INSEAD The Business School for the World®

Working Paper

2022/26/MKT

Artificially Intelligent Marketplaces

Ruiqi Lin,
Sciences Po
ruiqi.lin@sciencespo.fr

Pavel Kireyev
INSEAD, pavel.kireyev@insead.edu

Peer-to-peer marketplaces have exploded in popularity in the past decade. Companies and individuals have deployed bots into these marketplaces to generate profits by purchasing and reselling items. However, the profitability of such bots remains uninvestigated. We develop a framework for back-testing trading bots in peer-to-peer marketplaces. The framework infers outcomes for counterfactual actions taken by the bot and uses data on what actually happened in the marketplace to assess the bot's profitability out-of-sample. Hyperparameters allow developers to fine-tune bot strategies and high-dimensional variable selection models help isolate the effects of bot actions from a large number of confounders. We apply the framework to data from the CryptoPunks NFT marketplace which uses a combination of bidding and buy-it-now prices. We show that a strategy that "spams" low bids to a large number of listings can be profitable with a return-on-investment of 10.24% over 4 months. Bots that buy at buy-it-now prices struggle to generate significant profits. Developers can also deploy seller-rewarding bots that pay for themselves and stimulate the market by improving seller revenues. Our framework helps bot developers test different strategies and points to the feasibility and limitations of artificially intelligent peer-to-peer marketplaces in which bots participate together with humans.

Keywords: Marketplaces; Platforms; Artificial Intelligence; Bots; Back-Testing; Pricing; Nonfungible Tokens; Machine Learning

Acknowledgments: We thank Shen Ting Ang, Vineet Jain, and the INSEAD Advanced Analytics Team for helpful research assistance.

Introduction

Over the past decade, peer-to-peer marketplaces have become ubiquitous. Marketplaces like eBay and Mercari enabled participants to buy and sell predominantly physical goods. However, with the recent emergence of Nonfungible Tokens (NFTs) like CryptoPunks and the Bored Apes Yacht Club, marketplaces for entirely digital goods have grown in popularity. Even before, digital items have been traded in marketplaces for online games like Second Life and World of Warcraft, although these economies are closed ecosystems which do not allow participants to transfer items outside of the application, limiting their scope and growth potential.

Some marketplace participants have attempted to implement bots to buy and sell items with the goal of generating a profit. For example, players in online games developed bots that purchased cheap items as soon as they were listed by another player. NFT platforms often involve bots that attempt to quickly buy items that were listed for very low prices by accident. Furthermore, some physical goods marketplaces have deployed such bots but with mixed success. Real-estate marketplace Zillow introduced a program called iBuying,[1] where they algorithmically purchased homes and attempted to resell them at higher prices. This algorithm underperformed and was withdrawn.[2] Zillow attributed this partially to uncertainty in the housing market induced by the COVID-19 pandemic. Blockchain platform Forte, which raised $725m in funding, plans to use bots as their main source of revenue by buying and reselling game items.[3]

Despite the growing potential of trading bots, especially as marketplaces increasingly focus on digital goods with limited overhead (e.g., zero shipping costs), limited evidence exists of bot profitability. Furthermore, no frameworks exist for back-testing the performance of different bots, which limits the ability of both researchers and developers to understand their potential. We contribute by developing a framework for back-testing trading bots in peer-to-peer marketplaces and test the performance of several

[1] https://www.zillow.com/sellers-guide/what-is-an-ibuyer/
[2] https://www.wired.com/story/zillow-ibuyer-real-estate/
[3] https://venturebeat.com/2021/11/12/forte-raises-725m-for-compliant-blockchain-gaming-platform/

types of bots using data from the Cryptopunks NFT marketplace. Our results show that bidding bots that "spam" a large number of small bids and resell at higher prices can be profitable, while buy-it-now bots that purchase items at listing prices struggle to make significant profits. We also show that marketplaces can deploy seller-rewarding bots that "pay for themselves" and stimulate the market by increasing seller revenues. Our analysis explains why, anecdotally, most bots in NFT marketplaces like OpenSea place very low bids. We also point out how challenging it is for bots to make profits using other strategies, which may explain the failure of tools like Zillow's iBuying.

Our framework draws inspiration from back-testing approaches for liquid assets like stocks or cryptocurrencies. Liquid asset trading bots can be back-tested by using the time series of an asset's price (such as the stock price of Tesla) and studying how different buy/sell decisions made by the bot would lead to profit/loss over time. These buy/sell decisions can be functions of past data. Back-testing works in these contexts as the markets are liquid, with a "market" price available at any point in time for each asset. It is usually assumed that bots are small actors and cannot influence supply or market prices with their actions. However, items in peer-to-peer marketplaces are highly illiquid. As a result, it is not possible to obtain a "market" price at which an item is guaranteed to sell at any point in time for each item. Rather, sellers make pricing decisions. Nevertheless, it would be desirable to use data on realized transactions to inform the out-of-sample performance of a bot, even if the bot sets different prices than observed in the data.

Our framework involves two steps. First, we estimate "full-data" models of the relationships between listing-level outcomes (like sales incidence and bid acceptance decisions) and bot actions (like bid amounts or prices set) using the entire dataset. We use high-dimensional variable selection models from the "causal" machine learning literature to identify the effects of counterfactual bot actions while accounting for a large set of confounders. Second, we estimate statistical models of these decisions using an expanding window of data, and generate optimal bot decisions for the subsequent day using these "limited-data" models. The quality of out-of-sample bot decisions is evaluated using probabilities

obtained from Bayes' rule, combining data on actual outcomes with counterfactual probabilities predicted from the "full-data" models estimated in step one. Thereby, we generate out-of-sample bot performance statistics for counterfactual actions not observed in the data. Our framework uses hyperparameters that developers can tune to improve bot performance or achieve specific objectives. While our framework can suffer from misspecification in the "full-data" models used to infer counterfactual probabilities out-of-sample, it is nevertheless the best available option in peer-to-peer marketplaces given the impossibility of obtaining market prices with certainty.

Literature

Most research on peer-to-peer marketplaces has focused on market design (Lucking-Reiley 1999, Bajari and Hortaçsu 2004, Wang et al. 2008, Yao and Mela 2008, Bauner 2015, Choi and Mela 2019) but not the role of bots. The decision to introduce a bot into a marketplace can be viewed as a previously unexplored market design lever. We study the profitability of bots both from the perspective of a developer as well as the market designer.

Pricing research has focused on firms selling items to consumers (Kannan et al. 2009, Caro and Gallien 2012, Ferreira et al. 2016, Dubé et al. 2017, Dubé and Misra 2019, Amanoet al. 2022) where the main goal involves setting a price (or discount) to optimize the seller's objective. We study a multi-sided market that involves both bidding and listing prices. The bot we develop uses novel strategies that combine bidding and pricing actions to generate a profit for itself. This brings us closer to literature on back-testing trading strategies for liquid assets like stocks and cryptocurrencies (Ehling et al. 2018, Fang et al. 2022). Our work can be viewed as bridging these two literatures.

In addition to the emerging research on blockchains and cryptocurrencies (Halaburda et al. 2022), research on NFT markets has started to emerge (Nadini et al. 2021, Kireyev and Lin 2021; Vasan et al. 2022, Kireyev 2022) but still remains sparse. Researchers have studied returns on NFTs from the perspective of human investors (Borri et al. 2022, Oh et al. 2022). We contribute by highlighting the

feasibility of deploying bots in NFT marketplaces leading to greater returns and increasingly automated markets.

Data

We use data from the CryptoPunks marketplace, which features 10,000 unique NFTs, each associated with a pixelated image of a "punk" (Figure 1). The NFTs represent blockchain-based digital certificates of ownership, which can be viewed as digital items that users can trade. See Kireyev and Lin (2021) for more on NFTs and the underlying technology. The following discussion mirrors Kireyev (2022) who provides additional information on the CryptoPunks market.

Figure 1: CryptoPunk NFTs

In 2017, all CryptoPunks were offered to buyers for free by their developer. Further transactions occurred between participants, making this a peer-to-peer marketplace. CryptoPunks are one of the most popular collections, generating $2.2b in transaction volume as of May 2022. A rare "alien" CryptoPunk sold for ~$12m through Christie's auction house in 2021. Many owners use their CryptoPunk as a profile picture on Twitter. The marketplace allows sellers to list any CryptoPunk they hold while specifying a listing price. Prices are in Ether (ETH) and participants must have an Ethereum wallet to participate. The marketplace also allows bidders to place bids on active listings. The seller can choose to accept or reject any bid but is not obliged to sell to any bidder. This is in contrast to most eBay auctions where sellers

may specify an initial "buy-it-now" price but must accept the highest bid if bidding begins and exceeds the reserve price.

Table 1 presents summary statistics for our sample, which includes 50,556 listings for 5,762 NFTs made by 2,677 sellers and accounts for $871,701,419 of transaction volume from June 2017 (market instantiation) until August 2021.

Table 1: Summary Statistics

	Min	Median	Mean	Max
NFT Characteristics				
Token ID	1	5,604	5,444	9,998
Male			0.62	
Female			0.37	
Rare Type			0.01	
Number of Extra Attributes	1	4	3.79	8
log(Rarity)	-23.53	-9.58	-9.63	-0.50
Number of Listings per Token	1	6	8.77	88
Listing Characteristics				
Listing Date	2017-06-23	2021-04-12	2021-03-04	2021-08-27
Listing Price (ETH)	0.03	31	296.86	5,100,000
Bid Placed			0.16	
Number of Bidders	1	1	1.31	19
Maximum Bid (ETH)	<0.01	20	32.92	2,200
Sold for Listing Price			0.21	
Sale Price (ETH)	0.03	22	30.67	4,200
Hours Until Sale	<0.01	20	420.43	32,553
Bid Accepted			0.02	
Accepted Bid Amount (ETH)	<0.01	20	30.53	667
Hours Until Accepted Bid	0.06	15	237.92	32,057
Seller Characteristics				
Number of Listings per Seller	1	10	18.89	936
Number of Listings	50,556			
Number of Sellers	2,677			
Number of Bids	10,747			
Number of Top Bidders	1,929			
Number of Buyers	3,349			
Number of Tokens	5,762			
Transaction Volume	363,019 ETH (≈$871,701,419)			

Note: An observation is a listing.

Focusing on NFT characteristics, the vast majority of listings involve Male or Female Cryptopunks as these are most common. About 1% of listings involve a rare (Alien, Ape, or Zombie) CryptoPunk which will typically attract a higher price. CryptoPunks possess up to 8 additional attributes which affect their appearance (hats, earrings, beards, etc.) and rarity, which we calculate for each NFT as the probability of

randomly drawing its attributes from the attribute pool. We summarize this variable in the row "log(Rarity)." NFTs with a lower log(Rarity), indicating a lower-probability attribute set, should attract higher prices. The NFT's attributes exclusively determine its appearance, limiting concerns about omitted appearance-related variables. Each NFT is listed 8.77 times on average. As we only observe 5,762 tokens, the remaining 4,238 tokens were not listed.

Focusing on listing characteristics, we notice a very large dispersion in listing prices. This is primarily driven by differences over time and outlier listings. CryptoPunks used to sell for very small amounts early on but experienced a price increase in 2021 as NFTs gained popularity. We present time trends in Web Appendix A. The median listing price was 31 ETH which was ~$124,000 as of December 2021. While the ETH/USD exchange rate is volatile, we observe a 98% correlation between listing prices in ETH and USD. We use ETH throughout as it is the native marketplace currency.

Regarding bidding behavior, 16% of the listings are bid on. The median number of bidders is 1. About 21% of listings sell for the listing price, and 2% sell for the bid amount, yielding a total sales rate of 23%. The median sale price is 22 ETH and the median accepted bid is 20 ETH. The median time until a listing price sale occurs is 20 hours after listing creation. The median time until bid acceptance is 15 hours after listing creation, suggesting that bids are usually placed and accepted early relative to when sales occur.

We construct 106 control variables (excluding time fixed effects) that help explain listing-level outcomes beyond listing price. These include competitor-related variables such as the number of open listings at the time when the focal listing was created, dummy variables for each possible discrete NFT attribute, functions of attribute rarity, and more (see Table A.3).

Framework

Our framework involves two steps. First, we estimate "full-data" models of the relationships between listing-level outcomes (like sales incidence and bid acceptance decisions) and bot actions (like bid amounts or prices set) using the entire dataset. Second, we estimate statistical models of these decisions

using an expanding window of data, and generate bot decisions for the subsequent day based on these "limited-data" models. The quality of out-of-sample bot decisions is evaluated using probabilities obtained from Bayes' rule, combining data on actual outcomes with counterfactual probabilities predicted from the "full-data" models estimated in step one.

We require an objective function, which we specify as the expected profits of the bot. The expected profits of the bot for listing j at time t are

$$\pi_{jt}(B_{jt}, P_{jt}) = \Pr(A_{jt} = 1 | B_{jt}) [\Pr(S_{jt} = 1 | P_{jt}) P_{jt} - B_{jt}]$$

where $A_{jt} = 1$ is the event that the bot's bid is accepted by the seller, $S_{jt} = 1$ is the event that a sale occurs (by some buyer in the market), and B_{jt} and P_{jt} are the bid placed by the bot and bot's resale price, respectively. The first term can be expressed as:

$$\Pr(A_{jt} = 1 | B_{jt}) = \left(\Pr(B_{jt}^{-} = 0) + \Pr(B_{jt}^{-} > 0) \Pr(B_{jt}^{-} < B_{jt} | B_{jt}^{-} > 0)\right) \Pr(A_{jt} = 1 | B_{jt}; B_{jt}^{-} < B_{jt})$$

where B_{jt}^{-} is the maximum bid placed by other market participants. In other words, the seller considers accepting the bot's bid if no other bid is placed $(B_{jt}^{-} = 0)$ or if another bid is placed $(B_{jt}^{-} > 0)$ but it is lower than the bot's bid $(B_{jt}^{-} < B_{jt})$. The bidding bot chooses B_{jt} and P_{jt} for each listing to maximize $\pi_{jt}(B_{jt}, P_{jt})$. The bot places a bid on a listing if $\pi_{jt}(B_{jt}^{*}, P_{jt}^{*}) > 0$, where $\{B_{jt}^{*}, P_{jt}^{*}\}$ are the optimal bid-price combination.

In the first step, we estimate models for $\Pr(S_{jt} = 1 | P_{jt})$, $\Pr(B_{jt}^{-} = 0)$, $\Pr(A_{jt} = 1 | B_{jt}; B_{jt}^{-} < B_{jt})$ and $\Pr(B_{jt}^{-} < B_{jt} | B_{jt}^{-} > 0)$ using the full data. These models will be used to infer "ground truth" probabilities for outcomes of counterfactual bot actions out-of-sample. It is important that they accurately capture the relationship between the decision variables $\{B_{jt}, P_{jt}\}$ and outcomes. We use methods from the "causal" machine learning literature to obtain as accurate an effect as possible (absent experimental variation) while accounting for a large set of confounders. Note that "causal" machine learning methods do not guarantee that the estimated effect is causal, rather they perform better than standard variable selection

techniques at identifying a causal effect if all confounders are accounted for. We believe this to be the case in our setting as the NFTs are digital goods whose characteristics we observe in their entirety, limiting concerns about omitted variables. However, this is not provable absent experimental variation, but we offer several tests to support this assumption in Web Appendix C. Other studies may rely on other techniques or (quasi-)experimental variation, if available. An important caveat of our approach is that if there is significant endogeneity of the decision variables or miss-specification in the full-data models, then the out-of-sample bot performance statistics will not be reliable.

In the second step, we estimate models for the same objects using an expanding window to simulate how a bot would use the limited data available at each time period to update its model estimates. We split the data into a "training" period $t = 0, \dots, t_1$ and an "evaluation" period $t = t_1 + 1, \dots, T$, where T is the final period. For $t > t_1$ we estimate models for $\Pr(S_{jk} = 1|P_{jk})$, $\Pr(B_{jk}^- = 0)$, $\Pr(A_{jk} = 1|B_{jk}; B_{jk}^- < B_{jk})$ and $\Pr(B_{jk}^- < B_{jk}|B_{jk}^- > 0)$ such that $k < t$. While our framework is model-agnostic, we use logistic models for the binary-outcome variables as they have the desirably property that the predicted probabilities tend to zero as prices tend to infinity (or to 1 as bids tend to infinity). More flexible models like gradient-boosted trees and random forests do not have this property, which can be limiting as it may imply infinite optimal prices. We use linear models for continuous-outcome variables and find that they fit well. The results do not change even if we use more flexible models for the continuous-outcome variable.

Incorporating "hyperparameters"

We can extend the models to allow for "hyperparameters" that can either be manually adjusted by bot developers to alter the objectives or risk-tolerance of the bot, or selected automatically to maximize bot performance out-of-sample. We introduce three hyperparameters $\{\nu^A, \nu^S, \nu^\pi\}$. The bot's profit expression can be augmented as follows:

$$\pi_{jt}\left(B_{jt}, P_{jt} | \nu^A, \nu^S\right)$$
$$= \Pr\left(A_{jt} = 1 \middle| B_{jt}\right)\left[\Pr\left(S_{jt} = 1 | P_{jt}\right) P_{jt} - B_{jt}\right] + \nu^A \Pr\left(A_{jt} = 1 \middle| B_{jt}\right)$$
$$+ \nu^S \Pr\left(S_{jt} = 1 | P_{jt}\right).$$

We can require the bot to place a bid only if $\pi_{jt}\left(B_{jt}^*, P_{jt}^* | \nu^A, \nu^S\right) > \nu^\pi$ at optimal bids/prices $\{B_{jt}^*, P_{jt}^*\}$. In other words, ν^A places a weight on the probability that the bot's bid is accepted. If it is higher, then the bot will place lower bids and will have a bias towards bidding only on listings where its bids are more likely to be accepted. Similarly, ν^S places a weight on the resale probability, and higher values will encourage the bot to set lower resale prices or bid on listings where it has a higher chance of resale. The hyperparameter ν^π is a threshold above which the bot places a bid on a listing. If this value is negative, the bot will take more risk and bid on listings where its expected profit may be negative. A positive value means that the bot will be more conservative and bid on listings where expected profits exceed zero by some amount.

In practice, the data-generating-process in future out-of-sample data may differ from the data-generating-process in-sample. Hence, it may be optimal to set non-zero hyperparameter values to encourage the bot to be either more conservative or aggressive out-of-sample to maximize the original expected profit objective (in which the hyperparameters are all zero). Developers can find the optimal values of the hyperparameters by optimizing over ν to find the optimal profit in out-of-sample data. This is akin to how machine learning algorithms use prediction performance on data not used in training to decide on optimal hyperparameters (e.g., LASSO can choose its penalty based on out-of-sample mean-squared-error model performance; xgboost can choose the number of trees based on out-of-sample prediction performance), although the purpose of hyperparameters in these models is to control model complexity, whereas in our case, we are placing explicit biases on different components of our expected profit function which directly affects bot decisions. In some cases, developers may have alternative objectives. The hyperparameters

also enable them to manually tune the model to achieve these objectives. For example, we demonstrate how developers can design a seller-rewarding bot and deploy it at no cost by adjusting ν^A in Table A.5.

Evaluating out-of-sample probabilities

After we obtain the full-data and limited-data models, as well as the optimal decisions $\{B_{jt}^*, P_{jt}^*\}$ one day ahead based on the limited-data models, we can evaluate the probabilities that form the basis of our expected profit function using Bayes' rule. This way, we incorporate information about actual outcomes in the data as opposed to only using simulated probabilities, akin to how back-testing frameworks in liquid markets uses data on realized out-of-sample market prices to assess a trading algorithm's performance. We can write the out-of-sample probability of accepting the bot's bid conditional on the observed data as

$$\Pr\left(A_{jt}=1\middle|B_{jt}^*;A_{jt}^{data},B_{jt}^{data}\right)$$

$$=\begin{cases}1 \text{ if } A_{jt}^{data}=1 \text{ and } B_{jt}^{data}\le B_{jt}^*,\\ 0 \text{ if } B_{jt}^{data}>B_{jt}^*,\\ \dfrac{\Pr\left(A_{jt}=1\middle|B_{jt}^*\right)-\Pr\left(A_{jt}=1\middle|B_{jt}^{data}\right)}{1-\Pr\left(A_{jt}=1\middle|B_{jt}^{data}\right)} \text{ if } A_{jt}^{data}=0 \text{ and } B_{jt}^{data}\le B_{jt}^*,\\ \Pr\left(A_{jt}=1\middle|B_{jt}^*\right) \text{ if } B_{jt}^{data}=\emptyset,\end{cases}$$

where the $data$ superscript indicates the observed value of the variable in the data, and all probability expressions are based on the full-data models. This probability is equal to 1 if a bid was placed and accepted in the data, and the bid amount was less than the bot's bid. It's equal to zero if a greater bid was placed in the data, regardless of whether or not it was accepted. If a smaller bid was not accepted in the data, then the bot's higher bid may still be accepted according to a probability term derived from Bayes' rule as the probability of acceptance of the bigger bid conditional on the seller not accepting the smaller bid. Finally, if no bid is placed in the data, then we must use the unconditional probability of bid acceptance. Similarly, we can write the probability of a sale conditional on the data as

$$\Pr\left(S_{jt}=1\middle|P_{jt}^{*};S_{jt}^{data},P_{jt}^{data}\right)=\begin{cases}1 \text{ if } S_{jt}^{data}=1 \text{ and } P_{jt}^{data}>P_{jt}^{*},\\ 0 \text{ if } S_{jt}^{data}=0 \text{ and } P_{jt}^{data}\leq P_{jt}^{*},\\ \dfrac{\Pr\left(S_{jt}=1\middle|P_{jt}^{*}\right)}{\Pr\left(S_{jt}=1\middle|P_{jt}^{data}\right)} \text{ if } S_{jt}^{data}=1 \text{ and } P_{jt}^{data}\leq P_{jt}^{*},\\ \dfrac{\Pr\left(S_{jt}=1\middle|P_{jt}^{*}\right)-\Pr\left(S_{jt}=1\middle|P_{jt}^{data}\right)}{1-\Pr\left(S_{jt}=1\middle|P_{jt}^{data}\right)} \text{ if } S_{jt}^{data}=0 \text{ and } P_{jt}^{data}>P_{jt}^{*},\end{cases}$$

where this probability is equal to 1 if a sale occurred in the data at a higher price and equal to 0 if no sale occurred in the data at a lower price. If a sale occurred at a lower price, then the probability of a sale occurring at the bot's higher price is derived from Bayes' rule. Similarly, if no sale occurred at a higher price in the data, then the probability of a sale occurring at the bot's lower price is also derived from Bayes' rule. We use these conditional probability expressions to incorporate out-of-sample data into the bot's profit calculation when we evaluate the quality of the optimal bid-price combinations implied by the limited-data models.

Discussion

Our framework imposes several assumptions. First, we assume that the behavior of the bot does not affect the behavior of other agents in the market. This assumption is akin to how back-testing strategies in liquid asset markets assume that bots are small actors who do not influence other agents. We believe this assumption is a reasonable simplification for our NFT market, where there's an average of 1,261 parallel listings, making it unlikely that changes induced by the bot will significantly affect other market participants. The assumption of independence between listings is often made in research (Boudreau, Lakhani and Menietti, 2016; Kireyev, 2020; Yao and Mela, 2008; Yoganarasimhan, 2013). Second, our models assume that a bid may happen (by another participant) and be accepted before the bot has a chance to place its bid, but the bot places its bid before other buyers in the market have a chance to purchase the item for its original listing price. This is motivated by the descriptive statistic that bids are usually placed and accepted earlier (median: 15 hours after listing time) compared to sales which happen later (median: 20 hours after listing time). We also assume there is only one possible bidder who is

competitive with the bot which is motivated by the descriptive statistic that the median number of bidders per listing is one. Finally, our bot has a short-term objective and does not consider the long-term possibility of reselling the NFTs it acquires later at a lower price if they fail to sell immediately. We relax this assumption in Table A.6 but find that our main results are unchanged.

Results

Table 2 shows estimates for the full information models on the entire dataset, using both post-lasso (column i) and double selection lasso (column ii) approaches. Post-lasso estimates a lasso model to select coefficients and then re-estimates an un-regularized standard model on the selected coefficients (Tibshirani 1996). This approach may exhibit biases (Belloni et al. 2014, Belloni et al. 2016) when we wish to obtain a "causal" estimate for our variable of interest. As discussed in the previous section, we wish to evaluate probabilities at counterfactual prices and bids, which is why it is important that we obtain estimates that are causal and not purely for predictive purposes (as would be the case with post-lasso). To re-iterate, double selection models do not guarantee that the resulting estimate is causal but do a better job of obtaining a causal estimate than post-lasso approaches when the set of controls includes all possible confounders. We cannot guarantee that our estimate is causal absent experimental variation but provide evidence of this in Web Appendix C.

Belloni et al. (2014) propose using a double selection lasso which consists of three stages in the case of a linear model: 1) estimate a lasso of the outcome variable on the full set of controls; 2) estimate a lasso of the variable of interest on the full set of controls; 3) estimate an OLS model of the outcome on the variable of interest including all controls selected in steps 1 and 2. This approach ensures that all appropriate controls are included and reduces bias in the coefficient on the variable of interest. It can also be extended to logistic regression models that we use for sale incidence, bid acceptance, and bid placement decisions (Belloni et al. 2016). To evaluate and model our bot, we use double selection models for sales and bid acceptance as these decisions involve counterfactual bids and prices chosen by the bot.

We use post-lasso models for bid placement and bid amount decisions by other actors in the market (as the goal here is to predict their behavior given initial listing characteristics rather than to see how they react to counterfactual decisions by the bot).

Table 2 shows that the two approaches can select different sets of control variables, where the double selection lasso usually selects fewer coefficients than the post-lasso approach (except for the bid acceptance model). However, the coefficients of interest (on price and max bid terms) are relatively similar across the two approaches with some minor differences.

Table 2: Full Information Model Estimates

	Sale		Bid Accept		Bid Place		Bid Amount	
	i	ii	i	ii	i	ii	i	ii
Intercept	-3.784**	-2.319*	-0.549	-1.759	-2.493**	-0.111	3.346	-6.219***
	(1.211)	(1.125)	(1.748)	(2.631)	(0.935)	(1.005)	(1.822)	(1.014)
log(Token	-0.093***	-0.093***	-0.040		0.017	0.023	-0.070**	-0.066**
ID)	(0.017)	(0.017)	(0.043)		(0.018)	(0.018)	(0.026)	(0.021)
log(Rarity)	-0.042***			-0.022	0.022			
	(0.011)			(0.017)	(0.011)			
log(Price)	-1.853***	-1.679***			-0.369***	-0.358***	0.168***	0.240***
	(0.033)	(0.187)			(0.020)	(0.051)	(0.030)	(0.051)
log(Max			0.301***	0.353***				
Bid)			(0.044)	(0.070)				
Month FE	Y	Y	Y	Y	Y	Y	Y	Y
# Selected	126	85	51	57	120	93	127	57
Observations	50,556	50,556	8,208	8,208	50,556	50.556	8,208	8,208
LL / R^2	-22,096	-22,303	-3,082	-3,099	-21,144	-21,186	0.475	0.466

Note: ***: p<0.001, **: p<0.01, *: p<0.05. Column i shows post-lasso estimates and column ii shows double selection estimates. Models for "Sale," "Bid Accept," and "Bid Place" are logistic whereas the model for "Bid Amount" is linear. "# Selected" refers to the number of selected non-zero coefficients.

As discussed in the previous section, we use limited-data versions of the above models with an expanding window to obtain optimal bids and prices for each day that is part of our 4 month out-of-sample period. On day t, we search over a 2-dimensional bid-price grid for each listing to maximize expected profits based on the estimated limited-data models up to but excluding day t. Figure 2 shows an example of the optimal bid-price combinations implied by the models for token ID 7233 listed on 2021-08-02 using expected profits constructed from models estimated with data up to 2021-08-01. The left plot shows that the impact of resale price on profits is small as the lines are relatively flat. However, bid amount has a

significant impact on profits as the lines associated with higher bids appear much lower on the graph. This can be seen more explicitly in the right plot where the curves decrease very quickly as the bot's bid increases at different levels of resale prices. This pattern occurs because the bot is not guaranteed to be able to resell the item after it is acquired but must pay the bid upfront if the seller accepts it. As a result, only small bids justify the risk of carrying the NFT in inventory with an uncertain resale probability.

Figure 2: Examples of Optimal Bid-Price Combinations

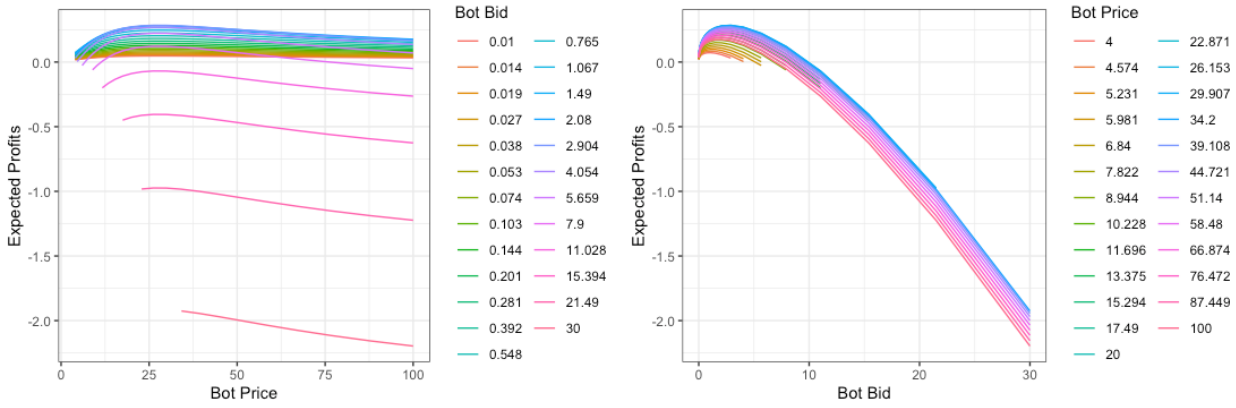


Table 3 summarizes the bot's performance statistics. We optimize over the bot's out-of-sample profits and obtain the optimal hyperparameters $\nu = \{0.132, 0.011, -0.017\}$. While these hyperparameters slightly increase profits they do not have a significant impact on return-on-investment (ROI), suggesting that no hyperparameter adjustments are required in this setting. This is not unexpected as we use an expanding window to make decisions only one day ahead.

Table 3: Bot Performance Summary

	Optimized Hyperparameters	Zero Hyperparameters
Profit	7,442.13	7,436.54
Total Gains	7,598.66	7,599.92
Total Losses	-156.54	-163.38
Total Invested	73,514.37	72,621.25
Return-on-Investment	10.12%	10.24%
Bids Placed (% Listings)	27,179 (100%)	27,179 (100%)
Total Bid Amount Placed	2635.473	2592.248
Bids Accepted (% Bids Placed)	1,011 (3.72%)	1,005 (3.70%)
Resales (%)	42.32%	42.35%
Min. Bid	0.201	0.144
Mean. Bid	2.705	2.672
Med. Bid	2.080	2.080
Max. Bid	30	30
Min. Price	4	4
Mean. Price	26.37	26.88
Med. Price	22.87	22.87
Max. Price	100	100

Table 3 shows that we can achieve a profit of 7,436.54 ETH and an ROI of 10.24% in the zero hyperparameter scenario over the 4 month out-of-sample period. The losses are relatively small (-163.38 ETH) compared to gains (7,599.92 ETH). The bot placed a total of 72,621.25 ETH worth of bids on 100% of the listings while only 2,592.25 ETH worth of bids (3.7% of listings) were accepted by sellers. The bids are quite low at 2.672 ETH on average. Resale prices are lower than observed listing prices (median: 23 ETH versus 31 ETH). The bot appears to adopt a "spamming" strategy where it places a large number of very low bids with the hope that some of them will be accepted (accidentally or out of desperation) by a small fraction of sellers. It then resells items at a discount relative to their original listing price but earns a large margin in cases where its bids are accepted. This strategy is not uncommon in practice and several cases of sellers accidentally accepting very small bids have been documented in the press.[4] Our analysis explains why such a strategy is popular with bot developers. In Web Appendix B we illustrate the bot's performance over time.

[4] See several examples here: 1) https://www.benzinga.com/markets/cryptocurrency/21/11/23824144/cryptopunk-nft-accidentally-sold-for-19k-instead-of-19m; 2) https://bitcoinist.com/cryptopunks-nft-accidentally-sells-for-one-penny-worth-of-ethereum; 3) https://edition.cnn.com/style/article/nft-accidentally-bored-ape-sold-intl-scli/index.html; 4) https://coinmarketcap.com/alexandria/article/nft-collector-loses-391-000-as-bored-ape-yacht-

Extensions and Robustness

We consider several alternative bot designs which can be achieved by changing the objective function or adjusting the hyperparameters.

Buy-it-now Bot –This bot does not place bids but rather buys items at their listing price and attempts to resell them at a higher price. Clearly, this is a riskier strategy as the bot must commit to paying the listing price upfront. Web Appendix E presents the equations for this bot's objective and Table A.4 summarizes its performance with optimized hyperparameters (which improve performance in this case). The number of NFTs purchased by the optimized bot is very small (5), and only 1 (0.896) is successfully resold in expectation. The attempted resale prices are high (mean: 78.61 ETH) suggesting that the bot purchases very underpriced but valuable NFTs and attempts to resell them at high prices. There are not many opportunities to do this in practice, and the bot struggles to achieve profitability if it depends only on one expected successful resale over 4 months. These results show why bots that purchase at listing prices (like Zillow's iBuying algorithm) and do not place bids at significant discounts may struggle to achieve profitability.

Seller-Rewarding Bot – This bot can be designed to increase seller revenues simply by adjusting the hyperparameters in the bot's expected profit function. Marketplaces may issue such bots to stimulate activity (for example, if they wish to reward sellers and present them with more opportunities to sell their NFTs to increase loyalty). Web Appendix E presents equations and Table A.5 shows how such a bot can "pay for itself." In the original optimized hyperparameter scenario, bot profits are 7,442.13 ETH and seller revenues are 251,960.8 ETH. When we set $\nu^A = 26.5$, bot profits decrease to 130.85 ETH but seller revenues increase to 259,916.4 ETH. The bot can be tuned to create more value for the sellers while still generating a small profit for itself. As we increase ν^A to 100, the bot's profit becomes slightly

club-nft-accidentally-sells-for-just-115; 5) https://www.crypto-news-flash.com/an-nft-collector-accidentally-sold-etherrock-for-0/

negative at -123.33 ETH and seller revenues decrease to 240,707 ETH. Increasing the bot's weight on bid acceptance by too much can have a non-linear effect on seller revenues as it reduces the percentage of listings on which the bot bids (92.93% at $\nu^A = 26.5$ but 6.39% at $\nu^A = 100$), which may reduce opportunities for sellers to sell their items.

Robustness Checks – in Web Appendix F, we test the robustness of our analysis to realistic bidding fees (50 USD = 0.017 ETH per listing), holding values (which assume that the bot can resell its NFTs at lower prices in the future if they fail to sell immediately), and higher bids (where we examine the sensitivity of the results to forcing the bot to bid higher). Our main conclusions are robust. Bidding fees do not significantly affect the results. Holding values (equal to 30% of each NFT's predicted bid amount from the full-data bid amount model) increase the bidding bot's ROI only slightly (from 10.24% to 11.94%) and increasing the number of NFTs purchased by the buy-it-now bot from 5 to 8, of which only 1-2 sell in expectation, which does not improve its profitability much compared to the baseline case. Finally, slightly higher bids can significantly reduce the bidding bot's ROI, suggesting that its profitability is highly contingent on its ability to place very small bids.

Conclusion

In this research, we explore the possibility of "artificially intelligent" peer-to-peer marketplaces where bots participate together with humans and attempt to generate profits by reselling items. We develop a back-testing framework which allows developers to test different bots and use it to find bot strategies that are effective in the CryptoPunks marketplace. Our framework incorporates out-of-sample outcome data as well as full- and limited-data statistical models to evaluate bot performance. We show that bots that "spam" a large number of very small bids to every listing, win infrequently (3.70% of the time), and resell at a slight discount compared to listing prices can be profitable with an ROI of 10.24% over 4 months. However, buy-it-now bots that buy at listing prices and attempt to resell are rarely active and can struggle to achieve profitability. Our framework allows developers to test seller-rewarding bots to stimulate the

seller-side of the market at no expense, allowing for a variety of bots with different objectives. Our findings help explain when it makes sense to use trading bots in peer-to-peer marketplaces and provide a framework for developers to test alternative strategies.

References

Amano T, Rhodes A, Seiler S (2022) Flexible demand estimation with search data. Working Paper.

Bajari P, Hortaçsu A (2004) Economic insights from internet auctions. J. Econ. Lit. 42(2):457-486.

Bauner C (2015) Mechanism choice and the buy-it-now auction: A structural model of competing buyers and sellers. Int. J. Ind. Organ. 38:19-31.

Belloni A, Chernozhukov V, Hansen C (2014) Inference on treatment effects after selection among high-dimensional controls. Rev. Econ. Stud. 81(2):608-650.

Belloni A, Chernozhukov V, Wei Y (2016) Post-selection inference for generalized linear models with many controls. J. Bus. Econ. Stat. 34(4):606-619.

Boudreau KJ, Lakhani KR, Menietti M (2016) Performance responses to competition across skill levels in rank-order tournaments: Field evidence and implications for tournament design. RAND J. Econ. 47(1):140-165.

Borri N, Liu Y, Tsyvinski A (2022) The economics of non-fungible tokens. Working Paper.

Caro F, Gallien J (2012) Clearance pricing optimization for a fast-fashion retailer. Oper. Res. 60(6):1404-1422.

Choi H, Mela CF (2019) Monetizing online marketplaces. Market. Sci. 38(6):948-972.

Dubé JP, Fang Z, Fong N, Luo X (2017) Competitive price targeting with smartphone coupons. Market. Sci. 36(6):944-975.

Dubé JP, Misra S (2019) Personalized pricing and customer welfare. Working Paper.

Ehling P, Gallmeyer M, Srivastava S, Tompaidis S, Yang C (2018) Portfolio tax trading with carryover losses. Manage. Sci. 64(9):4156-4176.

Fang F, Ventre C, Basios M, Kanthan L, Martinez-Rego D, Wu F, Li L (2022) Cryptocurrency trading: a comprehensive survey. Fin. Innov. 8(1):1-59.

Ferreira KJ, Lee BH, Simchi-Levi D (2016) Analytics for an online retailer: Demand forecasting and price optimization. Manuf. Serv. Op. 18(1):69-88.

Halaburda H, Haeringer G, Gans JS, Gandal N (2022) The microeconomics of cryptocurrencies. J. Econ. Lit. Forthcoming.

Kannan PK, Pope BK, Jain S (2009) Practice prize winner—Pricing digital content product lines: A model and application for the National Academies Press. Market. Sci. 28(4):620-636.

Kireyev P (2020) Markets for ideas: Prize structure, entry limits, and the design of ideation contests. RAND J. Econ. 51(2):563-588.

Kireyev P, Lin R (2021) Infinite but rare: valuation and pricing in marketplaces for blockchain-based nonfungible tokens. Working Paper, INSEAD.

Kireyev P (2022) NFT Marketplace Design and Market Intelligence. Working Paper, INSEAD.

Lucking-Reiley D (1999) Using field experiments to test equivalence between auction formats: Magic on the internet. Am. Econ. Rev. 89(5):1063-1080.

Nadini M, Alessandretti L, Di Giacinto F, Martino M, Aiello LM, Baronchelli A (2021) Mapping the NFT revolution: market trends, trade networks, and visual features. Sci. Rep. 11(1):1-11.

Oh S, Rosen S, Zhang AL (2022) Investor experience matters: Evidence from generative art collections on the blockchain. Working Paper.

Tibshirani R. Regression shrinkage and selection via the lasso. J. R. Stat. Soc. B 58(1):267-288.

Kishore V, Janosov M, Barabási AL (2022) Quantifying NFT-driven networks in crypto art. Sci. Rep. 12(1):1-11.

Wang X, Montgomery A, Srinivasan K (2008) When auction meets fixed price: A theoretical and empirical examination of buy-it-now auctions. Quant. Mark. Econ. 6(4):339-370.

Yao S, Mela CF (2008) Online auction demand. Market. Sci. 27(5):861-885.

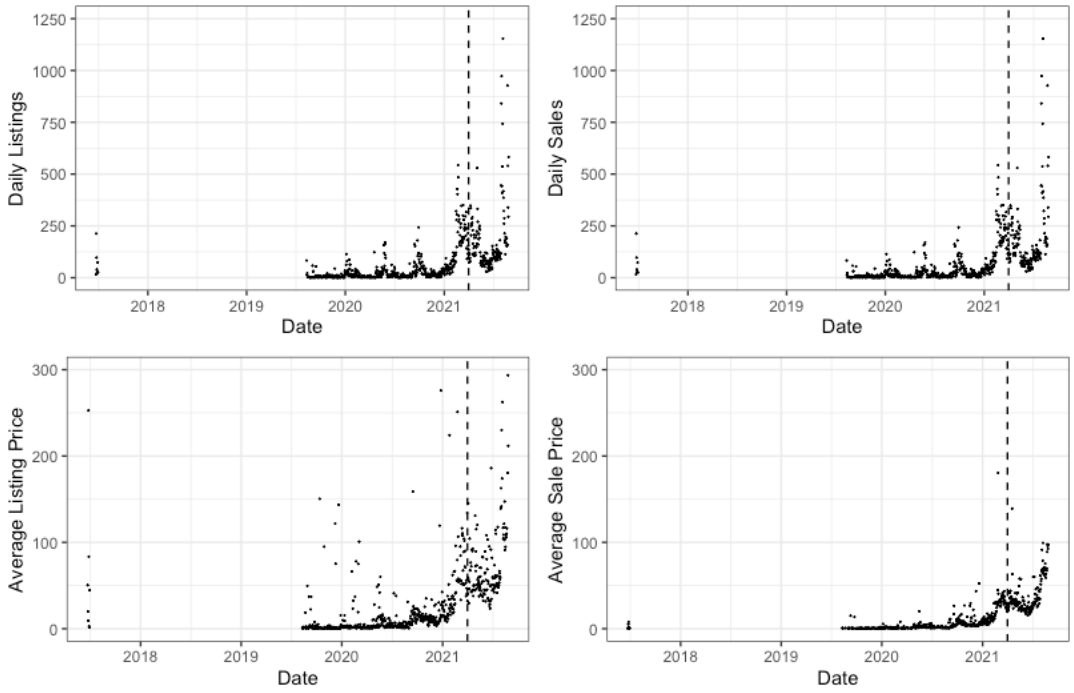
Yoganarasimhan H (2013) The value of reputation in an online freelance marketplace. Market. Sci. 32(6):860-891.

Web Appendix

A. Marketplace Trends

Figure A.1 displays the marketplace trends for daily listings, sales, average listing price, and average sale price. The vertical line shows the period when our out-of-sample evaluation data begin. The number of listings in our evaluation data is 27,179. Note that market behavior in our evaluation data may differ significantly from behavior in the training data. This is why we use an expanding window to estimate our limited-data models. The non-stationary nature of the market also helps us test the bots we design in a realistic and potentially unstable environment where the data-generating-process may change over time.

Figure A.1: Marketplace Trends

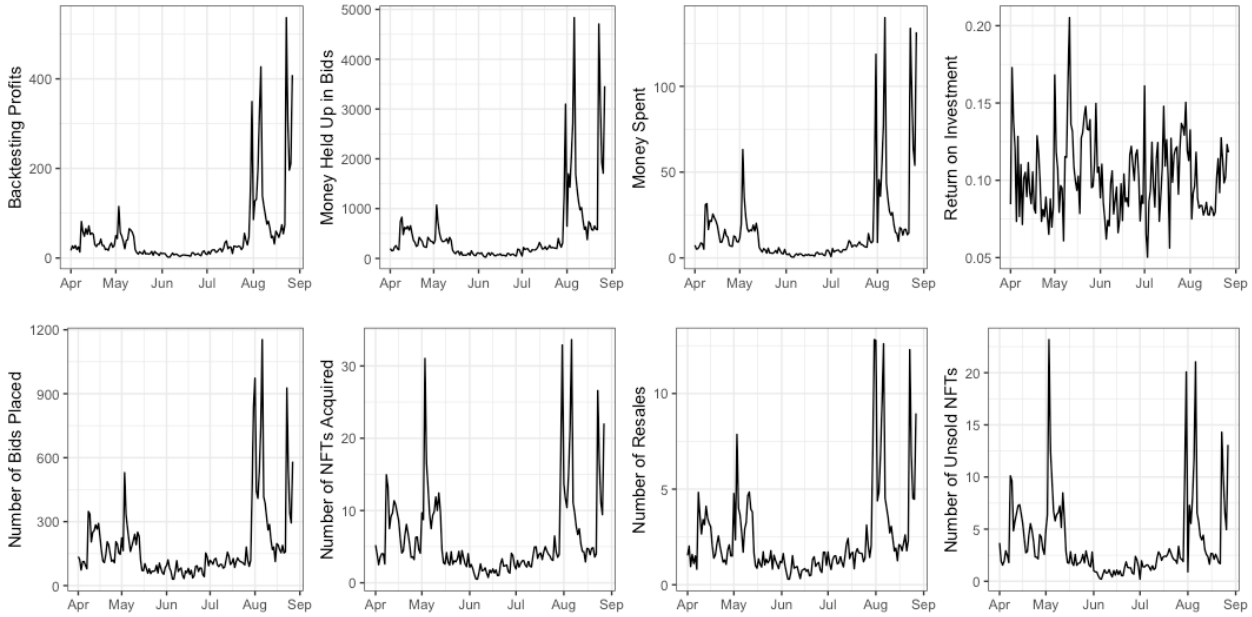


Note: 19 observations are excluded from the bottom left plot where daily average listing price > 300 ETH.

B. Bot Performance Over Time

Figure A.2 tracks the performance of our bidding bot over time, starting from April when the out-of-sample evaluation period begins. Most metrics appear to follow the same trend as the trend in daily listings as indicated by the above plots in Figure A.1. While the daily return-on-investment is volatile, it oscillates around the overall ROI of 10.24%. Bot developers are likely to see that their bot's performance is very dependent on marketplace activity. If more listings are available, then the bot has more opportunities to acquire and resell NFTs. Naturally, most profits can be obtained during periods of high activity in the market (bull markets) whereas the bot takes fewer actions in low activity periods (bear markets) which results in lower profits during these periods.

Figure A.2: Bot Behavior Over Time



Note: Backtesting profits, money held up in bids, and money spent are measured in ETH.

C. Tests for Impact of Unobserved NFT Attributes

We estimate additional models to explore the effects of unobserved attributes on price coefficients in this market. Omitted attribute concerns are limited because the data contain the full set of item characteristics that explain the appearance of each NFT. Table A.1 shows estimates from a linear probability model of sales incidence with the dependent variable equal to 1 if a sale occurs and 0 otherwise. Each column includes different types of fixed effects. Column i includes a limited set of variables that excludes product-specific fixed effects and many of our controls. Column ii introduces fixed effects for each possible attribute, which amounts to almost 100 additional estimated coefficients. Column iii considers seller fixed effects to see if seller-specific unobservables may be correlated with prices. Column iv considers the most granular seller/token ID fixed effects and uses only variation from when the same seller lists the same token multiple times for different prices to identify the price coefficient, thereby controlling for seller and token-specific unobservables. Finally, column v considers the most granular weekly time fixed effects together with seller/token ID fixed effects. The price coefficient remains relatively stable across all specifications although it becomes more negative after seller/token ID fixed effects are considered. This may correspond to unobserved seller marketing efforts that are positively correlated with listing prices for specific tokens. We find that our main results are unchanged even if we force the price coefficient to be more negative in our full-data model (results available upon request). We also find that the results do not change if we include seller-specific sales history variables as additional controls in our models (results available upon request).

Table A.1: Tests for Endogeneity in Demand Model

DV: Sale = 1	i	ii	iii	iv	v
Intercept	0.756***	0.735***			
	(0.033)	(0.045)			
log(Token ID)	-0.011***	-0.013***	-0.017***		
	(0.002)	(0.002)	(0.004)		
log(Rarity)	-0.003***	-0.005	-0.004**		
	(0.001)	(0.005)	(0.001)		
Male	-0.264***	-0.236***	-0.251***		
	(0.020)	(0.029)	(0.026)		
Female	-0.287***	-0.270***	-0.277***		
	(0.020)	(0.028)	(0.026)		
log(Price)	**-0.143*** **	**-0.152*** **	**-0.153*** **	**-0.187*** **	**-0.195*** **
	(0.002)	**(0.003)**	**(0.003)**	**(0.003)**	**(0.004)**
Month FE	Y	Y	Y	Y	
Week FE					Y
Attribute FE		Y			
Seller FE			Y		
Token ID - Seller FE				Y	Y
Observations	49,440	49,440	49,199	48,576	48,576
R^2	0.108	0.116	0.082 (within)	0.084 (within)	0.094 (within)

Note: ***: p<0.001, **: p<0.01, *: p<0.05. Observations count is obtained by subtracting listings with only one observation per fixed effect level from the total number of listings.

Table A.2 shows estimates from a binary choice model of demand where we include all attribute fixed effects. We find that this makes the price coefficient more negative indicating that it is important to consider the large set of traits that we include in our main specification.

Table A.2: Impact of Including Attribute Fixed Effects in a Logistic Demand Model

DV: Sale = 1	i	ii
Intercept	2.423***	2.541***
	(0.252)	(0.348)
log(Token ID)	-0.091***	-0.106***
	(0.016)	(0.017)
log(Rarity)	-0.042***	-0.120***
	(0.006)	(0.036)
Male	-2.751***	-2.842***
	(0.181)	(0.248)
Female	-2.841***	-3.056***
	(0.180)	(0.239)
log(Price)	**-1.465*** **	**-1.735*** **
	(0.025)	**(0.031)**
Month FE	Y	Y
Attribute FE		Y
Observations	49,440	49,440
Log-Likelihood	-22,473	-22,192

Note: ***: p<0.001, **: p<0.01, *: p<0.05.

D. Full Set of Variables Used in Estimation

In addition to dummy variables for all of the discrete traits of the CryptoPunks (92 such variables), Table A.3 shows additional variables that we construct and use as potential confounders in estimation of the full-data and limited-data models.

Table A.3: List of Variables Used in Estimation (Excluding Trait Fixed Effects)

Variable Name	Description
log(Token ID)	The logarithm of the token ID of the NFT.
Last Sale Price	The last sale price of the NFT in ETH.
Last Sale Price (USD)	The last sale price of the NFT in USD.
ETH Price	The ETH/USD exchange rate at the time of listing.
Gas	The gas price at the time of listing.
Monthly Fixed Effects	Fixed effects for each month.
log(Rarity)	The logarithm of the rarity of the NFT.
Max(Rarity)	The rarity of the least rare attribute of the NFT.
Min(Rarity)	The rarity of the most rare attribute of the NFT.
Number of Traits	The total number of traits of the NFT.
Weighted Other Price	Average price of other NFTs listed on the same day, weighted by inverse absolute log(rarity) difference with the focal NFT.
Weighted Other Number	Number of other NFTs listed on the same day, weighted by inverse absolute log(rarity) difference with the focal NFT.
Weighted Other Sum	Sum of prices of other NFTs listed on the same day, weighted by inverse absolute log(rarity) difference with the focal NFT.
# Other Listings	Number of other listings on the same day as the focal listing.
Skin Color	Skin color of the CryptoPunk.

E. Alternative Bots

We consider several alternative designs of the bot which can be achieved by changing the objective function or adjusting the hyperparameters.

Buy-it-now Bot –This bot does not place bids but rather buys items at their listing price and attempts to resell them at a higher price. Clearly this is a riskier strategy as the bot must commit to paying the listing price upfront. The expected profits of the buy-it-now bot can be expressed as:

$$\pi_{jt}^{Buy}(P_{jt}) = \Pr(S_{jt} = 1|P_{jt})P_{jt} + \Pr(S_{jt} = 0|P_{jt}) \Pr(B_{jt}^{-} > 0|P_{jt}) \mathrm{E}(B_{jt}^{-}|B_{jt}^{-} > 0; P_{jt}) - P_{jt}^{data}$$

where the bot makes a purchase at the original listing price P_{jt}^{data} if and only if $\pi_{jt}^{Buy}(P_{jt}^{*}) > 0$ at optimal resale price P_{jt}^{*}. We assume that the buy-it-now bot moves quickly and acquires NFTs before other participants have a chance to bid on them. As a result, a bid by others may be placed on the NFT after the bot acquires it. In contrast to the bidding bot models, we use double selection lasso to model bid placement and bid amount decisions (by other participants) in this case as we wish to model how these decisions change in response to the bot's counterfactual pricing decisions. The hyperparameter version of expected profits can be written as:

$$\pi_{jt}^{Buy}(P_{jt}|\nu^{S}) = \Pr(S_{jt} = 1|P_{jt})P_{jt} + \Pr(S_{jt} = 0|P_{jt}) \Pr(B_{jt}^{-} > 0|P_{jt}) \mathrm{E}(B_{jt}^{-}|B_{jt}^{-} > 0; P_{jt}) - P_{jt}^{data} + \nu^{S} \Pr(S_{jt} = 1|P_{jt})$$

with purchases occurring when $\pi_{jt}^{Buy}(P_{jt}^{*}) > \nu^{\pi}$.

Table A.4 summarizes the buy-it-now bot's performance. We find optimal hyperparameters $\nu^{S} = 0.1$ and $\nu^{\pi} = 0.1$ suggesting that the bot should be more conservative out-of-sample, placing a higher weight on resale probability and purchasing only if expected profits exceed a positive threshold. However, the number of NFTs purchased by the optimized bot is very small (5) of which only 1 (0.896) is successfully resold in expectation. The attempted resale prices are high (mean: 78.61 ETH) suggesting that the bot purchases very underpriced but valuable NFTs and attempts to resell them at a high price. There are not many opportunities to do this in practice, and the buying bot struggles to achieve profitability if it relies only on one expected successful resale over 4 months to achieve its goal. These results show why bots that purchase at listing prices and do not make bids at significant discounts (like Zillow's iBuying algorithm) may struggle to achieve short-term profitability.

Table A.4: Buy-it-now Bot Performance Summary

	Optimized Hyperparameters	Zero Hyperparameters
Profit	10.028	7.778
Total Gains	17.368	17.368
Total Losses	-7.339	-9.590
Total Invested	82.49	97.99
Return-on-Investment	12.16%	7.94%
Items Purchased (% Listings)	5 (0.018%)	6 (0.022%)
Resales (% of Items Purchased)	0.896 (17.92%)	1 (18.15%)
Min. Price	47.34	47.34
Mean. Price	78.61	74.95
Med. Price	82.28	74.21
Max. Price	100	100

Seller-Rewarding Bot – This bot can be designed to increase seller revenues simply by adjusting the hyperparameters in the bot's expected profit function. Marketplaces may issue such bots to stimulate activity in their market (for example, if they wish to reward sellers and present them with more opportunities to sell their NFTs to increase loyalty). We demonstrate that marketplaces can launch such bots at almost no cost.

We write seller revenue as follows:

$$SellerRevenue = \sum_{jt \mid bid\ is\ placed} \left(\Pr\left(A_{jt} = 1 \middle| B_{jt}, A_{jt}^{data}\right) B_{jt} + \Pr\left(A_{jt} = 0 \middle| B_{jt}, A_{jt}^{data}\right) Q_{jt}^{data}\right) + \sum_{kt \mid no\ bid\ is\ placed} Q_{kt}^{data}$$

where the first sum is over all listing where the bot placed a bid, the second sum is over all other listings, and Q_{jt}^{data} is the outcome observed in the data, i.e., $Q_{jt}^{data} = 0$ if no sale or bid acceptance happened for listing jt, $Q_{jt}^{data} = B_{jt}^{data}$ if a bid was made and accepted in the data, and $Q_{jt}^{data} = P_{jt}^{data}$ if a sale happened at the original listing price in the data. The term A_{jt}^{data} in the probability expression indicates that we evaluate the Bayesian probability of the acceptance or sale event occurring in simulations conditional on whether it occurred in the data, as we do when evaluating out-of-sample bot performance. All calculations are performed out-of-sample.

Table A.5: Seller-Rewarding Bot Performance Summary

	Optimized Hyperparameters	$\nu^A = 26.5$	$\nu^A = 100$
Seller Revenues	251,960.8	259,916.4	240,707.2
Profit / ROI	7,442.13/10.12%	130.85/0.05%	-123.33/-0.24%
Total Gains	7,598.66	4785.36	546.27
Total Losses	-156.54	-4654.51	-669.60
Total Invested	73,514.37	251,503.3	52,140
Bids Placed (% Listings)	27,179 (100%)	25,257 (92.93%)	1738 (6.39%)
Total Bid Amount Placed	2,635.473	14,655.7	2,477.418
Bids Accepted (% Bids Placed)	1,011 (3.72%)	1,534 (6.07%)	82 (4.72%)
Resales (%)	42.32%	40.47%	42.56%
Min. Bid	0.201	5.659	30
Mean. Bid	2.705	9.958	30
Med. Bid	2.080	11.028	30
Max. Bid	30	30	30
Min. Price	4	5.981	34.20
Mean. Price	26.37	28.072	68.94
Med. Price	22.87	26.153	66.87
Max. Price	100	100	100

Table A.5 summarizes the performance of the seller-rewarding bot under different hyperparameter choices, with the left column indicating the optimized hyperparameter bot performance for comparison. In all other cases, the ν^S hyperparameter is held at its optimal level and ν^π is fixed at -0.5 to ensure that the bot bids more aggressively and does not simply avoid bidding altogether if it anticipates a loss. In the optimized hyperparameter scenario, bot profits are 7,442.13 ETH and seller revenues are 251,960.8 ETH. When we set $\nu^A = 26.5$, bot profits decrease to 130.85 ETH but seller revenues increase to 259,916.4 ETH. This example illustrates how the bot can be tuned to create more value for the sellers while still generating a small profit for itself. As we increase ν^A to 100, the bot's profit becomes slightly negative at -123.33 ETH. However, seller revenues decrease to 240,707 ETH. Increasing the bot's weight on bid acceptance by too much can have a non-linear effect on seller revenues as it reduces the percentage of listings on which the bot bids (92.93% at $\nu^A = 26.5$ but 6.39% at $\nu^A = 100$) holding fixed the profit threshold, which may reduce opportunities for sellers to sell their items.

F. Robustness

We test the robustness of our analysis to several phenomena. Table A.6 summarizes the findings, all with optimized hyperparameters.

Table A.6: Robustness Tests

	Bidding Fees (50USD = 0.017ETH)	Holding Values (30% of bid estimate) Bid. bot/Buy. bot	Higher Bids ($v^A = 10$)
Profit	6,979.76	14,989.69 /69.74	5,728.83
Total Invested	73,616.51	125,562.6 /166.95	137,880.8
Return-on-Investment	9.48%	11.94%/41.77%	4.15%
Bids Placed/Items Purchased (% Listings)	27,179 (100%)	27179(100%)/8(0.029%)	24,833 (91.37%)
Total Bid Amount Placed	2,358.707	5,386.55/NA	5,796.087
Bids Accepted (% Bids Placed)	1,011 (3.72%)	1,221 (4.49%) /NA	1,233 (4.97%)
Resales (%)	43.23%	25.79%/12.83%	40.86%
Min. Bid	0.201	0.765/NA	2.904
Mean. Bid	2.709	4.620/NA	5.552
Med. Bid	2.080	4.054/NA	5.659
Max. Bid	30	30/NA	30
Min. Price	4	5.981/63.43	5.231
Mean. Price	26.41	43.00/95.43	28.317
Med. Price	22.87	39.11/100	26.153
Max. Price	100	100/100	100

Bidding Fees – We assume that the bot incurs a fee of about 0.017 ETH (50 USD) every time it must place a bid. This fee is approximately equal to the gas fee incurred by users of the Ethereum blockchain when making similar transactions during the period of the study. We modify the expected profit expression as follows:

$$\pi_{jt}(B_{jt}, P_{jt}) = \Pr(A_{jt} = 1 | B_{jt}) [\Pr(S_{jt} = 1 | P_{jt}) P_{jt} - B_{jt}] - F$$

where F is the bidding fee. The bot's ROI decreases slightly from 10.12% to 9.48% but its overall behavior remains largely unaffected (it still bids on 100% of the listings) suggesting that our findings are robust to the introduction of realistic bidding fees.

Holding Values – We relax the assumption that an NFT is worth zero to the bot if it does not sell, and allow for an NFT-specific holding value equal to 30% of the predicted bid amount from the full-data

model. This specification assumes that the bot may be able to sell the NFT at some point in the future and assigns a positive discounted future value of holding the NFT. While we do not calculate this value explicitly in a dynamic programming framework, we approximate it as 30% of the predicted bid amount and adjust this value upwards and downwards to see how sensitive our results are to it. We do this for both the bidding bot and the buy-it-now bot.

In the case of the bidding bot, expected profits become:

$$\pi_{jt}(B_{jt}, P_{jt}) = \Pr(A_{jt} = 1|B_{jt})[\Pr(S_{jt} = 1|P_{jt})\, P_{jt} + \Pr(S_{jt} = 0|P_{jt})\, H_{jt} - B_{jt}]$$

where $H_{jt} = .3 \times \hat{B}_{jt}^{-}$ is the holding value - an approximation of the discounted future value of holding the NFT. The middle column of Table A.6 shows that ROI increases slightly to 11.94% but the results remain generally unchanged. Similarly, for the buy-it-now bot:

$$\pi_{jt}(P_{jt}) = \Pr(S_{jt} = 1|P_{jt})P_{jt} + \Pr(S_{jt} = 0|P_{jt})\left(\Pr(B_{jt}^{-} > 0|P_{jt})\, \mathrm{E}(B_{jt}^{-}|B_{jt}^{-} > 0; P_{jt}) + \Pr(B_{jt}^{-} = 0|P_{jt})\, H_{jt}\right) - P_{jt}^{data}.$$

The ROI appears to increase significantly, to 41.77% relative to 12.16%. However, this is because the bot still purchases a very small number of items (8, compared to 5 when $H_{jt} = 0$). Hence, our main results from the buy-it-now bot extension carry over, as the profitability of this bot relies on purchasing a very small number of items and hoping that 1-2 of them result in a sale, which is a very risky and unreliable strategy.

Higher Bids – We force the bot to make higher bids to examine the sensitivity of our results to the lowest bids the bot is allowed to place. The final column shows that the bot can still achieve a positive ROI of 4.15% even when the average bid is increased from 2.705 ETH to 5.552 ETH. In this case, the bot will bid on about 9% fewer listings. However, it is clear that profits are sensitive to bid amounts and it is in the best interest of the bot to "spam" a large number of very low bids.