Marketing Science Institute Working Paper Series 2022

Report No. 22-117

Detecting Fake Review Buyers Using Network Structure: Direct Evidence from Amazon

Sherry He, Brett Hollenbeck, Gijs Overgoor, Davide Proserpio and Ali Tosyali

Detecting Fake Review Buyers Using Network Structure: Direct Evidence from Amazon

Sherry He* Brett Hollenbeck† Gijs Overgoor‡ Davide Proserpio§
Ali Tosyali¶

July, 2022 ‖

Abstract

Online reviews significantly impact consumers' decision-making process and firms' economic outcomes and are widely seen as crucial to the success of online markets. Firms, therefore, have a strong incentive to manipulate ratings using fake positive reviews. This presents a problem that academic researchers have tried to solve over two decades and on which platforms expend a large amount of resources. Nevertheless, the prevalence of fake reviews is arguably higher than ever. To combat this problem, we collect a dataset of reviews and ratings for thousands of Amazon products and develop a general and highly accurate method for detecting fake reviews. A unique difference between previous datasets and ours is that we directly observe which sellers buy fake reviews. Thus, while prior research has trained models using lab-generated reviews or proxies for fake reviews, we are able to train a model using actual fake reviews. We start by showing that products that buy fake reviews are highly clustered in the product-reviewer network. Therefore, features constructed from this network are highly predictive of which products buy fake reviews, and models trained on these

*UCLA Anderson School of Management; sherry.he.phd@anderson.ucla.edu
†UCLA Anderson School of Management; brett.hollenbeck@anderson.ucla.edu
‡RIT Saunders College of Business; govergoor@saunders.rit.edu
§University of Southern California; proserpi@usc.edu
¶RIT Saunders College of Business; atosyali@saunders.rit.edu

‖Authors listed in alphabetical order. We thank the Morrison Center for Marketing Analytics for generous funding. We thank seminar participants at the Johnson School of Management at Cornell University, the Ross School of Business at the University of Michigan, Pontificia Universidad Catolica de Chile, the Fox School of Business at Temple University, the RIT Marketing Workshop, the USC Marshall AI Workshop, and SCECR 2022 for helpful comments.

features outperform methods not accounting for them. We then show that our network-based approach is also successful at detecting fake reviews even without ground truth data, as unsupervised clustering methods can accurately identify fake review buyers by simply identifying clusters of products that are closely connected in the network. While review text or metadata can be manipulated by sellers who have a strong incentive to evade detection, network-based features are very difficult to manipulate because they follow directly from the economics of fake review marketplaces, making network-based fake review detection approaches robust to manipulation.

1 Introduction

Online reviews have a significant impact on consumer purchase decisions and are widely seen as crucial to the success of online markets(Tadelis, 2016, Molm et al., 2006). Review and rating systems allow buyers and sellers to develop credible reputations in settings that are otherwise mostly anonymous. Because these reputations are crucial for seller outcomes, sellers have a large incentive to manipulate their ratings and inflate their reputation. As a result, online review platforms like Amazon, Yelp, or Tripadvisor have struggled since their inception with the problem of sellers manipulating their ratings with fake reviews. Rating manipulation can cause buyers to buy from lower-quality sellers than they otherwise would, it allows sellers to charge higher prices than if their true reputation was observed, and it lowers trust in reviews and review platforms, making it difficult for high-quality and honest sellers to compete. There is growing empirical evidence that fake reviews harm consumers (He et al., 2022, Akesson et al., 2022). In addition to violating the platform's policies, these practices are the subject of ongoing investigations by the FTC, the UKCMA, and other regulators.

Despite the vast amount of academic research over the past two decades (see Wu et al., 2020 for an extensive review) and the large amounts of time, effort, and money invested by online platforms to detect and remove fake reviews, they are nevertheless as prevalent as ever. Recent studies have found that millions of products on Amazon are using fake reviews, a large share of all reviews are fake (He et al., 2022), and consumers express very low levels of trust in online reviews as a result (Dong et al., 2019).

Most research on the challenge of fake review detection relies on machine learning techniques that exploit features associated with reviews, such as ratings, helpful votes, and text content (Wu et al., 2020). A primary challenge in this approach is the lack of ground-truth data with which to train and test models. In other words, in order to develop models that can identify fake reviews, one must first have a corpus of existing fake reviews (and real reviews) with which to train the model. Researchers have attempted to overcome the inherent

challenge this poses largely by using lab-generated reviews and considering platform-filtered reviews as a proxy for fake reviews. Scholars have criticized these approaches as having serious limitations (Heydari et al., 2015, Mukherjee et al., 2013, Crawford et al., 2015, Vidanagama et al., 2019, Wu et al., 2020.) Lab-generated fake reviews may lack authenticity, and bot-generated or other low-quality reviews (those from non-English speakers or containing many grammatical errors) at best pick the low-hanging fruit and miss the bulk of actual fake reviews. Moreover, using only reviews that are flagged and filtered by the platform's algorithms by definition cannot progress our understanding of how to identify the large number of fake reviews that currently evade those filters (He et al., 2022). Moreover, methods trained on these data have been repeatedly shown to lack external validity (Crawford et al., 2015).

In addition to the lack of high-quality ground truth data, another shortcoming of current methods is that sophisticated actors can evade filtering algorithms trained on historical features. For example, methods relying on text analysis are fundamentally limited in that, even with sophisticated models, human reviewers have strong incentives to evade detection and will therefore strive to write fake reviews that are indistinguishable from organic reviews. If particular phrases or tendencies become used to filter reviews, they can then avoid using these in their reviews. More generally, extant approaches to detecting which products are manipulating their ratings have not grappled with the economic incentives of buyers and sellers. We use unique data in which we observe sellers buy fake reviews and demonstrate how these economic incentives can be harnessed to design an approach that can detect fake review buyers with high accuracy and with limited scope for evasion.

We argue that these problems can be overcome by focusing on the product-reviewer network and identifying products that buy fake reviews rather than the fake reviews themselves. We hand-collect a large new dataset that provides accurate ground truth data by directly identifying a large and representative set of products that buy fake reviews on Amazon.com. Using this data, we study the relative effectiveness of different approaches for detecting what

products manipulate their ratings. We find that, compared to products not using fake reviews, products that use fake reviews are highly connected and clustered in the network, implying that products using fake reviews tend to have more reviewers in common than other products. This follows naturally from the fact that, while regular products receive their reviews from a dispersed set of millions of Amazon customers, products buying fake reviews must rely on the relatively small number of reviewers participating in the fake review marketplace. Therefore, features derived from the product-reviewer network are especially useful for regulating fake reviews because, in addition to being more predictive than text or metadata features, they are more difficult to manipulate by sellers than text or review timing, as they relate directly to the economics of fake review marketplaces.

2 Data

To analyze fake review behavior on Amazon, we begin by collecting data from the private Facebook groups in which sellers buy reviews. These groups are the primary channel by which sellers find reviewers.

From March 2020 to October 2020, we identify about 23 fake review-related groups daily. These groups are large and quite active, each having about 16,000 members on average and 568 fake review requests posted per day per group. Within these Facebook groups, sellers can obtain a five-star review that looks organic. Sellers post product pictures and review requests, after which the potential reviewer and the seller communicate via private Facebook messages. The vast majority of sellers buying fake reviews compensate the reviewer by refunding the cost of the product via a PayPal transaction after the five-star review has been posted along with the cost of the PayPal fee, sales tax, and in some cases, an additional commission. Reviewers are compensated for creating realistic seeming five-star reviews that evade Amazon's filters. This process differs from "incentivized reviews," where sellers offer free or discounted products or discounts on future products in exchange for reviews that

disclose the transaction and are not required to be five stars (Burtch et al., 2018).

To identify which products are buying fake reviews, we hire a group of research assistants to visit these groups and select a random sample of products posted in them. We collect data from these random Facebook fake review groups using this procedure on a weekly basis from October 2019 to June 2020, and the result is a sample of roughly 1,500 unique products.

After identifying products whose ratings are manipulated, we collect data for these products on Amazon.com. We collect reviews and ratings for each of the products on a daily basis. For each review, we observe the rating, product ID, review text, review photos, and helpful votes. Additionally, we collect the full set of reviews for each product twice per month. This allows us to measure to what extent Amazon responds by deleting reviews that it deems as potentially fake.

In addition to collecting this data for the focal products, we collect daily and twice-monthly review data for a set of 2,714 competitor products to serve as a comparison set. To do so, for each focal product, we select the two competitor products that show up most frequently on the same search page as the focal product in the seven days before and seven days after their first FB post. The rationale is that we want to create a comparison set of products that are in the same subcategory as the focal products and have a similar search rank before fake reviews are posted.

3 Identifying which products buying fake reviews

There is an extensive literature on fake review detection on online platforms such as Amazon, Yelp, and Tripadvisor. This literature has proposed methods for detection based on text features, image features, spatiotemporal differences, network features, sentiment, and others (see the many references in Wu et al., 2020.) One of the main hurdles in creating algorithms capable of detecting fake reviews is the lack of ground truth data, i.e. reviews that are known to be fake with absolute certainty, and this literature has been criticized for relying

on low-quality proxies (Wu et al., 2020).

We overcome this issue by focusing on the products that buy fake reviews. As described in Section 2, our dataset allows us to know with absolute certainty which Amazon products bought fake reviews, therefore providing accurate ground truth for our product-level analysis. We can then compare the performance of previously suggested methods and, in particular, test the performance of models that take advantage of the structure of the product-reviewer network.

3.1 Network construction and features generation

Our approach to detecting sellers buying fake reviews exploits the network structure of Amazon products. We start by constructing a product network $G = (V, E)$ using our data, where G is the network, V is the set of nodes (i.e., products), and E is the set of edges. An edge between two products represents the existence of common reviewers. Figure 1 shows the network structure of Amazon products in our dataset. Then, for each node (i.e., product), we compute its degree, eigenvector centrality, PageRank score, and clustering coefficient. Appendix A.1 provides the mathematical details of how we compute these measures.

The degree of a product is the total number of reviewers it has in common with other products in the network. Eigenvector centrality measures the structural importance by considering the product's proximity to other structurally important products in the network. A product's centrality score increases by having reviewers in common with other products that themselves share many reviewers with other products in the network. PageRank, like eigenvector centrality, considers the importance of a product's neighbors when assigning a score to it. However, it mainly differs from eigenvector centrality by normalizing a neighbor product's structural importance by the number of its neighbors.

The above three measures (degree, eigenvector centrality, and PageRank) help us compare products' structural importance and understand how they relate to the overall network. In addition, we measure the products' connectivity within their neighborhood using the

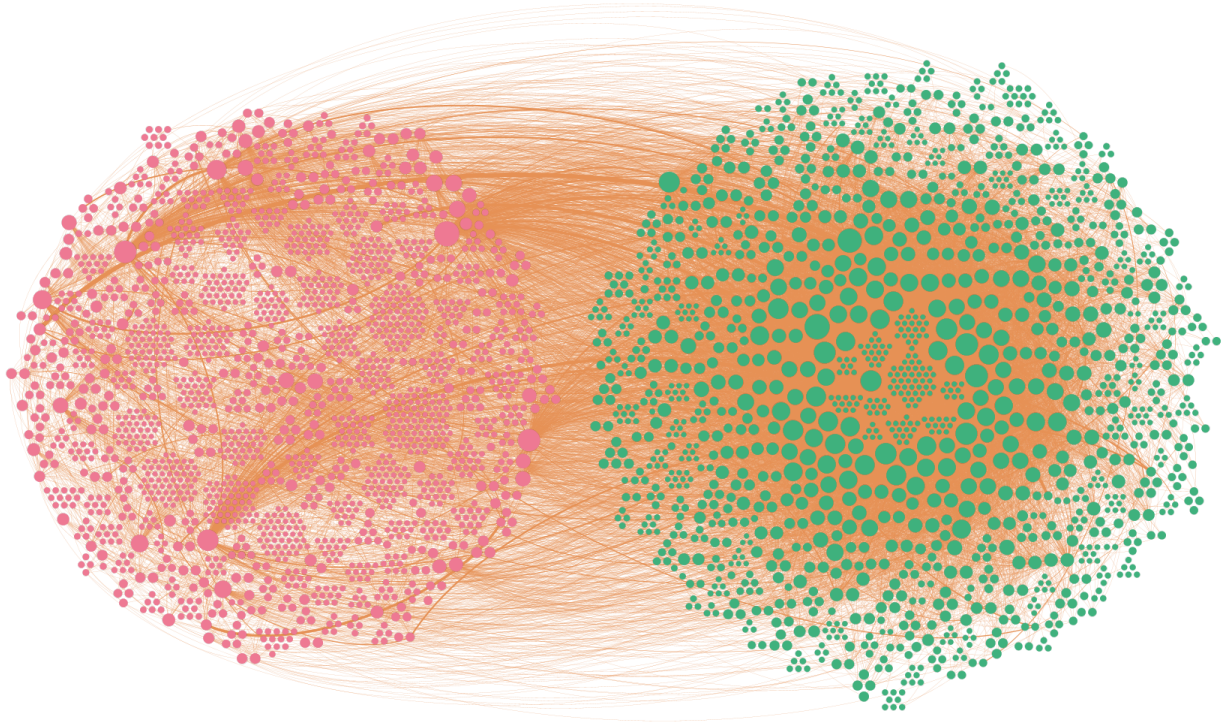

Figure 1: The network structure of Amazon products in our dataset. Green-colored nodes are the fake review buyers. We filter the edges to show only the ones between pairs of products with ≥ 2 reviewers in common. A product's size is proportional to the number of reviewers in common with other products. The figure shows that fake review buyers are mainly the larger nodes in the network and are highly clustered, highlighting the dense connectivity between fake review buyers due to their participation in the fake review marketplace.

clustering coefficient. The clustering coefficient checks all pairs of a product's neighbors and considers how many of them have reviewers in common.

3.2 Additional features

To compare the performance of different types of information at detecting fake review buyers, we also generate features from available product metadata and review content such as review text, ratings, timestamps, and images.

Metadata features The first set of features is generated from available metadata. These features include the number of reviews, average review rating, summary statistics of the time gap between reviews, the ratio of reviews that other consumers found helpful, the

ratio of reviews with 1-star review rating, the ratio of reviews with 5-star review rating, the ratio of reviews that include review images, and the average text similarity among the reviews of a product. These metadata should capture evidence of rating manipulation, such as having a disproportionate share of five-star reviews, excessive helpfulness votes, or odd timing characteristics such as long gaps in the arrival of reviews followed by the appearance of many at once.

Image features We generate a second set of product-level features using review and product images. First, we use a pre-trained deep Convolutional Neural Network (He et al., 2016) for all images (both product and review) to extract the features as shown in Figure A.1. We then calculate the angular similarity of all the images belonging to the same product using the cosine distance between the vector representation of each image. Cosine distance is particularly effective for high-dimensional data (France et al., 2012). We calculate three sets of features that summarize the degree of image similarity between all product reviews (controlling for multiple images belonging to the same review), the similarities between the seller's product images and the review images, and the similarity among all pairs of review images. For each set of features, we calculate the minimum, maximum, average, and standard deviation of the pair-wise similarities within a product.

Text features The third set of product-level features generated from the review content is text features. We first combine the review bodies of a product and treat each product as a document. Then, we calculate the TF-IDF score of words in each document. We only consider the top 1000 features that receive the highest TF-IDF scores for each product. Finally, each product is represented by a feature vector of length 1000.

Table 1 shows all the product-level features we generate to consider various aspects of reviews, products, and reviewer behaviors to detect fake review buyers.

Table 1: Product-level features generated from review content and product network.

Type	Feature	Description
Network	degree	Total number of reviewers in common with other products
	clustering coef	Clustering coefficient of the product
	eigenvector cent	Eigenvector centrality score of the product
	pagerank	PageRank score of the product
Metadata	tf-idf sim	Avg similarity of TF-IDF features b/w reviews of a product
	# reviews	Number of reviews
	avg review rating	Avg review ratings
	time b/w reviews	Avg, min, max, and stdev of time in days b/w reviews
	share helpful	Share of reviews with helpful votes
	share 1star	Share of reviews with 1-star rating
	share 5star	Share of reviews with 5-star rating
	share photo	Share of reviews with review photo
Image	img sim	Avg, min, max, and stdev of image similarity b/w product reviews
	sim review	Avg, min, max, and stdev of similarity b/w all pairs of review images
	sim product	Avg, min, max, and stdev of similarity b/w product and review images
Text	tf-idf	Top 1000 TF-IDF features of a product

3.3 Results of Supervised approach

To test the predictive performance of the features we created, we employ a set of random forest classifiers.[1] Following the standard approach in the literature, we train the classifiers on a random subset of 80% of the products and evaluate its prediction performance on the remaining 20%. We measure the area under the receiver operating characteristic curve (AUC), classification accuracy, true positive rate (TPR), true negative rate (TNR), and the F1 score. The technical details of our estimated model and model building process are in Appendix B.

Table 2 shows the performance of each type of feature set. We find that features constructed from the product-reviewer network outperform metadata, text, and image features across all accuracy metrics. The combined model that includes all features performs only slightly better than the model using only network features. In addition to performing best on balanced measures of accuracy, the network feature model performs best on the true positive rate. This is especially important for a rating platform for whom avoiding false positives is

[1] We also test other classifiers and obtain qualitatively similar results, which can be found in Appendix C.

an important goal.

Table 2: Out-of-sample prediction performance of the random forests classifier with varying sets of features.

Features	AUC	Accuracy	TNR	TPR	F1 Score
Network	**.890**	**.821**	.839	**.797**	**.821**
Top-2 Network	.879	.812	.832	.787	.812
Metadata	.868	.791	.834	.734	.791
Text	.857	.770	**.929**	.559	.759
Image	.592	.599	.792	.343	.577
All Features	.933	.859	.879	.832	.859

Figure 2 shows the relative importance of different individual features from the all-feature model in terms of their contribution to predictive power. The two most important features by a large margin are network features, specifically the clustering coefficient and the eigenvector centrality score. The fact that these two features rank highest in importance highlights the reason why the network-based model performs so well. Products buying fake reviews rely on a common set of reviewers, causing them to be more closely clustered in the product network compared to regular products. Table 2 shows that when we test a more concise model using only these top two network features, it still outperforms all other models. This suggests that the predictive power of network features is sufficiently strong that, even when implemented in a simple way, they are more useful than models based on large sets of features derived from metadata, text, or images.

3.4 External validity using unsupervised approach

While we have shown that a model trained on features derived from the product network is highly accurate at detecting fake review buyers, platforms may not have the ground truth data required to estimate this type of model. Therefore, to further validate the strength of a network-based approach, we extend our analysis to a much larger dataset of Amazon product reviews where we lack ground truth data and test whether network data allow us to identify ex ante which products are likely to be manipulating their ratings.

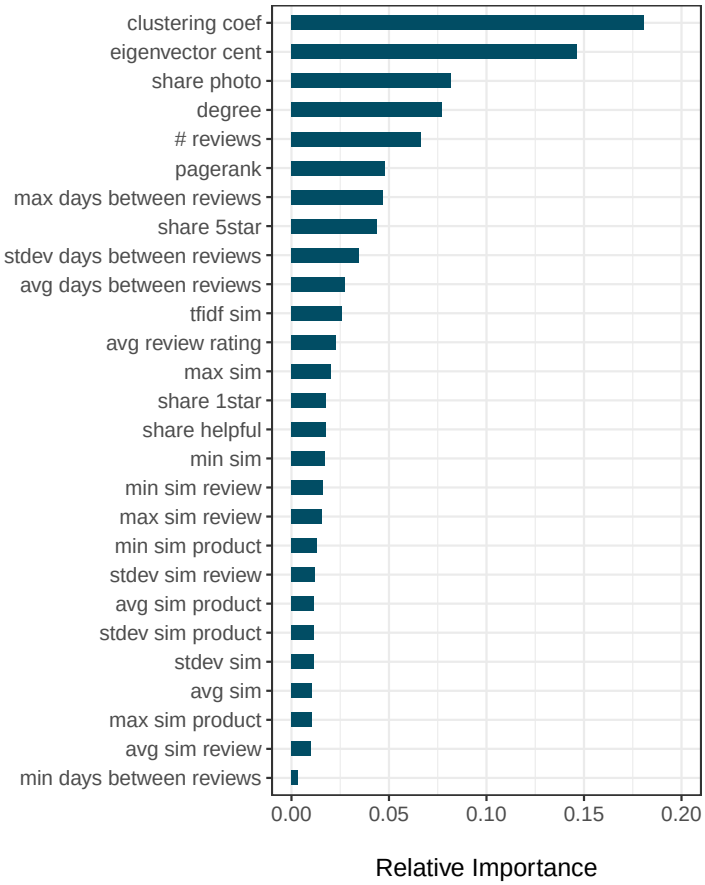

Figure 2: Relative importance of features in terms of their contribution to prediction performance of the random forests classifier including all features.

We use data that contains the entire universe of Amazon product reviews in the Home & Kitchen category, which contains about 65 thousand products, 11 million reviews, and 6.1 million reviewers (Ni et al., 2019). Because we no longer know which products buy fake reviews in this larger dataset, we follow an unsupervised approach. We start by creating a product network in the same manner as in Section 3.1. We then partition the products into 20 groups using the K-means clustering algorithm based on products' metadata and network features (Fraley & Raftery, 2006). Once the products are clustered, we test if certain clusters are disproportionately likely to contain fake review products. To test this, we use our pre-trained classifier that was trained on all features to estimate the proportion of products buying fake reviews in each cluster. Although this cannot provide an exact measure of accuracy, the all-features classifier was able to identify fake review buyers with

93.5% out-of-sample accuracy.

Figure 3 shows the percentage and the total number of products identified as buying fake reviews in each cluster using our classifier. We report summary statistics of clusters and the details of the unsupervised approach in Appendix D.

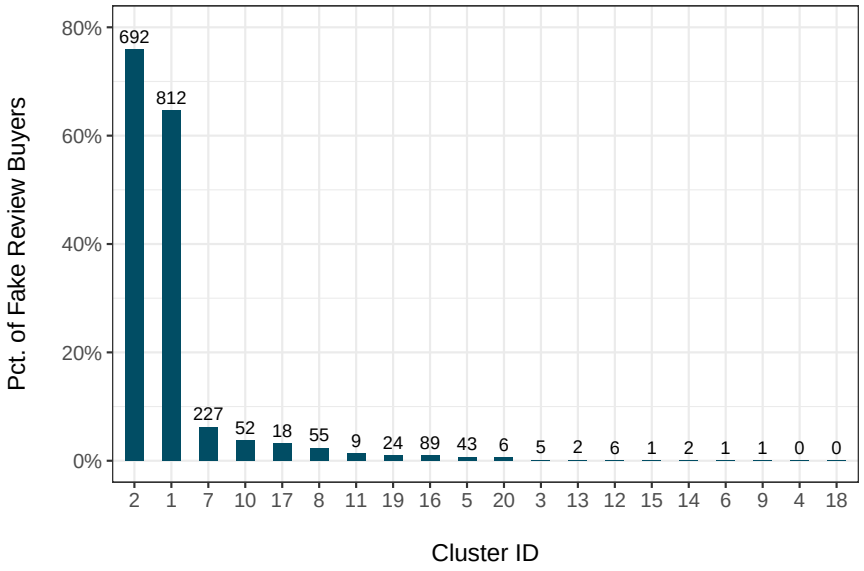

Figure 3: The percentage of products that are identified as fake review buyers by the random forest classifier in each cluster.

This analysis shows that just two clusters (containing only 3.3% of products) contain 74% of the products identified as fake review buyers. In addition, within these two clusters, a substantial majority of products are identified as buying fake reviews (64% and 76%). Table D.1 shows the mean features in each cluster. The two clusters that contain the majority of fake reviews are highly distinctive, and only in terms of their network features. This suggests that a platform could use unsupervised methods to identify these tightly-connected clusters of products without any ground truth data and use this information to identify likely fake review buyers.

4 Discussion and Conclusions

In this paper, we use a unique dataset and method to address a longstanding problem in the digital economy: rating manipulation. We use direct observation of what sellers buy fake reviews and propose a network-based approach to detect these products.

Our analysis of the product network shows that products that buy fake reviews are highly clustered due to the fact that, compared to the full universe of Amazon reviewers, a relatively small number of reviewers participate in the fake review marketplace. Because of this, a classifier based on just two network features, clustering coefficient and eigenvector centrality, can identify products buying fake reviews with high accuracy, outperforming models trained on large numbers of features constructed from metadata, review characteristics, text, and images.

In addition to being powerful features to detect products that buy fake reviews, a crucial implication of this insight is that network features cannot easily be manipulated, making network-based detection methods robust to manipulation. While the specific setting we study is Amazon.com, the disproportionate clustering of rating manipulators in the network structure is likely to hold across applications where fake reviews are found. This is because it will generally be the case that the set of reviewers used by rating manipulators is likely to be substantially smaller than the general population of reviewers, leading to the same pattern of unusual clustering observed in our network. Moreover, since we show that no ground truth data is necessary to identify these clusters, doing so for platform firms should be easily achievable. While no method can identify fake reviews with 100% accuracy, this would allow rating platforms to apply greater scrutiny to these products, add warning flags for customers, or otherwise selectively reduce their incentive to manipulate their ratings. Overall, our results suggest that the product network structure provides a robust source of information for identifying and regulating the products buying fake reviews.

References

Akesson, J., Hahn, R. W., Metcalfe, R. D., & Monti-Nussbaum, M. (2022). The impact of fake reviews on demand and welfare. *Working Paper*.

Burtch, G., Hong, Y., Bapna, R., & Griskevicius, V. (2018). Stimulating Online Reviews by Combining Financial Incentives and Social Norms. *Management Science*, *64*(5), 2065–2082. https://doi.org/mnsc.2016.2715

Crawford, M., Khoshgoftaar, T. M., Prusa, J. D., Richter, A. N., & Najada, H. A. (2015). Survey of review spam detection using machine learning techniques. *Journal of Big Data*, *2*, 1–24.

Dong, B., Li, M., & Sivakumar, K. (2019). Online review characteristics and trust: A cross-country examination. *Decision Science*, *50*(3), 537–566.

Fraley, C., & Raftery, A. E. (2006). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association*, *97*(458), 611–631.

France, S. L., Carroll, J. D., & Xiong, H. (2012). Distance metrics for high dimensional nearest neighborhood recovery: Compression and normalization. *Information Sciences*, *184*(1), 92–110.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, S., Hollenbeck, B., & Proserpio, D. (2022). The market for fake reviews. *Marketing Science*.

Heydari, A., ali Tavakoli, M., Salim, N., & Heydari, Z. (2015). Detection of review spam: A survey. *Expert Syst. Appl.*, *42*, 3634–3642.

Jackson, M. O. (2010). Social and economic networks. Princeton University Press.

Molm, L., Hardin, R., & Levi, M. (2006). Cooperation without trust? *Administrative Science Quarterly - ADMIN SCI QUART*, *51*, 305–307. https://doi.org/10.2189/asqu.51.2.305

Mukherjee, A., Venkataraman, V., Liu, B., & Glance, N. S. (2013). What yelp fake review filter might be doing? *ICWSM*.

Ni, J., Li, J., & McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 188–197.

Tadelis, S. (2016). Reputation and feedback systems in online platform markets. *Annual Review of Economics*, *8*, 321–340.

Vidanagama, D. U., Silva, T. P., & Karunananda, A. S. (2019). Deceptive consumer review detection: A survey. *Artificial Intelligence Review*, *53*, 1323–1352.

Wu, Y., Ngai, E. W., Wu, P., & Wu, C. (2020). Fake online reviews: Literature review, synthesis, and directions for future research. *Decision Support Systems*, *132*, 113280.

Appendix A Details of features generation

A.1 Network features

In this section, we describe how we compute the network features. The degree of a product is calculated as follows:

$$d_i = \sum_{j \in N_i} r_{ij} [\mathbf{A}]_{ij}, \tag{1}$$

where d_i is the degree of product i, N_i is the set of products that have reviewers in common with product i, r_{ij} is the total number of reviewers that products i and j have in common, and $\mathbf{A}$ is the adjacency matrix of the network. The i, jth element of $\mathbf{A}$ (i.e., $[\mathbf{A}]_{ij}$) is 1 if products i and j have at least one reviewer in common, 0 otherwise. Notice that because the network is undirected with no self-loops, $[\mathbf{A}]_{ij} = [\mathbf{A}]_{ji}$ and $[\mathbf{A}]_{ii} = 0$.

We calculate the eigenvector centrality of a product as follows:

$$e_i = \lambda_1^{-1} \sum_{j \in N_i} [\mathbf{A}]_{ij} e_j, \tag{2}$$

where e_i is the eigenvector centrality of product i and λ_1 is the largest eigenvalue of $\mathbf{A}$. With Equation (2), a product's centrality score increases by having reviewers in common with other products that themselves share many reviewers with other products in the network.

PageRank is an extension of eigenvector centrality (Jackson, 2010), which we calculate as follows:

$$p_i = \frac{1 - \alpha}{n} + \alpha \sum_{j \in N_i} [\mathbf{A}]_{ij} \frac{p_j}{|N_i|}, \tag{3}$$

where p_i is the PageRank of product i, α is the damping factor, n is the total number of products in the network, and $|\cdot|$ is the cardinality of a set. PageRank mainly differs from eigenvector centrality by normalizing a neighbor product's structural importance by the number of its neighbors.

We calculate the clustering coefficient of a product as follows:

$$c_i = \frac{2 \sum_{j,k \in N_i} [\mathbf{A}]_{ij}}{|N_i|(|N_i| - 1)}, \tag{4}$$

where c_i is the clustering coefficient of product i. The clustering coefficient takes values between 0 and 1.

A.2 Computation of TF-IDF features

We calculate the TF-IDF of a word in a product review as follows:

$$tfidf(w, r, R) = tf(w, r) + idf(w, R), \tag{5}$$

where w is the word, r is the product review, R is the set of a product's all reviews,

$$tf(w, r) = \frac{f_{w,r}}{\sum_{k \in r} f_{k,r}}, \tag{6}$$

and

$$idf(w, R) = \log \left(\frac{|R|}{|\{r \in R : w \in r\}|} \right). \tag{7}$$

In Equation (6), $f_{w,r}$ is the frequency of word w in review r and the denominator represents the total number of words in a review. According to Equation (7), a word that frequently appears across a product's reviews will get a low score.

A.3 Image features

Figure A.1 visualizes the process of calculating the similarity between pairs of images. First, we extract features using the ResNet-152, a deep Convolutional Neural Network (CNN) pre-trained on the ImageNet database (He et al., 2016). Next, we use the output of the last fully connected layer to get a 2048 dimensional vector representation of each image. This

layer roughly represents information related to objects, patterns, and colors. It structurally processes each image the same way, which allows us to compare them.

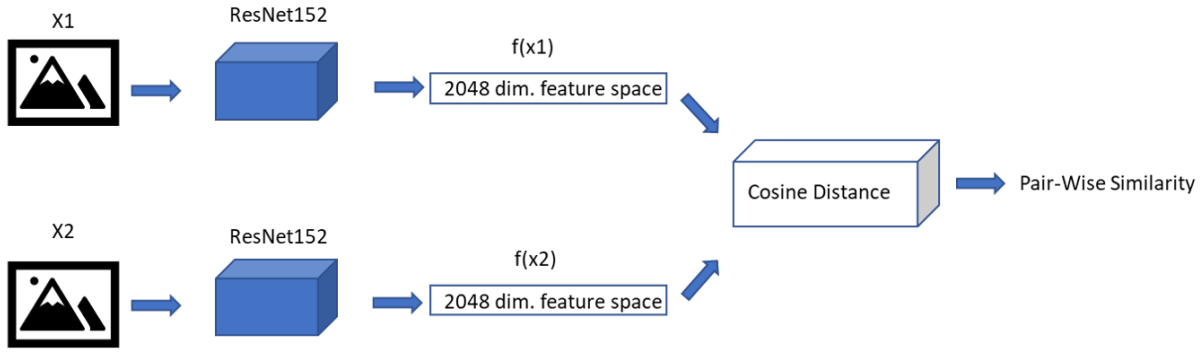

Figure A.1: Illustration of the process of calculating the pair-wise similarity between two images. 1) Extract a feature representation of each image using a pre-trained CNN (ResNet152). 2) Calculate the cosine distance between the two feature representations to obtain the similarity. The shorter the distance the higher the similarity.

Appendix B Details of classification model

A random forests classifier is an ensemble model that uses a multitude of decision trees as base learners. A decision tree recursively splits the input space into nonoverlapping regions by minimizing the total variance across classes. Each tree is built on bootstrapped training observations and randomly selected input features and assigns an observation to the most commonly occurring class in the region it belongs. Then, the class of an observation is decided by the majority vote of trees. Below, we detail the model building process.

Parameter tuning Random forests classifier has some parameters that need to be tuned. Table B.1 shows the tuned parameters and the values we use in our analysis. Before we test the model's performance, we tune these parameters. We start with random search cross-validation (CV) approach. Random search CV allows us to try a wide range of parameter values without computing all the combinations. In this step, we try [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000] for n_estimators, [1, 2, 4] for min_samples_leaf, [2, 5, 10] for min_samples_split, ['auto', 'sqrt'] for max_features, [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None] for max_depth, and [True, False] for bootstrap. After 100 iterations of random search with 5-fold CV (i.e., total 500 different parameter combinations), the model performed the best with n_estimator=200, min_samples_leaf=4, min_samples_split=2, max_features=auto, max_depth=20, and bootstrap=True. Using the results of the random search CV, we narrow the range of parameter values and follow the grid search approach with a 5-fold CV. Unlike random search CV, grid search evaluates all parameter combinations we define. In this step, we try [100, 200, 300] for n_estimators, [3, 4, 5] for min_samples_leaf, [2, 4, 6] for min_samples_split, and [10, 20, 30, 40] for max_depth. After evaluating all combinations, the values we use are shown in Table B.1.

Performance measures To evaluate the model's performance, we train it using the randomly selected 80% of the products and test it on the remaining 20%. Before training the

Table B.1: Hyperparameters of random forests classifier.

Hyperparameter	Description	Value used
n_estimators	The number of decision trees to use as base learners	100
min_samples_leaf	The min. number of data points required in a leaf node	3
min_samples_split	The min. number of data points required before splitting a node	6
max_features	The max. number of features considered to split a node	auto
max_depth	The max. number of levels allowed in a decision tree	40
bootstrap	Whether to use bootstrapped samples when building trees	True

model, we standardize the features by removing the mean and scaling it to the unit variance to avoid the effect of varying scales of features in the dataset. We then report the model's AUC, accuracy, and F1 score on the test set. The receiver operating characteristic (ROC) curve shows the prediction performance of a classifier for all classification thresholds. It plots two values: true-positive rate (TPR) and false-positive rate (FPR). TPR (also known as recall or sensitivity) is the ratio of true positive observations to actual positive observations, and FPR is the ratio of true negative observations to actual negative observations. AUC measures the total area under the ROC curve. Accuracy is the percentage of observations that are correctly classified. F1 score is the harmonic mean of precision and recall, where precision is the ratio of true positive observations to all positive predicted observations. The higher the AUC, Accuracy, and F1 score values, the better the classification performance.

Appendix C Additional Classifiers

To further validate our analysis, we test other classification models and obtain qualitatively similar results. The additional models include logistic regression, support vector machine (SVM), and extreme gradient boosting (XGBoost).

A logistic regression classifier models the relationship between the probability that a seller is a fake review buyer and input features using a logistic function. The goal is to estimate the coefficients that are associated with input features. We use maximum likelihood to estimate the coefficients of the model. The SVM classifier obtains a $(p-1)$ dimensional hyperplane that separates the classes of training observations, where p is the total number of features in the dataset. We use linear SVM with a regularization parameter of 1.0. XGBoost, similar to random forests, is also an ensemble learning model that uses decision trees as base learners to make a prediction. XGBoost uses a gradient descent algorithm to train the model. Like random forests, the XGBoost classifier has hyperparameters that need to be provided, such as the number of decision trees, the learning rate, the maximum depth for each tree, and regularization parameters. We use the default parameter values provided by the Python library of XGBoost in our analysis.

Table C.1 shows the prediction performance of different classifiers with different sets of information. Results are aligned with the results of random forests classifier.

Table C.1: Comparisons of fake review buyer detection performance of additional classifiers with varying sets of features.

Model	Metric	Features					
		Image	Text	Metadata	Top-2 Network	Network	All Features
Log. Reg.	AUC	0.631	0.769	0.838	0.855	0.871	0.922
	Acc	0.620	0.731	0.790	0.793	0.802	0.851
	TNR	0.842	0.774	0.850	0.874	0.887	0.892
	TPR	0.325	0.675	0.710	0.685	0.689	0.797
	F1	0.591	0.731	0.788	0.790	0.799	0.851
SVM	AUC	0.629	0.749	0.839	0.854	0.874	0.921
	Acc	0.593	0.707	0.778	0.799	0.796	0.851
	TNR	0.900	0.750	0.845	0.855	0.911	0.892
	TPR	0.185	0.650	0.689	0.724	0.643	0.797
	F1	0.529	0.707	0.776	0.797	0.790	0.851
XGBoost	AUC	0.578	0.856	0.872	0.873	0.881	0.935
	Acc	0.571	0.773	0.802	0.809	0.817	0.869
	TNR	0.726	0.863	0.821	0.821	0.834	0.908
	TPR	0.364	0.654	0.776	0.794	0.794	0.818
	F1	0.557	0.770	0.802	0.810	0.817	0.869

Appendix D Details of unsupervised approach

To apply the insights from our analysis on the smaller Amazon dataset to the larger one, we first create a product network. The network has about 65 thousand nodes (i.e., products) and 16.2 million edges (i.e., existence of shared reviewers). Using the product network and review content, we then obtain the network features (degree, pagerank, eigenvector centrality, and clustering coefficient) and metadata features of the products in the larger dataset.

After obtaining the network and metadata features, we partition the products into 20 clusters using the K-means clustering algorithm. K-means is an unsupervised machine learning model that does not require ground truth labels and partitions the observations in a dataset into K distinct and non-overlapping clusters. To do so, the K-means clustering algorithm minimizes the total within-cluster variation over all clusters. Within-cluster variation of a cluster is the average of squared distances between pairs of observations in the cluster. The steps of the K-means clustering algorithm are summarized as follows:

1. Randomly assign products into one of K clusters.
2. Iterate until there is no changes.
    (a) Calculate the centroids of all clusters.
    (b) Assign the products to the cluster whose centroid is the closest.

We use Euclidean distance to measure the distances between pairs of observations in our analysis. To avoid the effect of varying scales of features, we standardize them by removing the mean and scaling to unit variance.

Table D.1: Average of products' feature values in each cluster. The values of clusters are standardized. Features are listed from left to right in the order of their importance in predicting fake review buyers according to the random forests classifier.

Cluster ID	Clust. Coef.	Eig. Cent.	Share Photo	Degree	# Rev.	Page Rank	Max Days	Share 5star	Std. Days	Avg. Days	Tfidf Sim.	Avg. Rating	Share 1star	Share Helpful	Min Days
1	1.76	1.61	-0.04	1.33	-0.31	0.47	-0.44	0.05	-0.46	-0.56	1.42	0.30	-0.43	-0.04	-0.36
2	1.76	2.66	-0.32	3.05	2.98	1.10	-0.39	0.54	-0.76	-1.18	0.10	0.56	-0.56	-0.86	-0.36
3	-0.53	-0.19	-0.04	-0.47	-0.30	-0.61	-0.43	0.37	-0.43	-0.35	0.46	0.38	-0.39	1.66	-0.36
4	-0.03	0.41	-0.18	2.01	2.29	-0.59	-0.45	0.55	-0.77	-1.16	0.10	0.54	-0.49	-0.70	-0.36
5	-0.35	-0.64	-0.43	-0.47	-0.57	-0.88	-0.22	1.16	-0.08	0.26	-0.74	1.01	-0.86	-0.67	-0.36
6	-0.40	-0.34	-0.47	-0.51	-0.53	-0.59	0.07	-0.78	0.24	0.59	-0.62	-0.57	0.31	-0.12	-0.36
7	-0.34	-0.34	4.10	-0.39	-0.33	-0.65	-0.59	0.65	-0.63	-0.69	-0.38	0.58	-0.51	-0.19	-0.34
8	-0.44	-0.94	-0.07	-0.55	-0.65	-0.29	-0.37	0.55	-0.19	0.43	-0.98	0.51	-0.50	-0.35	2.39
9	-0.56	-0.79	0.00	-0.36	-0.39	-0.99	-0.55	-0.02	-0.57	-0.58	-0.86	0.08	-0.21	-0.62	-0.36
10	3.10	-1.39	0.18	-0.48	-0.61	0.32	-0.39	0.36	-0.27	-0.08	-0.50	0.34	-0.31	-0.26	-0.28
11	-0.04	0.56	-0.79	-0.54	-0.60	1.96	1.27	-0.76	1.84	2.21	2.14	-0.57	0.46	2.54	-0.21
12	-0.56	-0.34	-0.11	0.62	0.67	-0.88	-0.50	0.98	-0.70	-0.98	-0.02	0.88	-0.76	-0.72	-0.36
13	-0.48	0.11	-0.11	-0.54	-0.37	0.17	-0.26	-2.66	-0.21	-0.10	-0.50	-3.09	3.46	-0.07	-0.19
14	-0.50	-0.34	0.22	-0.47	-0.38	-0.76	-0.53	-1.39	-0.54	-0.55	-0.98	-1.34	1.16	-0.40	-0.35
15	0.01	0.56	-0.57	-0.44	-0.23	0.49	-0.12	-1.17	-0.07	0.09	2.74	-1.04	0.93	2.32	-0.29
16	-0.56	-0.79	0.07	-0.33	-0.34	-0.99	-0.60	1.46	-0.64	-0.70	-0.38	1.20	-0.94	-0.88	-0.36
17	-0.38	1.31	-0.22	-0.54	-0.23	2.47	3.66	0.30	3.12	1.41	-0.38	0.25	-0.21	-0.34	-0.14
18	-0.53	0.41	-0.22	0.13	1.02	-0.52	-0.47	-0.79	-0.71	-1.02	-0.02	-0.67	0.49	-0.49	-0.36
19	-0.33	-0.34	-0.43	-0.50	-0.46	-0.33	0.96	0.81	1.05	1.06	-0.26	0.77	-0.69	-0.27	-0.36
20	-0.59	-1.24	-0.57	-0.56	-0.67	1.10	0.34	-0.21	0.78	1.89	-0.26	-0.13	0.06	0.47	3.37