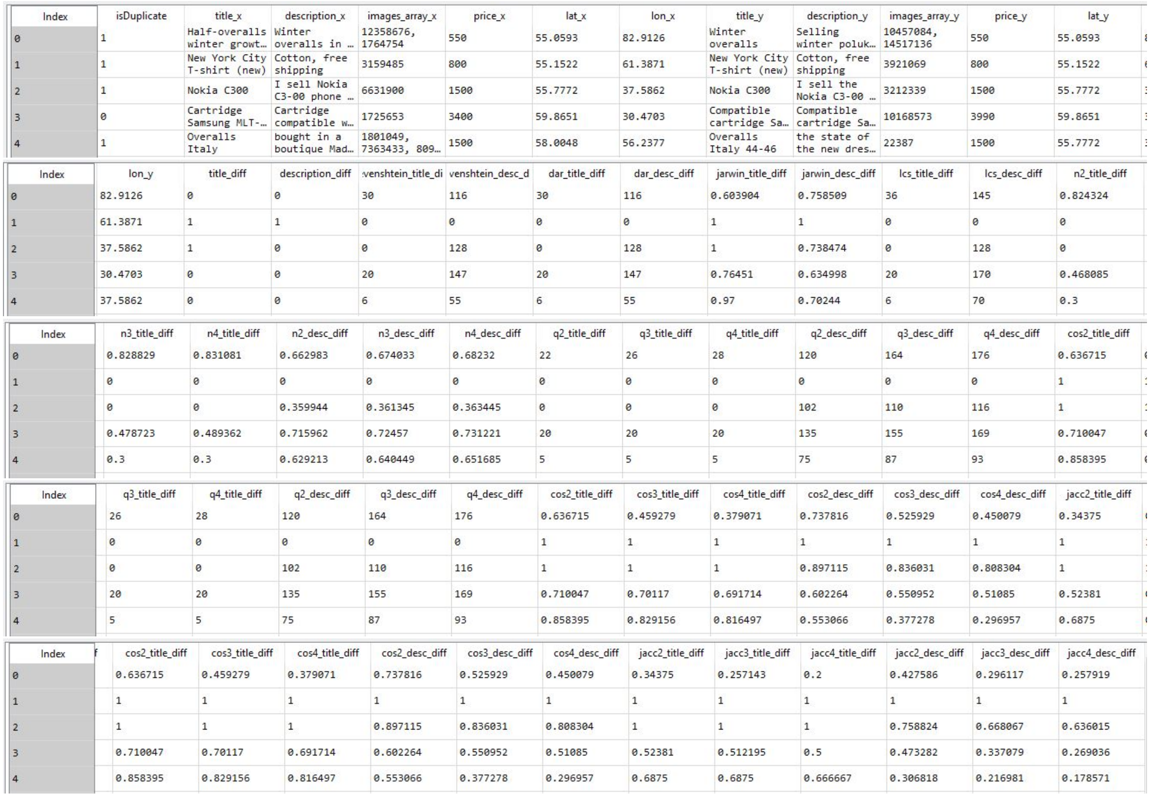
Index	itemID_1	itemID_2	isDuplicate	generationMethoc	itemID_x	title_x	description_x	images_array_x	price_x	lat_x	lon_x	itemID_y
0	73	1499	1	1	73	Half-overalls winter growt…	Winter overalls in …	12358676, 1764754	550	55.0593	82.9126	1499
1	1012	7289	1	1	1012	New York City T-shirt (new)	Cotton, free shipping	3159485	800	55.1522	61.3871	7289
2	1140	3913	1	1	1140	Nokia C300	I sell Nokia C3-00 phone …	6631900	1500	55.7772	37.5862	3913
3	1524	16930	0	1	1524	Cartridge Samsung MLT-…	Cartridge compatible w…	1725653	3400	59.8651	30.4703	16930
4	1686	8610	1	1	1686	Overalls Italy	bought in a boutique Mad…	1801049, 7363433, 809…	1500	58.0048	56.2377	8610

Index		description_x	images_array_x	price_x	lat_x	lon_x	itemID_y	title_y	description_y	images_array_y	price_y	lat_y	lon_y
0	ls t…	Winter overalls in …	12358676, 1764754	550	55.0593	82.9126	1499	Winter overalls	Selling winter poluk…	10457084, 14517136	550	55.0593	82.9126
1	ty w)	Cotton, free shipping	3159485	800	55.1522	61.3871	7289	New York City T-shirt (new)	Cotton, free shipping	3921069	800	55.1522	61.3871
2		I sell Nokia C3-00 phone …	6631900	1500	55.7772	37.5862	3913	Nokia C300	I sell the Nokia C3-00 …	3212339	1500	55.7772	37.5862
3	-…	Cartridge compatible w…	1725653	3400	59.8651	30.4703	16930	Compatible cartridge Sa…	Compatible cartridge Sa…	10168573	3990	59.8651	30.4703
4		bought in a boutique Mad…	1801049, 7363433, 809…	1500	58.0048	56.2377	8610	Overalls Italy 44-46	the state of the new dres…	22387	1500	55.7772	37.5862

Figure 4: Merged dataset

3.4 Feature creation

The feature selection and classification processes are dependent on each other. In order to train the model for improved performance, a number of features were developed according to the properties of the dataset. Text based features were obtained from the title and description of the advertisements, and they were as follows: relative difference between the title and description of a pair of ads, Levenshtein similarity, Damerau similarity, optical string alignment, Jaro-Winkler similarity, longest common subsequence, n-grams (where n is 2,3 and 4),q-grams, cosine, and jaccard similarity for the ngrams. Other features include the difference between the prices of two ads, location of the ads, and number of images in the ads.



Index	isDuplicate	title_x	description_x	images_array_x	price_x	lat_x	lon_x	title_y	description_y	images_array_y	price_y	lat_y
0	1	Half-overalls winter growt…	Winter overalls in …	12358676, 1764754	550	55.0593	82.9126	Winter overalls	Selling winter poluk…	10457084, 14517136	550	55.0593
1	1	New York City T-shirt (new)	Cotton, free shipping	3159485	800	55.1522	61.3871	New York City T-shirt (new)	Cotton, free shipping	3921069	800	55.1522
2	1	Nokia C300	I sell Nokia C3-00 phone …	6631900	1500	55.7772	37.5862	Nokia C300	I sell the Nokia C3-00 …	3212339	1500	55.7772
3	0	Cartridge Samsung MLT-…	Cartridge compatible w…	1725653	3400	59.8651	30.4703	Compatible cartridge Sa…	Compatible cartridge Sa…	10168573	3990	59.8651
4	1	Overalls Italy	bought in a boutique Mad…	1801049, 7363433, 809…	1500	58.0048	56.2377	Overalls Italy 44-46	the state of the new dres…	22387	1500	55.7772

Index	lon_y	title_diff	description_diff	venshtein_title_di	venshtein_desc_d	dar_title_diff	dar_desc_diff	jarwin_title_diff	jarwin_desc_diff	lcs_title_diff	lcs_desc_diff	n2_title_diff
0	82.9126	0	0	30	116	30	116	0.603904	0.758509	36	145	0.824324
1	61.3871	1	1	0	0	0	0	1	1	0	0	0
2	37.5862	1	0	0	128	0	128	1	0.738474	0	128	0
3	30.4703	0	0	20	147	20	147	0.76451	0.634998	20	170	0.468085
4	37.5862	0	0	6	55	6	55	0.97	0.70244	6	70	0.3

Index	n3_title_diff	n4_title_diff	n2_desc_diff	n3_desc_diff	n4_desc_diff	q2_title_diff	q3_title_diff	q4_title_diff	q2_desc_diff	q3_desc_diff	q4_desc_diff	cos2_title_diff
0	0.828829	0.831081	0.662983	0.674033	0.68232	22	26	28	120	164	176	0.636715
1	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0.359944	0.361345	0.363445	0	0	0	102	110	116	1
3	0.478723	0.489362	0.715962	0.72457	0.731221	20	20	20	135	155	169	0.710047
4	0.3	0.3	0.629213	0.640449	0.651685	5	5	5	75	87	93	0.858395

Index	q3_title_diff	q4_title_diff	q2_desc_diff	q3_desc_diff	q4_desc_diff	cos2_title_diff	cos3_title_diff	cos4_title_diff	cos2_desc_diff	cos3_desc_diff	cos4_desc_diff	jacc2_title_diff
0	26	28	120	164	176	0.636715	0.459279	0.379071	0.737816	0.525929	0.450079	0.34375
1	0	0	0	0	0	1	1	1	1	1	1	1
2	0	0	102	110	116	1	1	1	0.897115	0.836031	0.808304	1
3	20	20	135	155	169	0.710047	0.70117	0.691714	0.602264	0.550952	0.51085	0.52381
4	5	5	75	87	93	0.858395	0.829156	0.816497	0.553066	0.377278	0.296957	0.6875

Index	cos2_title_diff	cos3_title_diff	cos4_title_diff	cos2_desc_diff	cos3_desc_diff	cos4_desc_diff	jacc2_title_diff	jacc3_title_diff	jacc4_title_diff	jacc2_desc_diff	jacc3_desc_diff	jacc4_desc_diff
0	0.636715	0.459279	0.379071	0.737816	0.525929	0.450079	0.34375	0.257143	0.2	0.427586	0.296117	0.257919
1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	0.897115	0.836031	0.808304	1	1	1	0.758824	0.668067	0.636015
3	0.710047	0.70117	0.691714	0.602264	0.550952	0.51085	0.52381	0.512195	0.5	0.473282	0.337079	0.269036
4	0.858395	0.829156	0.816497	0.553066	0.377278	0.296957	0.6875	0.6875	0.666667	0.306818	0.216981	0.178571

Figure 5: Feature creation

3.5 Oversampling

An issue with machine learning models is that the training data points for each class must be similar in number. To achieve class balance in the dataset, six different oversampling approaches have been used in this study. Alternatively, undersampling techniques can be used, in which a part of majority class is dropped to attain the class balance, resulting in removal of information. This can result in the loss of useful information which could be relevant for training the model, thereby, compromising the performance of the classifier. Hence, oversampling technique was chosen over undersampling as a preferred technique.

For this study, the minority class is oversampled by synthesizing minority class samples and not by replicating the samples. Replication of samples can cause the classifiers to overfit, hence synthesis was considered as a more suitable option (Eshmawi and Nair; 2014). The results of the classifier were compared for random oversampling, different versions of SMOTE and ADASYN.

3.6 Classifiers

This study compares the performance of four classification algorithms for different oversampling methods. Generally, a supervised machine learning algorithm depends on the features and the class probability of the training data. Each classifier has a unique approach of utilizing the features and class information.

3.6.1 Gradient boosting tree

Gradient boosting produces a prediction model by estimating the weak prediction models. Relative to other classifiers in this study, gradient boost requires less time and power for computation.

3.6.2 Logistic regression

Logistic regression is widely implemented for dichotomous distinction between the two classes.

3.6.3 Naive Bayes

Since the algorithm assumes strong independence of features, it is efficient for high dimensional data. As a classifier, Naive Bayes requires comparatively less time and less computational power than any other model in this study.

3.6.4 Support Vector Machine

Despite the fact that SVM uses more computational time and power than any other classifier in this study, Yang and Liu (1999) found that SVM is one of the best performing classifiers for high dimensional text data.

3.7 Performance measure

Past research shows that accuracy is not a reliable measure for imbalanced datasets because high accuracy is achieved when predicting for the data as the majority class (López et al.; 2013). According to Powers (2011), ROC (Receiver Operating Characteristic) curve and F1 score are selected as the appropriate performance measurements.

The performance measure for this study is taken as F1 score. F1 score represents the trade-off between the precision and the recall. It avoids using the true negative, which can be extremely high for an imbalanced dataset classification. The results of the classifiers are compared against the study conducted by Suh et al. (2017). The aim of this research was to study the effect of different oversampling techniques on topic based classification of Korean news articles.

4 Implementation

4.1 Data pre-processing

The python library *googletrans* was used to translate the Russian content into English. The result of the translation was verified by visually inspecting a random sample of data.

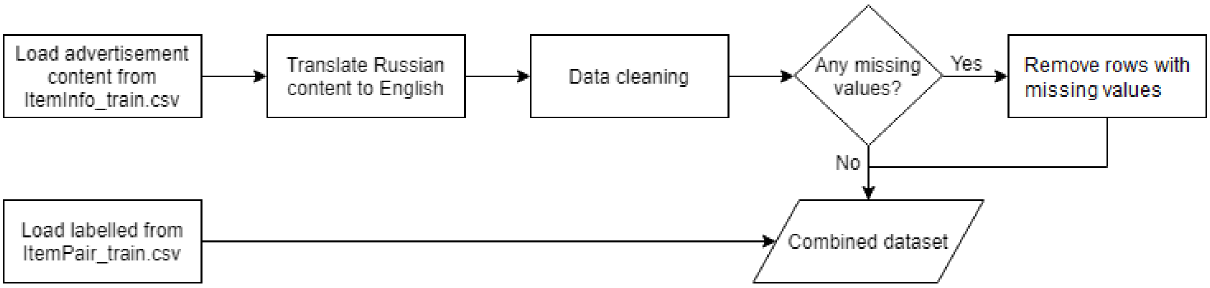

Load advertisement content from ItemInfo_train.csv
Translate Russian content to English
Data cleaning
Any missing values?
Yes
Remove rows with missing values
No
Load labelled from ItemPair_train.csv
Combined dataset

Figure 6: Data pre-processing workflow diagram

Though the majority of translated content was clean, there were a few instances where Russian characters were converted into dummy variables. For example, some Russian characters were converted to '???', 'x047',or spaces between two characters was converted to the pattern 'u200bu200b'. Once the data was translated into English, the Python package *re*, for regex, was used to remove the unwanted patterns in the datasets. Once the data was translated the unnecessary fields such as 'locationID', 'metroID', 'categoryID', 'attrsJSON' were removed. Since the research data is divided into two datasets, the ad content dataset is merged using an inner join on the itemID_1 and itemID_2 columns of the labelled dataset. The merged dataset was then checked for 'NaN' values and the respective rows were removed. The final merged dataset had 2,403,189 records with 35 columns.

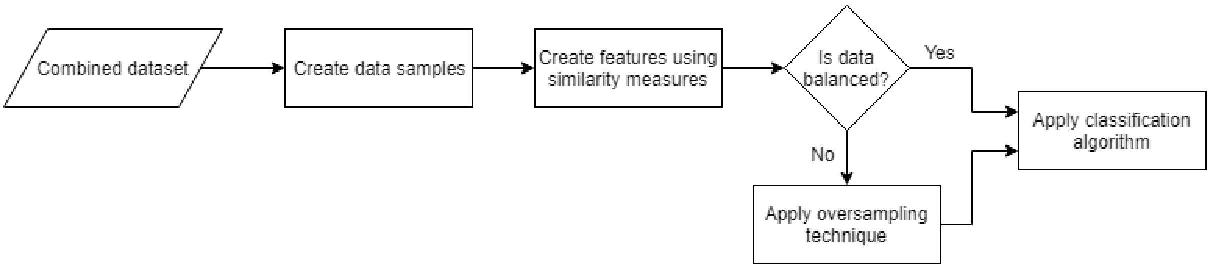

Combined dataset
Create data samples
Create features using similarity measures
Is data balanced?
Yes
No
Apply classification algorithm
Apply oversampling technique

Figure 7: Duplicate detection workflow diagram

4.2 Data sampling

One of the major challenges while using nearly 3 million records was the computational time and power required. The translation of data took more than a week, and running an XGBoost with minimalistic features took around 35 hours.The python console threw a *memory full exception* when the SVM classification model was trained with 3 million records. Hence it was not feasible to utilise the entire dataset. A viable solution to the problem was to sample the data. Hence, the dataset was randomly sampled into subsets of 50000, 100000, 500000, 1 million and 1.5 million records. The performance of the classifier was measured against each of these sample size, resulting in 120 computations.

4.3 Feature creation

The feature creation was done by using the strsim library in Python. This library was developed by luozhouyang and is available free-source on GitHub (GitHub; 2018). The similarity between the titles and the descriptions of the pair of ads was measured against the list similarity indexes mentioned in section 3.4. The number of images per ad is calculated by counting the number of comma-separated values in the images_array column. The difference between the absolute numerical values of price, number of images, and latitude-longitude is taken as three additional features.

4.4 Oversampling

For each computation, the dataset was randomly sampled into training and testing data in a ratio of 3:1, with the percentage of class representation similar over each set. Oversampling methods were implemented using the Python imblearn library. The minority class of the dataset was oversampled until a class balance of 1:1 was attained.

4.5 Classification model

All the classifiers, except the gradient boosting tree, were implemented in Python using the scikit-learn libraries. Gradient boosting tree was implemented using the XGBoost package. The parameters for XGBoost were selected as follows: the maximum depth of a tree was 5, the learning rate was 0.1 and the fraction of observations and columns to be randomly sampled for each tree was 0.8. The parameters for Logistic Regression are: penalty was 'l1', inverse of regular strength was 1x10^6 and maximum iteration was 5000. The SVM and Naive Bayes classifiers used default parameters. The performance of the classifiers is measured in terms of F1 score, implemented using scikit-learn library. The best model is considered to be the one which has the best F1 score across the four classifiers.

5 Evaluation

Japkowicz et al. (2000) conducted a study to determine the influence of varying data size, class imbalance level and complexity on the performance of the classifiers. The tables below show the comparison of the performances of each classifier for various oversampling methods, across different sample sizes.

Table 1 shows the F1 scores of different classifiers on 1.5 million samples of imbalanced data. When the training data was imbalanced, the F1 score of all the classifiers was averaged at 0.61. The best performing model for the imbalanced dataset was Logistic regression with an F1 score of 0.64 whereas the worst performing model was Naive Bayes with an F1 score of 0.56

5.1 Experiment 1: Gradiant Boosting Tree

Table 2 shows the performance of gradient boosting trees for varying sample sizes using the six oversampling techniques. For smaller samples, the classifier showed better results when paired with ADASYN. The overall performance of the gradient boosting tree improved drastically as the sample size was increased. The best F1 score for 50000 records

Classifiers	F1 score
Gradient Boosting Tree	0.6038
Logistic Regression	**0.6459**
Naive Bayes	0.5612
SVM	0.6387

Table 1: The F1 score for imbalanced data set of sample size 1500000

Sample size	Random over-sampling	SMOTE	Borderline-SMOTE1	Borderline-SMOTE2	SVM-SMOTE	ADASYN
50000	0.6018	0.6141	0.6142	0.6152	0.6125	**0.6173**
100000	0.6279	0.6154	0.6411	0.6417	0.6219	**0.6432**
500000	0.7921	0.7974	0.7993	0.8018	0.7951	**0.8021**
1000000	0.8579	**0.8710**	0.8689	0.8704	0.8693	0.8595
1500000	0.8273	0.8739	0.8695	**0.8762**	0.8714	0.8606

Table 2: F1 score for Gradient Boosting Tree

was 0.61 whereas for 1.5 million records it was 0.87, which is an increase of 42%. For the sample size of 1 million, the performance of SMOTE was the best. With 1.5 million records, Borderline-SMOTE2 gave the best result.

5.2 Experiment 2: Logistic Regression

Sample size	Random over-sampling	SMOTE	Borderline-SMOTE1	Borderline-SMOTE2	SVM-SMOTE	ADASYN
50000	0.6028	0.6394	0.6427	**0.6608**	0.6561	0.6580
100000	0.7291	0.7562	0.7896	**0.7947**	0.7473	0.7618
500000	0.8014	0.8127	**0.8159**	0.8006	0.8058	0.8083
1000000	0.8650	0.8793	0.8691	**0.8803**	0.8392	0.8747
1500000	0.8753	0.8991	0.8692	0.8734	0.8816	**0.9143**

Table 3: F1 score for Logistic Regression

Table 3 shows the performance of logistic regression for different sample sizes while using the six oversampling techniques. Logistic regression classifier showed better results when paired with Borderline-SMOTE2. However, for the sample size of 1.5 million, the ADASYN oversampling technique performed the best.

5.3 Experiment 3: Naive Bayes

Table 4 compares the performance of different oversampling techniques, using varying sample sizes, for the Naive Bayes classifier. The classifier gave better results when paired

Sample size	Random over-sampling	SMOTE	Borderline-SMOTE1	Borderline-SMOTE2	SVM-SMOTE	ADASYN
50000	0.5617	0.5746	0.6017	0.6914	0.6750	**0.6992**
100000	0.6022	0.6397	0.6991	0.7028	0.6949	**0.7190**
500000	0.6803	0.6829	0.6963	0.6927	0.7195	**0.7308**
1000000	0.7064	**0.7593**	0.7343	0.7425	0.7396	0.7284
1500000	0.7841	0.7928	0.7934	**0.8065**	0.7992	0.7976

Table 4: F1 score for Naive Bayes

with ADASYN. The Naive Bayes classifier gave a spectrum of results for small dataset. For 50000 records, the classifier, when paired with random oversampling, gave the lowest F1 score of 0.56. However, it outperformed every other classification model when paired with ADASYN. The performance of Naive Bayes improved by only 15% when the sample size increased from 50000 to 1.5 million records.

5.4 Experiment 4: Support Vector Machine

Sample size	Random over-sampling	SMOTE	Borderline-SMOTE1	Borderline-SMOTE2	SVM-SMOTE	ADASYN
50000	**0.6495**	0.6396	0.5977	0.6218	0.6307	0.5999
100000	0.6593	**0.7302**	0.7194	0.7205	0.7249	0.7187
500000	0.7380	0.7706	0.7698	**0.8046**	0.7993	0.7290
1000000	0.8547	0.8688	0.8835	**0.8937**	0.8764	0.8803
1500000	0.8919	0.9042	0.8821	**0.9151**	0.8848	0.8962

Table 5: F1 score for SVM

Table 5 compares the performance of the SVM classifier paired with different oversampling techniques, using varying sample sizes. The classifier gave better results when paired with Borderline-SMOTE2.

5.5 Discussion

In this research, high performance scores were achieved when the training sample size was big and the data was balanced. 1.5 million records were randomly sampled from the dataset of 3 million. Subsequently, 75% of the sampled data was used to train the model, whereas 25% was used for testing. The natural dataset was fairly imbalanced and was provided to the classifiers. The best F1 score for the imbalanced dataset was 0.64 achieved with logistic regression. This data sample was then balanced using 6 oversampling techniques and it was found that the best result was obtained by SVM and logistic regression models, with an F1 score of 0.91. Hence, the study reiterates the results from the Weiss and Provost (2003) study which shows that a classifier performs better with a balanced class distribution rather than a natural distribution. The results also agrees with those obtained by Suh et al. (2017). The work by Suh et al. (2017) compares the performance of various classifiers on balanced data and different levels of imbalanced data, concluding that highest performance scores are reached when the data is balanced.

In this study, the experiment was performed on different sections of data and the results verify that the performance of the classifier, for balanced dataset, increases with an increase in data size. The dependency between the sample size and the performance of a classifier has been discussed in the work of Sordo and Zeng (2005). Sordo and Zeng (2005) compares the impact of the size of sample data on the classification accuracy for Naive Bayes, Decision Trees and Support Vector Machines over a set of 8500 text excerpts.

Suh et al. (2017) resampled 1000 data samples, which originally had a class balance of 9:1. The performance of the classifier on balanced dataset results in an F1 score of 0.81. In this study, an F1 score of 0.91 was achieved with 1.5 million data samples, which reiterates the fact that the performance of classifiers improves with an increase in data size.

Another observation made from the study is the similarity in the performance of SVM and logistic regression classifiers. When the sample size was 1.5 million, logistic regression and SVM showed similar results and were the best performing models, with F1 score of 0.91. This result verifies that close relations between SVM and logistic regression, as mentioned by Vapnik (1999) and Zhang et al. (2003). Both the models can be viewed as probabilistic models that minimise the cost associated with misclassification based on the likelihood ratio. Logistic regression worked best when paired with ADASYN, whereas SVM worked best when paired with Borderline-SMOTE 2.

The purpose of this study was to build a model that can detect duplicate advertisements in the online marketplace. A similar study was conducted by Burk et al. (2017) to build a model, called Apollo, for detecting duplicate job classifieds on online recruitment portals. The dataset was balanced with 500 samples of duplicate ads and 500 of non-duplicates. The F1 score for Apollo for 1000 samples was 0.45. For this study, when a random sample of 1000 records was taken, the performance of Logistic regression with SMOTE was 0.62. Thus, this model outperforms the Apollo for a 1000 records.

6 Conclusion and Future Work

This study compares the performance of various oversampling techniques and classification models to deal with the issue of duplicate ads in the online marketplace.

As mentioned in section 3, alternative approaches could have been taken for identification of duplicates. The area of future work lies in the development of the alternative approach; that is to investigate the information for each seller in order to determine whether they are in the business of duplicate ad posting. Essentially, the combination of this approach, along with the scope of this study, will give a better prediction of whether a seller is likely to post a duplicate ad, and the type of ads that are more likely to be reproduced.

This study focused on the text based fields of advertisements and compared the titles, descriptions, pricing and location of the pairs of ads. However, the scope of this study can be extended to image-based features. Features can be created by comparing the images published with the ads.

The limitation of this research is that it does not analyse the entire dataset that was available on Kaggle. This was due to the time constraint and limitations on the hardware. Further research could be conducted on the entire dataset to check if the results of this study still hold true. Big data storage solutions can be used to handle the huge volume of data. Alternatively, as the models developed in this study are generic for detecting duplicates, data can be obtained for any of the e-commerce websites and the experiments can be reconducted.

The scope of this model is not limited to detecting duplicate advertisements. This study can be applied to any text-based classification of imbalanced data. Depending on the nature of the dataset, different features can be extracted and relevant class balancing technique can be applied. In this study, similar to others in the field, the text-based features were first converted to numeric features and then oversampling techniques were applied across all the numerical data fields. However, a recent study by Castellanos et al. (2018) proposed the use of oversampling technique in string space. A similar approach for oversampling could be applied to this study. Additionally, different machine learning algorithms could be used for the classification model. If there is sufficient time and resources to train the model, more features could be engineered, and experiments could be conducted using artificial neural networks.

References

Aditsania, A., Saonard, A. L. et al. (2017). Handling imbalanced data in churn prediction using adasyn and backpropagation algorithm, *Science in Information Technology (ICSITech), 2017 3rd International Conference on*, IEEE, pp. 533–536.

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B. and Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques, *arXiv preprint arXiv:1707.02919* .

Bilke, A. and Naumann, F. (2005). Schema matching using duplicates, *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, IEEE, pp. 69–80.

Burk, H., Javed, F. and Balaji, J. (2017). Apollo: Near-duplicate detection for job ads in the online recruitment domain, *Data Mining Workshops (ICDMW), 2017 IEEE International Conference on*, IEEE, pp. 177–182.

Castellanos, F. J., Valero-Mas, J. J., Calvo-Zaragoza, J. and Rico-Juan, J. R. (2018). Oversampling imbalanced data in the string space, *Pattern Recognition Letters* **103**: 32–38.

Chawla, N. V., Bowyer, K. W., Hall, L. O. and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique, *Journal of artificial intelligence research* **16**: 321–357.

Chen, S., He, H. and Garcia, E. A. (2010). Ramoboost: ranked minority oversampling in boosting, *IEEE Transactions on Neural Networks* **21**(10): 1624–1642.

Ertl, O. (2017). Superminhash-a new minwise hashing algorithm for jaccard similarity estimation, *arXiv preprint arXiv:1706.05698* .

Eshmawi, A. and Nair, S. (2014). Semi-synthetic data for enhanced sms spam detection:[using synthetic minority oversampling technique smote], *Proceedings of the 6th International Conference on Management of Emergent Digital EcoSystems*, ACM, pp. 206–212.

Feng, L., Wang, H., Jin, B., Li, H., Xue, M. and Wang, L. (2018). Learning a distance metric by balancing kl-divergence for imbalanced datasets, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* .

Gaikwad, S. and Bogiri, N. (2015). Levenshtein distance algorithm for efficient and effective xml duplicate detection, *Computer, Communication and Control (IC4), 2015 International Conference on*, IEEE, pp. 1–5.

GitHub (2018). python-string-similarity, Website. Accessed on 2018-07-27.
URL: *https://github.com/luozhouyang/python-string-similarity*

Han, H., Wang, W.-Y. and Mao, B.-H. (2005). Borderline-smote: a new over-sampling method in imbalanced data sets learning, *International Conference on Intelligent Computing*, Springer, pp. 878–887.

Hassanian-esfahani, R. and Kargar, M.-j. (2018). Sectional minhash for near-duplicate detection, *Expert Systems with Applications* .

He, H., Bai, Y., Garcia, E. A. and Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning, *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, IEEE, pp. 1322–1328.

He, H. and Garcia, E. A. (2009). Learning from imbalanced data, *IEEE Transactions on knowledge and data engineering* **21**(9): 1263–1284.

Henzinger, M. (2006). Finding near-duplicate web pages: a large-scale evaluation of algorithms, *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 284–291.

Japkowicz, N. et al. (2000). Learning from imbalanced data sets: a comparison of various strategies, *AAAI workshop on learning from imbalanced data sets*, Vol. 68, Menlo Park, CA, pp. 10–15.

Kondrak, G. (2005). N-gram similarity and distance, *International symposium on string processing and information retrieval*, Springer, pp. 115–126.

Kovacevic, A., Devedzic, V. and Pocajt, V. (2010). Enhancing a core journal collection for digital libraries, *Program* **44**(2): 132–148.

Lin, W.-C., Tsai, C.-F., Hu, Y.-H. and Jhang, J.-S. (2017). Clustering-based undersampling in class-imbalanced data, *Information Sciences* **409**: 17–26.

Liu, X.-Y., Wu, J. and Zhou, Z.-H. (2009). Exploratory undersampling for class-imbalance learning, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **39**(2): 539–550.

López, V., Fernández, A., García, S., Palade, V. and Herrera, F. (2013). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, *Information Sciences* **250**: 113–141.

Lu, W., Li, Z. and Chu, J. (2017). Adaptive ensemble undersampling-boost: a novel learning framework for imbalanced data, *Journal of Systems and Software* **132**: 272–282.

Naumann, F. and Herschel, M. (2010). An introduction to duplicate detection, *Synthesis Lectures on Data Management* **2**(1): 1–87.

Nguyen, H. M., Cooper, E. W. and Kamei, K. (2011). Borderline over-sampling for imbalanced data classification, *International Journal of Knowledge Engineering and Soft Data Paradigms* **3**(1): 4–21.

Phankokkruad, M. (2017). Efficient similarity measurement by the combination of distance algorithms to identify the duplication relativity, *International Conference on Computer and Information Science*, Springer, pp. 219–232.

Powers, D. M. (2011). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation.

Shoen, E. J., Shoen, S. J., Venkataraman, S. and Kestner, J. A. (2012). Online marketplace for moving and relocation services. US Patent 8,135,627.

Smith, M. D., Bailey, J. and Brynjolfsson, E. (1999). *Understanding digital markets: review and assessment*, MIT press.

Sordo, M. and Zeng, Q. (2005). On sample size and classification accuracy: a performance comparison, *International Symposium on Biological and Medical Data Analysis*, Springer, pp. 193–201.

Suh, Y., Yu, J., Mo, J., Song, L. and Kim, C. (2017). A comparison of oversampling methods on imbalanced topic classification of korean news articles, *Journal of Cognitive Science* **18**(4): 391–437.

Urvoy, T., Chauveau, E., Filoche, P. and Lavergne, T. (2008). Tracking web spam with html style similarities, *ACM Transactions on the Web (TWEB)* **2**(1): 3.

Vapnik, V. (1999). *The nature of statistical learning theory*, Springer Verlag.

Vaughan, L. (2014). Discovering business information from search engine query data, *Online Information Review* **38**(4): 562–574.

Wan, X. (2012). A comparative study of cross-lingual sentiment classification, *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*, IEEE Computer Society, pp. 24–31.

Wang, Y., Zeng, D., Zheng, X. and Wang, F. (2009). Propagation of online news: dynamic patterns, *Intelligence and Security Informatics, 2009. ISI'09. IEEE International Conference on*, IEEE, pp. 257–259.

Weiss, G. M. and Provost, F. (2003). Learning when training data are costly: The effect of class distribution on tree induction, *Journal of Artificial Intelligence Research* **19**: 315–354.

Weissman, S., Ayhan, S., Bradley, J. and Lin, J. (2015). Identifying duplicate and contradictory information in wikipedia, *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries*, ACM, pp. 57–60.

Wu, Z., Lin, W., Zhang, Z., Wen, A. and Lin, L. (2017). An ensemble random forest algorithm for insurance big data analysis, *Computational Science and Engineering (CSE) and Embedded and Ubiquitous Computing (EUC), 2017 IEEE International Conference on*, Vol. 1, IEEE, pp. 531–536.

Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods, *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 42–49.

Zhang, C., Guo, J. and Lu, J. (2017). Research on classification method of high-dimensional class-imbalanced data sets based on svm, *Data Science in Cyberspace (DSC), 2017 IEEE Second International Conference on*, IEEE, pp. 60–67.

Zhang, J., Jin, R., Yang, Y. and Hauptmann, A. G. (2003). Modified logistic regression: An approximation to svm and its applications in large-scale text categorization, *ICML*, Vol. 3, pp. 888–895.