EVALUATING STOP WORDS IN SENTIMENT ANALYSIS OF AMAZON REVIEWS

Naz Yasar

A Thesis Submitted to the
University of North Carolina Wilmington in Partial Fulfillment
of the Requirements for the Degree of
Master of Science

Department of Computer Science

Congdon School of Supply Chain, Business Analytics, and Information Systems

University of North Carolina Wilmington

2022

Approved by

Advisory Committee

Ron Vetter

Jeffrey Cummings

Douglas Kline
Chair

Accepted By

Dean, Graduate School

TABLE OF CONTENTS

ABSTRACT

In this paper, we wanted to understand the stop words effect on sentiment analysis by predicting the star ratings of Amazon reviews. To accomplish this project, we used 3 different stop word levels and 3 different product datasets and built a model of Support Vector Regression.

DEDICATION

This thesis is dedicated to my professor Douglas Kline, my parents, and my best friend Emre Gokce who have supported me throughout my thesis process. Thank you all for making me see this adventure through to the end. I couldn't have done this without you.

LIST OF FIGURES

INTRODUCTION

The Internet is the best source nowadays for any organization to know public opinions about their products and services. Many consumers form an opinion about a product by reading reviews. Many reviews get generated, and it is difficult to handle such a huge number of reviews and analyze them. However, it is critical that these textual reviews are analyzed for understanding their sentiments from them. People are looking for product reviews before they purchase something.

Most of the data on the internet is raw data. It is hard to check billions of data without an algorithm so these data can be extracted using NLP and text analytics techniques. We have 3 different product review datasets to check users' sentiments and opinions about a particular product. In this paper, customer reviews from Amazon.com will be checked in three ways. First, we test the dataset including all its stop words, second, we remove stop words using the short stop word list which includes 25 stop words and third we remove stop words using the long stop word list which includes 318 stop words.

To accomplish this, we use the Support Vector Regression method to predict the star rating of the reviews. Sentiment analysis of product reviews also helps to enhance product features, improve customer service by understanding their expectations, improve marketing strategies, etc. Sentiment analysis particularly is helpful when it comes to negative reviews. It helps discover the exact shortcomings of the products. Generally, the ratings and the price of the product are simple heuristics used by the customers to decide upon the final purchase of the product.

Sentiment analysis is the process of analyzing text data to forecast the emotion that is connected with it. Sentiments can be classified as negative, neutral, and positive. This data can then be used for marketing purposes, such as to target ads at people who are more likely to buy certain products. People use social media platforms to express their ideas and feelings about any products or topics. Internet users create millions of posts in one second. These give huge amounts of raw data to review and analyze. It is impossible to go through billions of data manually so using sentiment analysis makes it possible to check all this data automatically. The main component in sentiment analysis is a machine learning algorithm that determines how people feel about different topics by analyzing their language.

Star Ratings Level	**General Meaning**
	I hate it.
	I don’t like it.
	It’s okay.
	I like it.
	I love it.

Figure 1: Rating System for Amazon.com

LITERATURE REVIEW

There are many papers about sentiment analysis and customer review analysis that have been published in the past years and it keeps increasing.

In [7], they focus on the text of the reviews and evaluate and predict the star ratings. They use eight different film reviews from Amazon. In the research, they exclude 4-star ratings from the datasets, and to have distinct classes they divide ratings into two categories: high (5 stars) and low (1, 2, 3 stars). In preprocessing, they remove stop words without explanation. They use the most common two classifiers in this area of research, Support Vector Machines (SVM) and Naïve Bayes to classify the ratings. They randomly select 90 percent of the data for training and leave 10 percent for the testing. In this research, they found out SVM classifier model using features selected based on information gain had better performance compared to Naïve Bayes.

In another study [8], authors try to find a sentiment polarity score (SP score) rather than binary polarity which is good or bad. They use Support Vector Regression (SVR) to implement a solution to this problem. They also use a Naïve Bayes classifier to evaluate the effect of sentence subjectivity detection. In preprocessing, they keep the stop words. With the SVR machine learning method, they predicted SP scores. And they compare two methods for estimating SP scores: pSVMs and SVR. Their result showed that SVR performs better, and it avoids high penalty mistakes.

In paper [5], they are talking about the noisy nature of tweets data which include abbreviations and regular forms. With the sentiment analysis, they check if removing stop words affect the overall meaning of the tweet in a helping way or hampers the effectiveness of Twitter sentiment classification methods. They use 6 different stop word identification methods from 6 different datasets, and they use 2 supervised sentiment classification methods. According to

their paper, the authors state ‘A well-known method to reduce the noise of textual data is the removal of stop words. They also explain the stop words removal methods that we use these methods in our project.

In the paper [17], the author’s goal is to build classification models that can accurately predict the ratings based on the reviews. They find that using binary bag-of-word representation, adding bi-grams, imposing minimum frequency constraints and normalizing texts have positive effects on model performance. They use the F1 score as comparison metrics and find out their simpler Logistic Regression and Support Vector Machine models predict sentiments more effectively than their more complex models such as Gradient Boosting, LTSM, and BERT. They also use the scikit-learn library in their project. Lastly, they support the idea of ‘there is no one-fits-all solution to building sentiment analysis models’

In today's world people are making online shopping from E-commerce websites such as Amazon.com, eBay etc. and these websites have big influence on products' reputation as they provide review options for the buyers to tell if they like the product or not. Companies interested in to do text mining customer opinions/sentiments on products to satisfy their customer needs, improve their products' quality and having the most profitable way at the end. [12] In [1], L. Jack and Y.D. Tsai says that text mining tools and algorithms can help uncover customer attitudes and sentiments on products they have purchased and used. The data we have online keep increasing every day and it cause overloading problems.

In 1998, Kline [4] mentioned this problem of getting thousands of documents even you are doing quick search online. So, in our project, we have total of 6638 reviews which combined all three datasets, and it would take too long to check each of them individually and getting the correct analysis so like Kline mentioned, reduced reading list makes sense in huge number of documents. In that paper he used vector space document and formulated six different optimization models for creating reduced reading lists. Kline noted that information overload has become a significant problem and researchers used that reduced reading lists in their projects to save running time on computers when they run their models.

STOP WORDS

When we are writing we write some words without thinking about them much, the words that look like they are not helpful, such as ‘the’ and ‘is’. Those two-example word is common words in English, and they are called stop words. Stop words can be any word in a stop word list but we know them as most used words in the languages. Stop words are generally used in Text Mining and Natural Language Processing (NLP) to eliminate words in data that has the less useful information. Stop words are typically articles, prepositions, conjunctions, or pronouns. They don't change the meaning of a query and are used when writing content to structure sentences properly.

The earliest work on stop words removal is attributed to Hans Peter Luhn, who suggested that words in natural language texts can be divided into keyword terms and non-keyword terms. He referred to the latter as stop words. [5] He reported the concept early 1959 in series of reports on automated text searching, indexing, and abstracting. Around that time Luhn was trying to use sixteen kilobytes of on-board memory as effective as possible. So, he considered the reduction of input through ‘pre-processing’ which you can remove the non-informing data and save a byte in onboard memory. Luhn’s first stop words list was including sixteen words. [6]

There is no universal list of stop words used by everyone or any agreed-upon rules for identifying the stop words. Any word can be a stop word depending on what you are doing. In most of the papers, authors remove stop words like it’s a default step of the experiment. In [11], the authors mention that they remove stop words in the preprocessing which they say those words are giving no useful information for the analysis.

AMAZON REVIEWS

Amazon.com is an e-commerce website that people can find millions of different products in one platform. Huge numbers of people visit Amazon.com every day and when they check the products they see the products' description, images, color options, information about the products and customer reviews. Product reviews are optional to make in Amazon, but they encourage their customers to review products after they receive them, such as sending 'share your opinion' emails to their customers. Those customer reviews are important to learn more about the product and decide if that product is right for them. Amazon doesn't approve the review right after the user submits their opinion. They check each review and post them in 48 hours if it doesn't violate any policy such as misleading or manipulating customers.

SCIKIT LEARN

Scikit Learn is a tool for predictive data analysis to find effective results. It's an open-source Machine Learning library for Python that provides supervised learning and unsupervised learning algorithms. It's built on some of the technology like NumPy, pandas and Matplotlib. Scikit Learn provides the functionality of regression, classification, clustering, model selection and preprocessing. We used the scikit learn library to build our sentiment analysis.

PROBLEM STATEMENT

There is good amount of paper published predicting star ratings in classification and some of them removing stop words like its default and some don't remove stop words. We want to find out that is it good to keep stop words in sentiment analysis or if we need to remove stop words to get better results from the models.

SUPPORT VECTOR REGRESSION

SVR has been shown to be an effective method in real-value function estimation, while being less popular than SVM. One of the key advantages of SVR is that its computational complexity is independent of the input space's dimensionality. It also has a high prediction accuracy and great generalization capabilities, says in Khanna's 'Efficient Learning Machines' book. [3] In Scikit-Learn's official website [14] it says that Support Vector Regression fit and works better in the datasets which less than 10000 samples, we have 6638 reviews in total. In the parameters of SVR, we used the default kernel which is called 'rbf', used the epsilon's default which is 0.1, and used the regularization parameter (C)'s default 1.0. The reason we picked epsilon and C's default values is that we tried different numbers in both sections, but default values gave the better results. In Appendix I, we showed the comparison of epsilon 0.1, 0.2, and 0.6.

We didn't want to do polarity classification like negative, neutral, and positive because there is 5 different star ratings and we wanted to find out the exact number of the star rating instead of finding it's either positive or negative.

DATA

We used to same dataset that Yasin Emre Gokce used for his thesis called “Computer Handled Text Reduction: Comparison of Reduced Reading List Creation Methods”. [9]

Three different products and their customer reviews from Amazon have chosen to test our model. One of the reasons for choosing these products is because of their potential difference of context and terminology. We wanted to observe if different sizes of dictionaries and different terms in contexts could affect our analysis. This way if they affect the model, we will be able to analyze it.

The products chosen include:

- **Less (Winner of the Pulitzer Prize)**

  This novel is written by Andrew Sean Greer and the number of reviews collected is 1801 for this product. The reason we wanted to have this product in our research is because we may see more descriptive reviews that cover more content with large vocabulary.

- **Crucial MX300 525GB 3D NAND SATA 2.5 Inch Internal SSD**

  This is an internal hard drive and the number of reviews we collected is 1040 for this product. The reason we wanted to have this product in our research is that we may see more technical, computer-related, and number including reviews.

- **Sylvestidoso Women's A-Line Pleated Sleeveless Little Cocktail Party Dress**

  This is a woman’s dress and the number of reviews we collected is 3797 for this product. The reason we wanted to have this product in our research is because we may see more visually descriptive reviews.

DATA PREPROCESSING

We concatenated the ReviewTitle column which title of the review written by customers and ReviewContent column which include the body of the reviews into the new column called 'Review'. We didn't want to miss any valuable information since some of the titles are longer and some reviews are shorter. Then we dropped the null rows in Review and Rating columns.

Then we imported String library in Python and removed punctuations from the Review column and we converted all the words to lowercase in the reviews. This step's main reason is to prevent possible complexity of the feature set, for example 'want' and 'Want' might be taken as different features by machine learning models, so with converting all the words to lowercase is to make training process and model works better. [11]

The reason we didn't use lemmatization because we didn't touch any word that would be helpful for that specific dataset. And we didn't remove the numbers since we are using book, dress, and SSD dataset and in some reviews, people mention the dress size, book page or the size of the SSD. Also, we don't do any Porter Stemmer or Snowball Stemmer because we didn't want to change any of the words in the corpuses.

BAG-OF-WORDS

In [13] authors say that bag-of-words is commonly used object categorization model. The model has been using in Natural Language Processing and information retrievals. To find the occurrence of each word in document we used the bag-of-words model to have features. With count vectorizer we had features from our dataset.

N-GRAM

In [15], they restate Zipf's Law as follows: *'The nth most common word in a human language text occurs with a frequency inversely proportional to n.'* In our preprocessing we used Count Vectorizer method where we count the occurrences of N-grams. With that method, our code does the following steps:

- Split reviews into the separate text tokens which consist only the letters, punctuation already removed before this step.
- Used Count Vectorizer to convert a collection of reviews to a matrix of token counts and called that 'features', see Appendix A.
- Then we concatenated WebScraperID with that matrix to find easily each tokenized review if we need to.

WebScraperID	Unique ID for each review
Author	Customer name and last name
ReviewTitle	Title of the review
ReviewContent	Review body
Review	Combination of ReviewTitle and ReviewContent
Rating	The 1–5-star rating of the review
VerifiedPurchase	Showing if the review is verified
HelpfulVotesGiven	Number of helpful votes
ReviewDate	The date of the review written.

Figure 2: Descriptions of Data Columns in the Datasets

DATA DETAILS

Ratings/Counts	Book	Dress	SSD
1	285	115	73
2	201	89	30
3	220	222	19
4	282	594	102
5	812	2776	816

Figure 3: Rating Counts of Each Dataset

Mean star rating of book dataset is 3.63 and standard deviation is 1.52.

Mean star rating of dress dataset is 4.54 and standard deviation is 0.93.

Mean star rating of SSD dataset is 4.5 and standard deviation is 1.41.

EXPERIMENT

We used Google Collab environment and used its GPU with Google Drive. We utilized the following programs:

- Excel for getting the outputs of MAE results,
- SPSS for ANOVA statistics,
- Downloaded the real statistics resource pack, called XrealStats, to have the box plots of the results for to read the statistics visually.

EXPERIMENTAL VARIABLES

In our project we used 3 steps of using stop words. In the first experiment, we didn't remove the stop words and wanted to know how model works, then second step we used small stop word list [2], this list includes 25 stop words, see Appendix C.

We used baseline method for our first experiment which means the non-removal of stop words. And there is classical method used in many papers which removing stop words obtained

from pre-compiled lists [5]. We wanted to see how our model works in different level stop words which we call small and large stop word lists.

NON-REMOVAL

In this experiment, we didn't remove any stop words from reviews. Before running the SVR model, the only change made in the dataset was to remove the punctuations and make the vocabulary lower letters. Find the code in Appendix A.

SMALL LIST

In this experiment we only removed the small stop words list (Appendix C) to find out if that 25 common words in English change any difference in the model.

LARGE LIST

In this experiment, we used Scikit Learn's default stop words list which includes 318 different stop words (Appendix C). We removed those words from the reviews and ran the experiment. To get all those 318 words we used the following code:

```
from sklearn.feature_extraction import _stop_words
large_stopwords = _stop_words.ENGLISH_STOP_WORDS
```

CORPUS

	Book	**Dress**	**SSD**
Number of Unique Words Before Removal	8064	4620	5212
Number of Unique Words After Removing Small List	8039	4595	5187
Number of Unique Words After Removing Large List	7778	4365	4954
Average # of Words in Each Review	62	34	76
Standard Deviation of the # of Words for Each Review	95	32	79

Figure 4: Number of Words Details in Corpuses

INDEPENDENT VARIABLES

In the experiment, there are 2 independent variables; datasets and stop word lists. Datasets include the book, SSD, and dress datasets. The level of stop word removal includes the non-removal, small stop words removal, and long stop words removal.

DEPENDENT VARIABLES

In the experiment, the only dependent variable is the Test Mean Absolute Error.

MEAN ABSOLUTE ERROR (MAE)

To measure our sentiment prediction accuracy, we used the Mean Absolute Error metric. The reason we wanted to use the MAE is that it's an easy method for interpreting and it's an evaluation metric for the regression models. It's showing the mean absolute errors of all-star ratings in the test dataset.

Used the following calculation to find the MAE:

MAE = ABS(Predicted Star Rating – Actual Star Rating)

HYPOTHESES

The ANOVA techniques apply when there are two or more independent variables [18]. We have the independent variables mentioned in Section 7.1.2 and we applied the ANOVA to test our hypothesis and interpret the results by comparing the variables.

The main null hypothesis (H_0) is that the Test MAE are equal regardless of the stop word list used. The second hypothesis is that the Test MAE are equal regardless of the corpus. The third hypothesis is that the Test MAE are equal regardless of the combination of stop words list and corpus.

H_0: $\mu_{none} = \mu_{small} = \mu_{large}$

H_0: $\mu_{book} = \mu_{SSD} = \mu_{dress}$

H_0: $\mu_{none,book} = \mu_{none,SSD} = \mu_{none,dress} = \mu_{small,book} = \mu_{small,SSD} = \mu_{small,dress} = \mu_{large,book} = \mu_{large,SSD} = \mu_{large,dress}$

STUDY

We split each dataset into 60 percent for training, 20 percent for validation and 20 percent for testing.

Dataset	# Of Reviews	Average # of Words in Reviews	# Words in Dictionary	Minimum # of Words in Reviews	Maximum # of Words in Reviews
Book	1801	61.69	8064	2	1405
SSD	1040	75.78	5212	2	1047
Dress	3797	33.58	4620	2	221

Figure 5:Number of Reviews and Words in Dictionaries of Each Dataset

# of reviews	Total	Train	Validation	Test
Book	1800	1080	360	360
SSD	1040	624	208	208
Dress	3796	2277	760	760

Figure 6: Number of Reviews in Train/Validation/Test

Train			
Data Set	**Non-removal**	**Small List Removal**	**Large List Removal**
Book	0.444	0.420	0.407
Dress	0.213	0.205	0.236
SSD	0.381	0.375	0.369
Validation			
Data Set	**Non-removal**	**Small List Removal**	**Large List Removal**
Book	1.081	1.014	0.955
Dress	0.507	0.483	0.510
SSD	0.696	0.691	0.687
Test			
Data Set	**Non-removal**	**Small List Removal**	**Large List Removal**
Book	1.139	1.095	1.006
Dress	0.508	0.483	0.497
SSD	0.505	0.499	0.495

Figure 7: Model Performance

In Figure 7, it shows all the results of each step. It's a little bit hard to analyze this table so we converted this to box plots to interpret them (Figure 10, 11, and 12).

Between-Subjects Factors

		N
stopwordlist	Large	1328
	None	1328
	Small	1328
dataset	Book	1080
	Dress	2280
	SSD	624

Figure 8: Portions of ANOVA

Tests of Between-Subjects Effects

Dependent Variable: y_diff

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	271.477[a]	8	33.935	77.308	<.001
Intercept	1451.997	1	1451.997	3307.867	.000
stopwordlist	1.339	2	.670	1.526	.218
dataset	267.881	2	133.940	305.136	<.001
stopwordlist * dataset	2.275	4	.569	1.295	.269
Error	1744.837	3975	.439		
Total	3724.685	3984			
Corrected Total	2016.313	3983			

a. R Squared = .135 (Adjusted R Squared = .133)

Figure 9: ANOVA Statistics

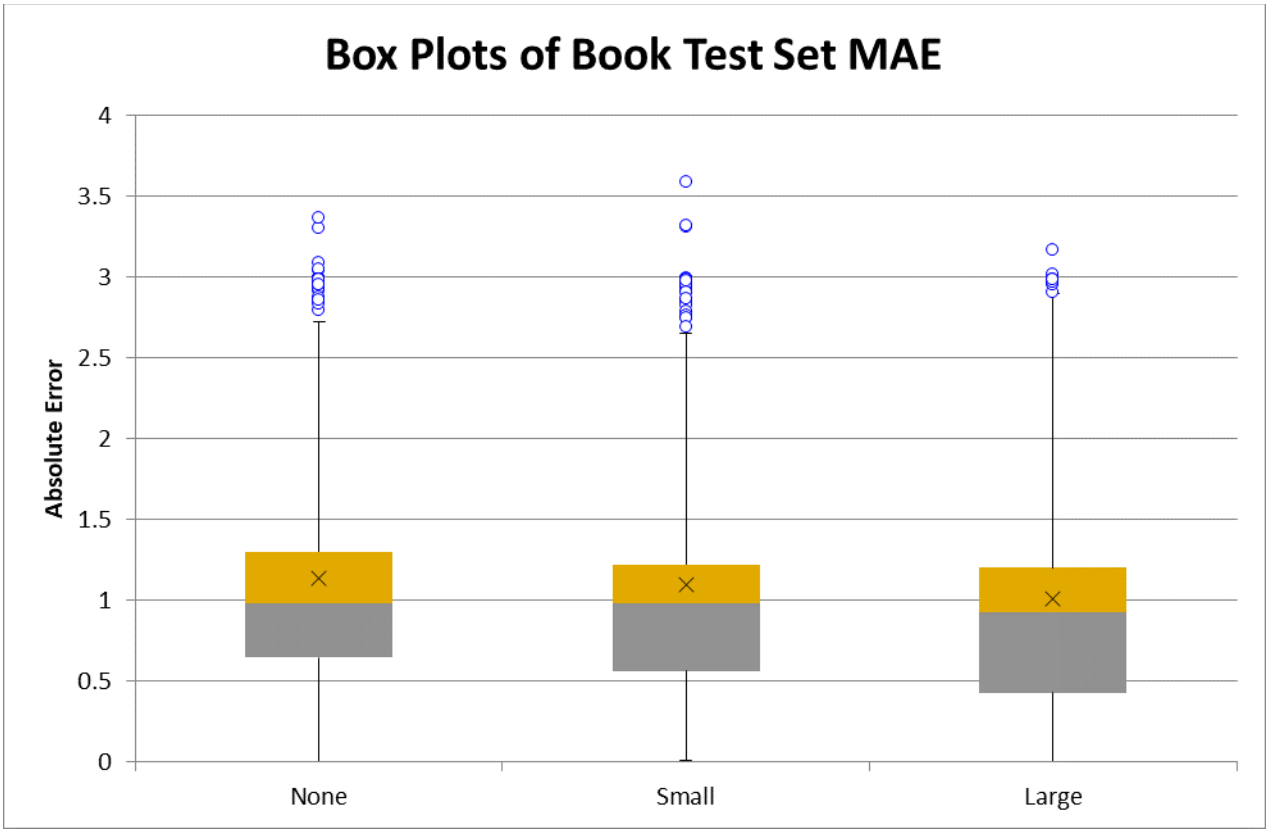


Figure 10: Box Plots of Book Test MAE by Stop Words Levels

In statistics we saw that there is no difference the numbers of outliers between the Non-Removal and Small stop words list but there is a specific difference when we removed the large stop words from the corpus. There are less outliers compared to other experiments. Number of outliers in each step of removal:

Non-Removal: **44**

Small List Removal: **43**

Large List Removal: **16**

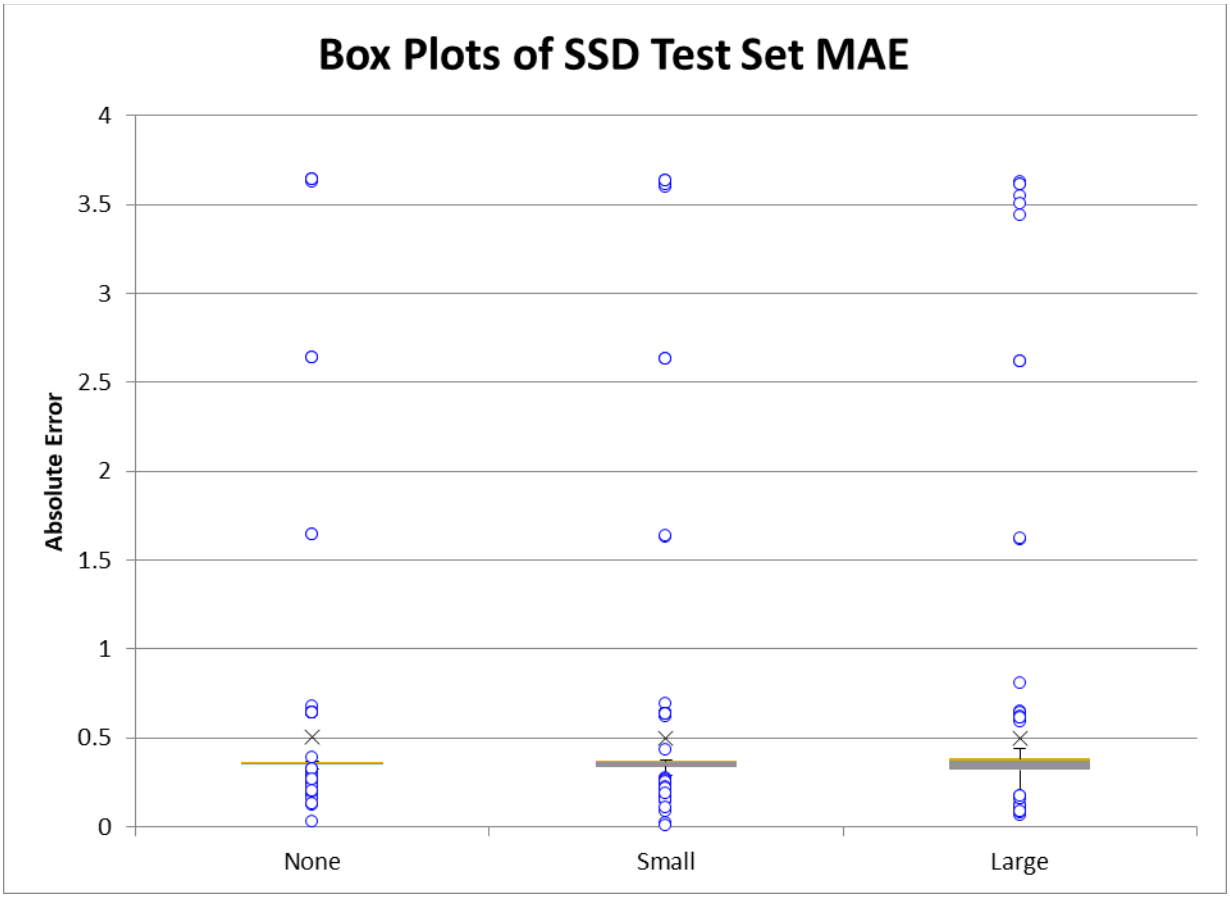


Figure 11: Box Plots of SSD Test MAE by Stop Words Levels

According to boxplot we can say our model worked better in this dataset because of the absolute error is less than 0.5 in every step of the model.

In these statistics we saw the number of outliers was more in non-removal experiment, it gradually going down in each level of stop words removal, we can say MAE is less when you reduce more text (large stop words list) from the corpus. Number of outliers in each step of removal:

Non-Removal: **61**

Small List Removal: **51**

Large List Removal: **39**

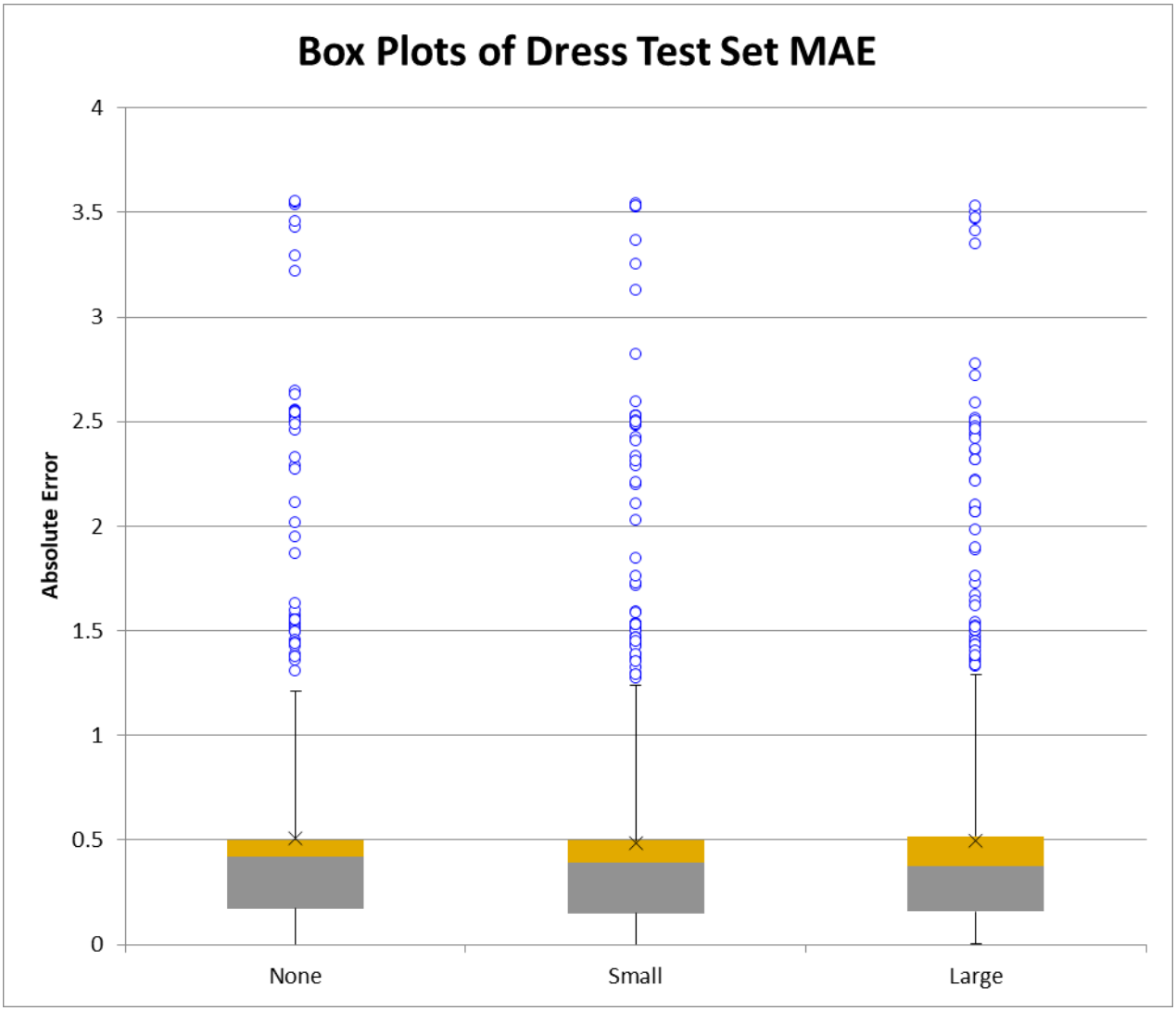


Figure 12: Box Plots of Dress Test MAE by Stop Words Levels

In this analysis we had more outliers compare to the other two datasets. In each level we saw the same number of outliers and there is no change if you remove the stop words or not.

Number of outliers in each step of removal:

Non-Removal: **69**

Small List Removal: **65**

Large List Removal: **67**

CONCLUSION

In our study we wanted to have an answer to how stop words affect in sentiment analysis. Results shows that there is no significant difference between the removal of stop words or keeping them in the corpus. But we did see significant difference between the products. So, we can say the dataset content can affect the accuracy of the model. Also, we used unbalanced datasets which 4-stars and 5-stars ratings numbers higher than others (1, 2, 3 stars).

We see that data reduction (stop words removal) has no significant effect on the running time of the model (see Appendix B) and we see that it makes it slower in larger dictionaries such as the Book dataset.

DISCUSSION

In our study, we saw our 3 dataset's rating numbers showed that we had mostly 4-star and 5-star ratings. So, it might be another reason that our model didn't work well. The only limitation I could say we only used the SVR model for our project. Trying different techniques might be helpful for that specific topic, for example Deep Neural Network.

FUTURE WORK

We used a pre-compiled stop words list in our project but the pattern of the significant result between the datasets shows that creating a custom stop word list for the corpus on the model building might be helpful. Instead of using lists, evaluating each individual stop word and ffindingits value of the word for sentiment analysis, and labeling that word as high-value or low-value would be research for future projects. The effect of the stop words might be helpful in text

analytics, recommendation systems, summarization etc. And like we say in the limitations, using different type of techniques might give different results.

REFERENCES

[1] Yi-Fang Tsai, Lleyana Jack, Using Text Mining of Amazon Reviews to Explore User-Defined Product Highlights and Issues - July 2015, Conference: DMIN 2015

[2] Manning, C., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511809071

[3] Awad, M., Khanna, R. (2015). Support Vector Regression. In: Efficient Learning Machines. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4302-5990-9_4

[4] Kline, D.M. (1998) Optimization Models for Creating Reduced Reading Lists. Working paper # 98-02MIS, Center for Business and Economic Development, Sam Houston State University, https://www.shsu.edu/centers/cbed/documents/working-papers/No.98-02MIS.pdf.

[5] Saif, Hassan & Fernandez, Miriam & Alani, Harith. (2014). On stopwords, filtering and data sparsity for sentiment analysis of twitter. Proceedings of the 9th International Language Resources and Evaluation Conference (LREC'14). 810-817.

[6] Rosenberg, Daniel. "Stop, words." *Representations* 127.1 (2014): 83-92.

[7] Kavousi, Mohammadamir & Saadatmand, Sepehr. (2018). Estimating the Rating of Reviewers Based on the Text.

[8] Okanohara, Daisuke & Tsujii, Jun'ichi. (2005). Assigning Polarity Scores to Reviews Using Machine Learning Techniques. 314-325. 10.1007/11562214_28.

[9] Gokce, Y., Kline, D, Vetter, R., Cummings, J. (2021) Automated Text Reduction: Comparison of Reduced Reading List Creation Methods. Annals of the Master of Science in Computer Science and Information Systems at UNC Wilmington, https://proc.conisar.org/2021/pdf/5595.pdf

[10] Fang, X., Zhan, J. Sentiment analysis using product review data. Journal of Big Data 2, 5 (2015). https://doi.org/10.1186/s40537-015-0015-2

[11] Mujahid, M.; Lee, E.; Rustam, F.; Washington, P.B.; Ullah, S.; Reshi, A.A.; Ashraf, I. Sentiment Analysis and Topic Modeling on Tweets about Online Education during COVID-19. *Appl. Sci.* 2021, *11*, 8438. https://doi.org/10.3390/app11188438

[12] Ashwin Ittoo, Le Minh Nguyen, Antal van den Bosch, Text analytics in industry: Challenges, desiderata and trends, Computers in Industry, Volume 78, 2016, Pages 96-107, ISSN 0166-3615, https://doi.org/10.1016/j.compind.2015.12.001.

[13] Zhang, Y., Jin, R. & Zhou, ZH. Understanding bag-of-words model: a statistical framework. Int. J. Mach. Learn. & Cyber. 1, 43–52 (2010)https://doi.org/10.1007/s13042-010-0001-0

[14] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

[15] Cavnar, William B. and John M. Trenkle. "N-gram-based text categorization." (1994).

[16] Xrealstats - https://www.real-statistics.com/free-download/real-statistics-resource-pack/

[17] Liu, Siqi. (2020). Sentiment Analysis of Yelp Reviews: A Comparison of Techniques and Models.

[18] Judd, Charles M, Gary H. McClelland, and Carey S. Ryan. *Data Analysis: A Model Comparison Approach to Regression, Anova, and Beyond*. , 2017. Internet resource.

APPENDIXES

A: Source Code of Book Dataset

```
import pandas as pd
import numpy as np

!pip install -U -q PyDrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials

auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)

# Load Book Reviews
#Review Title + Review Content combined under a new column calle
d "Review"

# Get file from Google Drive

# https://drive.google.com/file/d/1XIEfcyM54BSH6Gr6hxA2mf6vvu75f
yJL/view?usp=sharing
```

```
fileDownloaded = drive.CreateFile({'id':'1XIEfcyM54BSH6Gr6hxA2mf
6vvu75fyJL'})
```

```
fileDownloaded.GetContentFile('book.csv')
df = pd.read_csv('book.csv')
df.head()
```

	WebScraperID	Author	ReviewTitle	ReviewContent	Review	Rating	VerifiedPurchase	HelpfulVotesGiven	ReviewDate
0	1582229985-5749	Pat S. Zajac	Less is a trip - literally!	Less is a trip - literally! Fun to read, poig...	Less is a trip - literally! Less is a trip - l...	5.0	True	0.0	Saturday, January 4, 2020
1	1582230075-6175	225 Shelly D	Vey very good.	I have never rated a book before. Always left ...	Vey very good. I have never rated a book befor...	5.0	True	0.0	Sunday, January 6, 2019
2	1582230125-6400	Maria A Marana	Wonderful and refreshing	This book is so refreshing, worth it every sin...	Wonderful and refreshing This book is so refre...	5.0	True	0.0	Saturday, August 25, 2018
3	1582230062-6110	Fran	Good book	Very funny	Good book Very funny	5.0	True	0.0	Tuesday, January 1, 2019
4	1582229989-5774	jpack	Less is a terrific book.	I thoroughly enjoyed this book. I laughed out...	Less is a terrific book. I thoroughly enjoyed ...	5.0	True	0.0	Wednesday, September 5, 2018

```
df = pd.read_csv('book.csv', usecols= ['WebScraperID', 'Review',
 'Rating'])
df = df[df['Review'].notna()]
df = df[df['Rating'].notna()]
```

```
CompleteBookDataSet = df
CompleteBookDataSet.sample(5)
```

	WebScraperID	Review	Rating
157	1582230121-6386	One of the best books I've read this year One ...	5.0
640	1582230216-6818	MUCH less than I hoped for. I returned this bo...	1.0
1568	1582229905-5382	Such a disappointment Bought this because I he...	1.0
1528	1582230160-6565	A great book; deserves to be read Barb bought ...	4.0
845	1582230184-6675	Exquisite prose, surpassed only by its sense o...	4.0

```
CompleteBookDataSet['Rating'].value_counts()
```

```
5.0    812
1.0    285
4.0    282
3.0    220
2.0    201
Name: Rating, dtype: int64
```

```
# Preprocessing Text Python Package
# Punctuation Removal
#library that contains punctuation
import string
string.punctuation
```

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

```
#defining the function to remove punctuation
```

```
def remove_punctuation(text):
    punctuationfree="".join([i for i in text if i not in string.
punctuation])
    return punctuationfree
#storing the puntuation free text
CompleteBookDataSet['clean_rev']= CompleteBookDataSet['Review'].
apply(lambda x:remove_punctuation(x))
CompleteBookDataSet.head()
```

	WebScraperID	Review	Rating	clean_rev
0	1582229985-5749	Less is a trip - literally! Less is a trip - l...	5.0	Less is a trip literally Less is a trip lite...
1	1582230075-6175	Vey very good. I have never rated a book befor...	5.0	Vey very good I have never rated a book before...
2	1582230125-6400	Wonderful and refreshing This book is so refre...	5.0	Wonderful and refreshing This book is so refre...
3	1582230062-6110	Good book Very funny	5.0	Good book Very funny
4	1582229989-5774	Less is a terrific book. I thoroughly enjoyed ...	5.0	Less is a terrific book I thoroughly enjoyed t...

 # Lowering the text

```
CompleteBookDataSet['lower_rev'] = CompleteBookDataSet['clean_re
v'].apply(lambda x: x.lower())
CompleteBookDataSet.head()
```

	WebScraperID	Review	Rating	clean_rev	lower_rev
0	1582229985-5749	Less is a trip - literally! Less is a trip - l...	5.0	Less is a trip literally Less is a trip lite...	less is a trip literally less is a trip lite...
1	1582230075-6175	Vey very good. I have never rated a book befor...	5.0	Vey very good I have never rated a book before...	vey very good i have never rated a book before...
2	1582230125-6400	Wonderful and refreshing This book is so refre...	5.0	Wonderful and refreshing This book is so refre...	wonderful and refreshing this book is so refre...
3	1582230062-6110	Good book Very funny	5.0	Good book Very funny	good book very funny
4	1582229989-5774	Less is a terrific book. I thoroughly enjoyed ...	5.0	Less is a terrific book I thoroughly enjoyed t...	less is a terrific book i thoroughly enjoyed t...

```
CompleteBookDataSet = CompleteBookDataSet[['WebScraperID', 'Revi
ew', 'lower_rev', 'Rating']]
```

```
# Count Vectorizer
from sklearn.feature_extraction.text import CountVectorizer

# Count vectorizer

corpus = CompleteBookDataSet['lower_rev']
vectorizer = CountVectorizer(token_pattern=r'(?u)\b\w+\b')
termfrequencyvalues = vectorizer.fit_transform(corpus)


vectorizer.get_feature_names_out()
termfrequencyvalues = termfrequencyvalues.toarray()
features = vectorizer.get_feature_names_out()

termfrequencies = pd.DataFrame(data = termfrequencyvalues, colum
ns = features)

CompleteBookDatasetValues = np.concatenate((CompleteBookDataSet.
to_numpy(), termfrequencyvalues), axis=1)


# transpose the array
```

```
temp = np.array([CompleteBookDataSet['WebScraperID'].to_numpy()]
).T


np.shape(termfrequencyvalues)
(1800, 8064)
np.shape(temp)
(1800, 1)
termfrequencyvalues = np.concatenate((temp, termfrequencyvalues)
, axis=1)
print(termfrequencyvalues)
np.shape(termfrequencyvalues)


column_headings = np.concatenate((CompleteBookDataSet.columns, f
eatures))


CompleteBookDataSet = pd.DataFrame(data = CompleteBookDatasetVal
ues, columns = column_headings)


# train, val, test
from sklearn.model_selection import train_test_split
X = termfrequencyvalues
y = CompleteBookDataSet.Rating
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_s
ize=0.2, random_state=1)

X_train, X_val, y_train, y_val = train_test_split(X_train, y_tra
in, test_size=0.25, random_state=1)

# In this way, train, val, test set will be 60%, 20%, 20% of the
 dataset respectively.

temp = X_train[:,1:]

# MODEL

from sklearn.svm import SVR

svr_rbf = SVR(kernel='rbf', C=1, gamma=0.1, epsilon=0.1)

fit_model = svr_rbf.fit(X_train[:,1:], y_train)
train_predictions = fit_model.predict(X_train[:,1:])
val_predictions = fit_model.predict(X_val[:,1:])

# print mae
train_diff = np.absolute(y_train - train_predictions)
val_diff = np.absolute(y_val - val_predictions)
```

```
train_mae = np.mean(train_diff)
val_mae = np.mean(val_diff)

print(train_mae)
print(val_mae)

0.4436371433785085
1.0806204830443664

test_predictions = fit_model.predict(X_test[:,1:])
test_diff = np.absolute(y_test - test_predictions)
test_mae = np.mean(test_diff)

print(test_mae)

1.139209823953319

scraperIDs = np.array([X_test[:,0]])
scraperIDs = scraperIDs.transpose()
np.info(scraperIDs)
print(scraperIDs)

class:  ndarray
shape:  (360, 1)
```

```
strides:  (8, 2880)
itemsize:  8
aligned:  True
contiguous:  True
fortran:  True
data pointer: 0x558833157800
byteorder:  little
byteswap:  False
type: object
diffs = np.array([test_diff.to_numpy()])
diffs = diffs.transpose()
np.info(diffs)

class:  ndarray
shape:  (360, 1)
strides:  (8, 2880)
itemsize:  8
aligned:  True
contiguous:  True
fortran:  True
data pointer: 0x558833156c00
byteorder:  little
byteswap:  False
type: object
```

```
output = np.concatenate((scraperIDs, diffs), axis = 1)
np.info(output)
```

```
class:  ndarray
shape:  (360, 2)
strides:  (16, 8)
itemsize:  8
aligned:  True
contiguous:  True
fortran:  False
data pointer: 0x5588329f8800
byteorder:  little
byteswap:  False
type: object
```

```
print(output)
```

```
[['1582229981-5731' 1.1198840801819814]
 ['1582230165-6584' 1.0154213983485452]
 ['1582230229-6884' 0.998122200588722]
 ['1582229998-5809' 2.9165799726721127]
 ['1582230064-6122' 0.42189435898965577]
…… ['1582230021-5919' 0.9615879721168694]]
```

```
output_df = pd.DataFrame(output)

# download the output file

from google.colab import files

output_df.to_csv('output.csv', encoding= 'utf-8-sig')

files.download('output.csv')
```

Removal of Small Stop Words List Code (Example from Book Data)

```
from nltk.corpus import stopwords # Import stopwords from nltk.c
orpus
small_stopwords = ['a', 'an', 'and', 'are', 'as', 'at', 'be', 'b
y', 'for', 'from', 'has', 'he', 'in', 'is', 'it','its', 'of', 'o
n', 'that', 'the', 'to', 'was', 'were', 'will', 'with']

from sklearn.feature_extraction.text import CountVectorizer

corpus = CompleteBookDataSet['lower_rev']
vectorizer = CountVectorizer(token_pattern=r'(?u)\b\w+\b', stop_
words = small_stopwords) # Bolded part is for removing stop
words list
termfrequencyvalues = vectorizer.fit_transform(corpus)


vectorizer.get_feature_names_out()
termfrequencyvalues = termfrequencyvalues.toarray()
features = vectorizer.get_feature_names_out()

termfrequencies = pd.DataFrame(data = termfrequencyvalues, colum
ns = features)
```

Removal of Large Stop Words List Code (Example from Book Data)

```
from sklearn.feature_extraction import _stop_words
large_stopwords = _stop_words.ENGLISH_STOP_WORDS

from sklearn.feature_extraction.text import CountVectorizer

corpus = CompleteBookDataSet['lower_rev']
vectorizer = CountVectorizer(token_pattern=r'(?u)\b\w+\b', stop_
words = large_stopwords) # Bolded part is for removing stop
words list

termfrequencyvalues = vectorizer.fit_transform(corpus)


vectorizer.get_feature_names_out()
termfrequencyvalues = termfrequencyvalues.toarray()
features = vectorizer.get_feature_names_out()

termfrequencies = pd.DataFrame(data = termfrequencyvalues, colum
ns = features)
```

B: Running Time and Used Ram Statistics

# of Seconds (GPU)	**None**	**Small**	**Large**
Book	18.80 sec	24.56 sec	24.17 sec
Dress	42.56 sec	51.16 sec	47.66 sec
SSD	4.63 sec	5.14 sec	3.85 sec

Figure 13: Running Time of the SVR Model

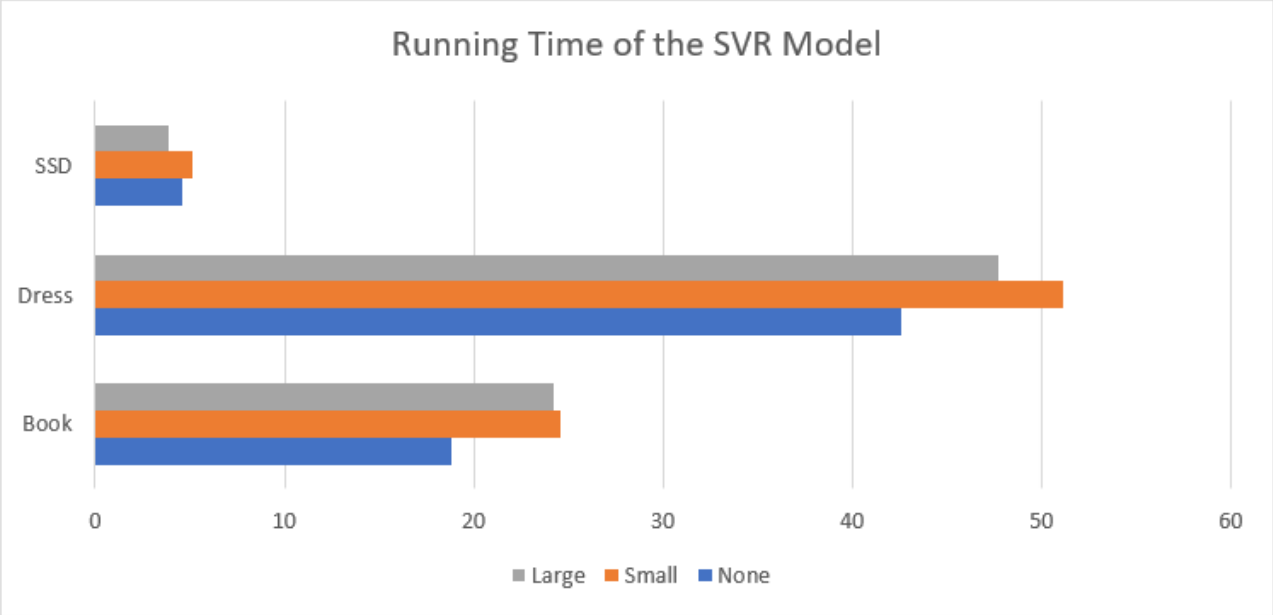


Figure 14: Bar Plot of SVR's Running Time

RAM used (GPU)	None	Small	Large
Book	1.71 GB	1.78 GB	1.77 GB
Dress	1.82 GB	1.93 GB	1.93 GB
SSD	1.48 GB	1.54 GB	1.40 GB

Figure 15: Used RAM when running the SVR Model

C: Small and Large Stop Words Lists

small_stopwords = ['a', 'an', ’and’, 'are', 'as', 'at', 'be', 'by', 'for', 'from', 'has', 'he', 'in', 'is', 'it','its', 'o
f', 'on', 'that', 'the', 'to', 'was', 'were', 'will', 'with']

large_stopwords = {'no', 'to', 'six', 'own', 'that', 'mostly', 'nine', 'should', 'this', 'etc', 'while', 'you',
'there', 'over', 'nobody', 'fifty', 'former', 'somehow', 'seem', 'rather', 'whether', 'although', 'inc',
'formerly', 'whatever', 'when', 'least', 'itself', 'sixty', 'seeming', 'by', 'we', 'also', 'an', 'even',
'between', 'about', 'where', 'must', 'never', 'may', 'system', 'everything', 'still', 'afterwards', 'from',
'my', 'such', 'few', 'through', 'myself', 'though', 'onto', 'since', 'against', 'every', 'off', 'again', 'will',
'else', 'last', 'they', 'elsewhere', 'bottom', 'is', 'amoungst', 'interest', 'be', 'latter', 'towards', 'couldnt',
'con', 'thin', 'go', 'third', 'twenty', 'which', 'her', 'meanwhile', 'see', 'become', 'your', 'name',
'therefore', 'yourself', 'she', 'via', 'noone', 'already', 'i', 'once', 'whence', 'either', 'many',
'throughout', 'mill', 'fifteen', 'everywhere', 'both', 'ten', 'thick', 'four', 'around', 'ours', 'among',
'hereby', 'de', 'yourselves', 'could', 'with', 'empty', 'top', 'it', 'along', 'for', 'two', 'take', 'sometimes',
'perhaps', 'why', 'cant', 'whereas', 'move', 'thence', 'found', 'might', 'toward', 'a', 'the', 'but', 'his',
'find', 'him', 'un', 'whither', 'into', 'fire', 'here', 'during', 'amongst', 'first', 'wherever', 'behind',
'whom', 'nor', 'everyone', 'per', 'whereafter', 'put', 'amount', 'thru', 'indeed', 'beside', 'if', 'so', 'out',
'than', 'until', 'call', 'in', 'anyone', 'ourselves', 'nowhere', 'each', 'one', 'these', 'under', 'eight',
'whereupon', 'at', 'were', 'hundred', 'full', 'themselves', 'give', 'sometime', 'whereby', 'always', 'side',
'or', 'been', 'hereafter', 'latterly', 'beforehand', 'below', 'on', 'their', 'anywhere', 'becomes',
'whenever', 'because', 'upon', 'most', 'are', 'back', 'being', 'very', 'show', 'therein', 'next', 'anyhow',
'often', 'namely', 'ltd', 'somewhere', 'alone', 'whose', 'get', 'now', 'can', 'none', 'us', 'our', 'well',

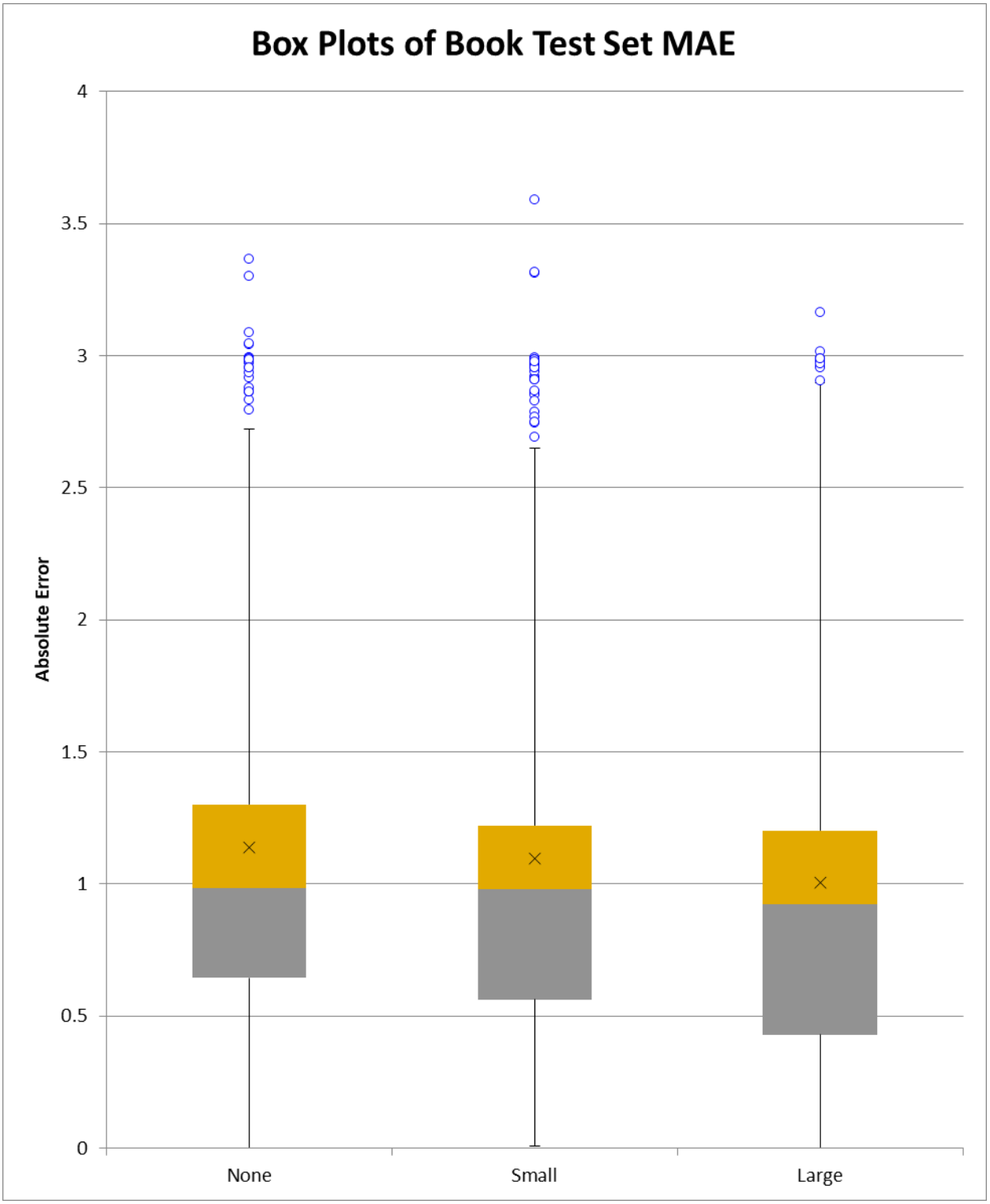
'seems', 'please', 'other', 'something', 'made', 'eg', 'then', 'thereafter', 'ie', 'otherwise', 'himself', 'of', 'forty', 'hereupon', 'up', 'all', 'down', 'am', 'anything', 'ever', 'others', 'much', 'part', 'without', 'herself', 'have', 'me', 'hers', 'those', 'further', 'due', 'beyond', 'above', 'not', 'what', 'whoever', 'front', 'had', 'done', 'would', 'as', 'keep', 'was', 'almost', 'twelve', 'bill', 'thereupon', 'sincere', 'across', 'except', 'herein', 'and', 're', 'another', 'thus', 'together', 'nothing', 'becoming', 'besides', 'hasnt', 'detail', 'fill', 'mine', 'several', 'thereby', 'five', 'became', 'someone', 'only', 'hence', 'who', 'do', 'moreover', 'any', 'cry', 'less', 'enough', 'however', 'too', 'neither', 'cannot', 'describe', 'yet', 'within', 'has', 'eleven', 'serious', 'how', 'wherein', 'anyway', 'yours', 'whole', 'three', 'before', 'after', 'seemed', 'he', 'more', 'nevertheless', 'co', 'same', 'them', 'its', 'some'}

D: Model Performance

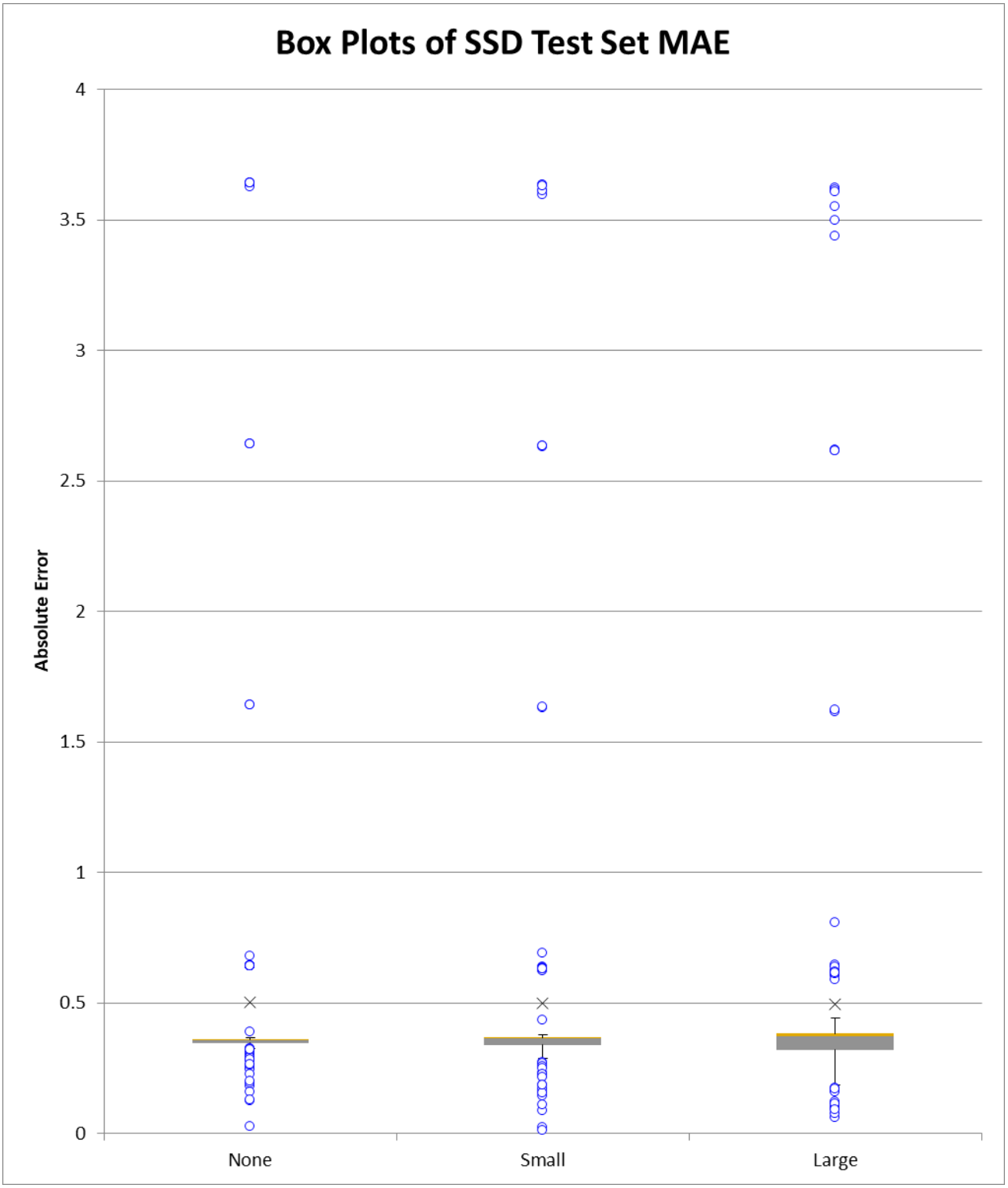
Training			
Data Set	**Non-Removal**	**Small List Removal**	**Large List Removal**
Book	0.444	0.420	0.407
Dress	0.213	0.205	0.236
SSD	0.381	0.375	0.369
Validation			
Data Set	**Non-Removal**	**Small List Removal**	**Large List Removal**
Book	1.081	1.014	0.955
Dress	0.507	0.483	0.510
SSD	0.696	0.691	0.687
Test			
Data Set	**Non-Removal**	**Small List Removal**	**Large List Removal**
Book	1.139	1.095	1.006
Dress	0.508	0.483	0.497
SSD	0.505	0.499	0.495

Figure 16: Model Performance of Training/Validation/Test

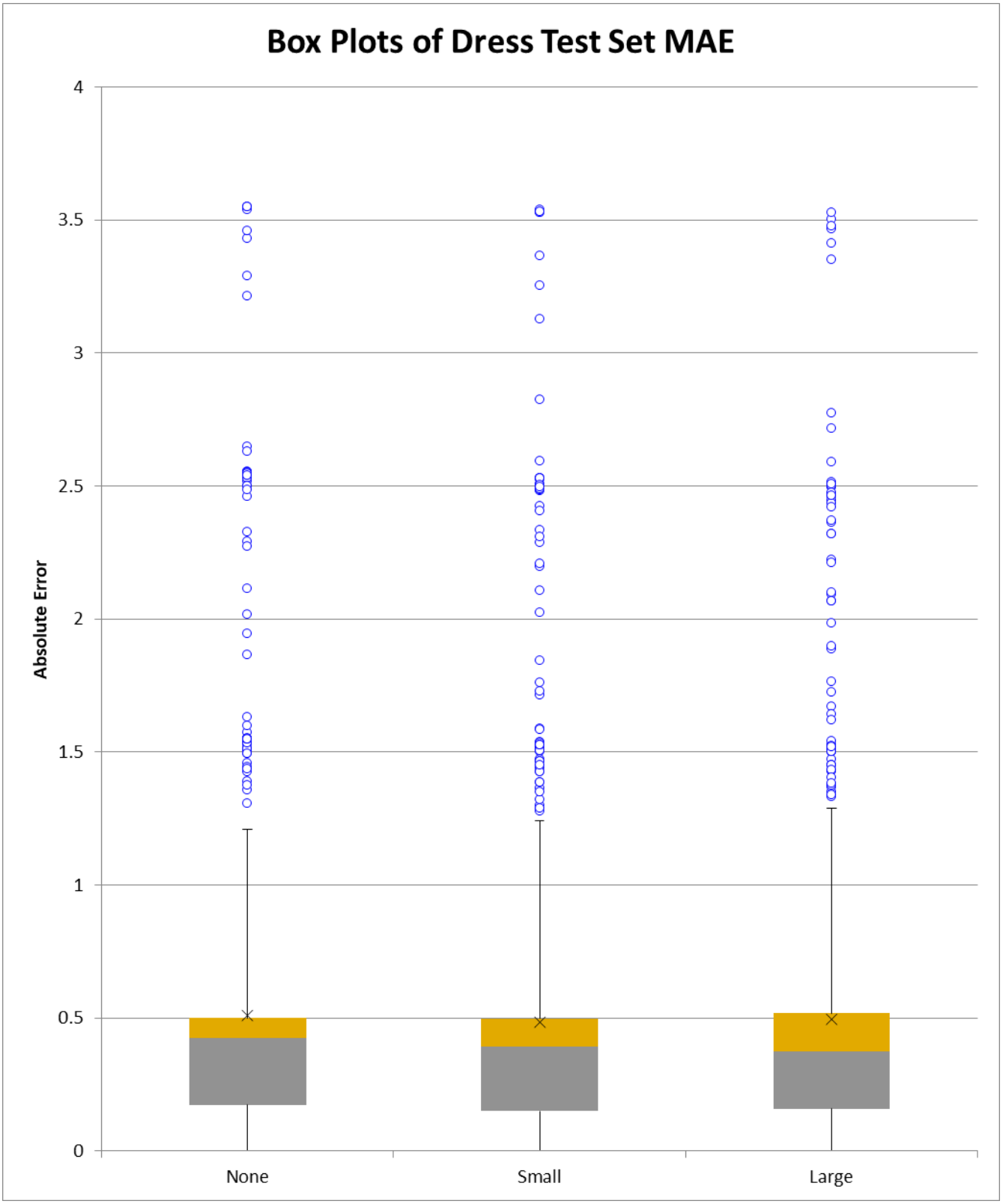
E: Box Plot of Book Test Set MAE



F: Box Plot of SSD Test Set MAE



G: Box Plot of Dress Test Set MAE



H: Descriptive Statistics of Model Test Errors

Descriptive Statistics			
	None	***Small***	***Large***
Mean	1.14	1.10	1.01
Standard Error	0.05	0.05	0.05
Median	0.98	0.98	0.92
Mode	0.02	0.02	#N/A
Standard Deviation	0.87	0.86	0.80
Sample Variance	0.76	0.74	0.65
Range	3.36	3.58	3.16
Maximum	3.37	3.59	3.17
Minimum	0.001	0.009	0.000
Sum	410.12	394.33	361.99
Count	360	360	360
Min	0.002	0.009	0.000
Q1-Min	0.64	0.55	0.43
Med-Q1	0.34	0.42	0.50
Q3-Med	0.32	0.24	0.28
Max-Q3	1.42	1.43	1.69
Mean	1.14	1.10	1.01
Min	0.001	0.009	0.000
Q1	0.64	0.56	0.43
Median	0.98	0.98	0.92
Q3	1.30	1.22	1.20
Max	2.72	2.65	2.90
Mean	1.14	1.10	1.01

Figure 17: Descriptive Statistics of Model Test Errors for Book Dataset

Descriptive Statistics			
	None	***Small***	***Large***
Mean	0.50	0.50	0.50
Standard Error	0.04	0.04	0.04
Median	0.36	0.37	0.37
Mode	0.36	0.37	#N/A
Standard Deviation	0.63	0.62	0.62
Sample Variance	0.39	0.39	0.38
Range	3.62	3.62	3.56
Maximum	3.64	3.63	3.62
Minimum	0.03	0.01	0.06
Sum	104.94	103.87	103.01
Count	208	208	208
Min	0.33	0.29	0.19
Q1-Min	0.02	0.05	0.14
Med-Q1	0.01	0.026	0.05
Q3-Med	0.00	0.00	0.01
Max-Q3	0.01	0.01	0.06
Mean	0.50	0.50	0.50
Min	0.33	0.29	0.19
Q1	0.35	0.34	0.32
Median	0.36	0.37	0.37
Q3	0.36	0.37	0.38
Max	0.37	0.38	0.44
Mean	0.50	0.50	0.50

Figure 18: Descriptive Statistics of Model Test Errors for SSD Dataset

Descriptive Statistics			
	None	***Small***	***Large***
Mean	0.51	0.48	0.50
Standard Error	0.02	0.02	0.02
Median	0.42	0.39	0.37
Mode	0.06	0.08	0.10
Standard Deviation	0.58	0.56	0.56
Sample Variance	0.34	0.31	0.32
Range	3.55	3.54	3.53
Maximum	3.55	3.54	3.53
Minimum	0.00	0.00	0.00
Sum	386.17	366.95	377.49
Count	760	760	760
Min	0.00	0.00	0.00
Q1-Min	0.17	0.15	0.16
Med-Q1	0.25	0.24	0.22
Q3-Med	0.08	0.10	0.14
Max-Q3	0.71	0.74	0.77
Mean	0.51	0.48	0.50
Min	0.00	0.00	0.00
Q1	0.17	0.15	0.16
Median	0.42	0.39	0.37
Q3	0.50	0.50	0.52
Max	1.21	1.24	1.29
Mean	0.51	0.48	0.50

Figure 19: Descriptive Statistics of Model Test Errors for Dress Dataset

I: Epsilon and Regularization Parameter Values

	Epsilon = 0.1	
C	**Train_MAE**	**Val_MAE**
1.5	0.27	1.09
1	0.44	1.08
0.75	0.59	1.11
	Epsilon = 0.2	
C	**Train_MAE**	**Val_MAE**
1.5	0.33	1.11
1	0.49	1.09
0.75	0.61	1.1
	Epsilon = 0.6	
C	**Train_MAE**	**Val_MAE**
1.5	0.61	1.16
1	0.7	1.15
0.75	0.78	1.14

Figure 20: Epsilon and Regularization Parameter Values