There's An App (Update) For That

Understanding Product Updating Under Digitization*

Benjamin T. Leyden
Dyson School of Applied Economics and Management
Cornell University
leyden@cornell.edu †

November 17, 2018

Abstract

The digitization of consumer goods gives firms the ability to monetize and update already purchased products, changing firms' product innovation incentives. I develop and estimate a structural model of the smartphone application (app) industry, to study how the availability of these tools affects the frequency and content of product updates. I construct a novel database of apps on Apple's mobile platform, and employ natural language processing and machine learning techniques to classify product updates and define precise categorical markets. I find that the availability of these tools via digitization result in an increase in the frequency of product updates of 63% to 142%, and, in particular, lead to an increase in the relative frequency of major, feature-adding updates compared to more minor, incremental updates. These results show that the manner in which product digitization changes firms' product innovation incentives has a significant effect on firm behavior, and should be accounted for in future research on digital and digitizing industries.

JEL Classifications: L13, L15, L86, O30
Keywords: Product innovation, Endogenous product characteristics, Dynamic oligopoly, Software, Digitization

*Portions of this paper were previously circulated under the title "Consumer Preferences for Product Updates Under Digitization: A Model of Demand for Smartphone Applications."

†I am grateful for guidance from Simon Anderson, Federico Ciliberto, Gaurab Aryal, and many other faculty members and students at the University of Virginia. This work has greatly benefited from comments from conference participants at IIOC 2017 & 2018, the 2017 ZEW/MaCCI Conference on Innovation and Patenting, the 2017 Searle Center Conference on Internet Commerce and Innovation, EARIE 2017, and NABE TEC 2018. I am indebted to the app developers who have spoken with me about their industry, and, in a some cases, provided data that have been essential to this project. I acknowledge the Bankard Fund for Political Economy and the Radulovacki Summer Research Fund for financial support. All errors are my own.

1 Introduction

Consumer goods are becoming increasingly software-dependent and internet-connected. This *digitization* of consumer goods changes the incentives firms face regarding when and how to engage in product innovation. This is because digitization has the potential to extend the interaction between consumers and firms, which has typically ended at the point of sale, by providing firms with two new strategic tools. First, under digitization firms are able to monetize the use of their products, for example, by including advertisements in the product or by selling additional, instantly available features to consumers. Second, the digital nature of these products makes it possible for firms to update a product after the consumer has purchased it. Together, these two changes — the ability to monetize the use of a product, and the ability to update a product past the point of sale — have the potential to change how firms choose to innovate their products.

I study how the availability of use monetization and post-purchase updating via digitization affects firms' product innovation. Specifically, I ask whether product updates are more or less frequent, and how the content of updates changes when firms have access to these tools. I answer these questions through an empirical analysis of consumer and firm behavior in the context of smartphone application markets on Apple's App Store marketplace.

The digitization of consumer durable good industries is rapidly accelerating. Of particular note is a growing collection of home appliances and gadgets often referred to as the "Internet of Things," due to their incorporation and reliance on internet-connected software. Digitization is also evident in the car industry, where electronic car manufacturer Tesla has repeatedly released free, downloadable updates that increase the quality and functionality of their vehicles. For example, in January of 2015, Tesla released an update that increased acceleration rate of their model P85D car, and in June of 2017, they released an update that improved the semi-autonomous driving capabilities of Tesla cars (Matthews, 2015; Lambert, 2017). It is anticipated that many more product categories will become increasingly digitized in the coming years.

Apps are an extreme case of a digitized durable good as they are fully digital. This provides a unique opportunity to study the effects of the changes brought on by digitization without facing additional modeling and empirical challenges brought forth by the specifics of a particular industry's non-digital production costs and technologies. And, while software has always been a digital

product, the modern app industry differs from the more traditional consumer software developed over the last few decades. In fact, modern apps have undergone a process of digitization, as defined in this paper, similar to many other durable good industries. Software, especially on mobile devices, is now nearly always connected to the internet, and with the increase in the speed and ubiquity of the internet, modern app developers are able to take advantage of the changes of digitization as defined above. To be clear, no single characteristic of the modern app industry is completely new, but rather, it is the confluence of these changes that distinguishes modern app stores from traditional software.

Additionally, app markets are an interesting industry in their own right. Modern app markets began in 2008 with the introduction of Apple's App Store, which was quickly followed by Google's Android Market, a precursor to what is now called Google Play.[1] The number of apps in the App Store has ballooned from 500 in 2008 to over two million, and, according to Apple, apps have been downloaded from the App Store over 130 billion times, resulting in over $100 billion in revenue (Apple, 2016, 2017).[2]

I approach the question of how the introduction of use monetization and post-purchase updating to consumer goods via digitization has affected firms' product innovation, or product updating, behavior through the lens of competition in durable goods. Digitization has the potential to change how firms compete by shifting a firm's focus from encouraging consumers to replace existing products toward encouraging consumers to continue to use existing products. In the standard durable goods model, a firm competes first against other firms to make a sale, and then it competes *against itself*, as firms update their products in order to encourage the consumer to purchase a replacement product. A risk associated with this second aspect of competition is that if a consumer returns to the store, that consumer may switch to a competitor's product. The digitization of consumer goods shifts the emphasis in this game away from trying to push the consumer back to the store, and instead toward trying to keep the consumer engaged with their current product through product updates, while earning a continued revenue stream through various forms of use monetization.

The question of how digitization affects the frequency and content of product updates is an inherently empirical one. It is theoretically ambiguous whether the ability to monetize the use

[1] Henceforth, any mention of the "App Store" is a reference to Apple's app marketplace.

[2] In a mid-2017 press release, Apple announced that nearly $70 billion had been given to developers (Apple, 2017). As is discussed in Section 2, developers keep 70% of the revenue their app collects.

of a product and to update a product after the point of sale will lead to increased or decreased updating, as the effect depends on the relative response to updates by potential new purchasers and by previous purchasers choosing whether or not to use the app. For example, Foerderer and Heinzl (2017), find evidence that while updates increase demand, existing users tend to react negatively (as measured by consumer reviews) to updates, which might result in decreased updating under digitization. Furthermore, it is also unclear how the content of updates will change as a result of digitization. For example, if consumers are relatively responsive to updates, but not to the content of those updates, firms might tend to produce more frequent but lower quality updates. On the other hand, if consumer behavior is particularly attuned to the content of product updates, firms may instead offer less frequent, but higher quality updates.

With this framework in mind, I develop and estimate a structural model of the demand for apps and of developers' dynamic updating decisions. On the demand side, I make a distinction between the extensive margin — consumers choosing to buy new apps — and the intensive margin — consumers choosing whether to use their already owned apps. In the model, consumers consider the expected future value of owning an app when making extensive-margin decisions. On the intensive margin, consumers make period-by-period decisions of whether to use their already-purchased apps or not. Using the model of extensive-margin demand, I estimate consumers' extensive margin preferences over app updates and over the various forms of use monetization the apps employ.

This paper differs from other recent work on the demand for durable goods as it allows for quality changes to an existing product. That is, instead of modeling consumers' dynamic product switching decisions, as in Gowrisankaran and Rysman (2007), I model consumers' purchase decisions in light of their expectations over future product updates and the associated future utility they will receive. Thus, in my model, the inherent quality or characteristics of a product change following developers' updating decisions, unlike, for example, in Lee (2013)'s model of demand for video game software, where a product's inherent quality is fixed, and period-to-period changes in the lifetime expected utility of a purchase occur only as a result of changes in the composition of available products, temporal preference changes, and idiosyncratic match values.

On the supply side, I model developers' dynamic product updating decisions. Developers choose each period whether, and if so, how, to update their app while considering the impact an update will have on future demand and on the behavior of their market-level competitors. Using this

model I estimate the fixed costs of producing minor, bug-fixing updates, and major, feature-adding updates. I then conduct a counterfactual analysis in which I "turn off" the digital aspects of the industry and simulate developers' updating decisions. By comparing developers' updating decisions without use monetization and post-purchase updating to observed behavior, I am able to estimate the extent to which digitization has changed developers' updating behaviors, both in terms of the frequency and content of updates.

This work contributes to a growing literature on endogenous product characteristics. While much of this work has focused on static models (see Eizenberg (2014); Fan (2013)), there has been some work on dynamic endogenous product characteristic decisions, including Goettler and Gordon (2011), who model innovation in the CPU manufacturing industry, and Sweeting (2013). Sweeting (2013)'s model of radio station format changes is closely related to this model developed in this paper. As in this paper, Sweeting estimates a dynamic model of firms making discrete choices from a menu of options. Additionally, in both cases, firms face somewhat atypical profit functions where marginal costs are zero and revenue does not result from consumers' purchase behavior. Unlike Sweeting (2013), though, consumers in my model are forward-looking.

To conduct my analysis, I have created a dataset on the universe of apps on Apple's mobile platform. For both economic and computational reasons, I focus my analysis on a subset of apps in the Productivity category of the App Store, which consists of note-taking apps, task-management apps, and virtual private network apps, among other categories.[3] This data has been collected from a variety of sources, including publicly available information from Apple, and in some cases proprietary data that has been collected directly from app developers.

While the app industry is a data-rich environment, much of the information on apps is in the form of unstructured text. This paper contributes to a growing body of work that integrates text data into economic analysis.[4] In particular, I observe text descriptions of every app and every app update. I use machine learning (ML) and natural language processing (NLP) techniques to process

[3]One reason for focusing on Productivity apps is that many other categories are largely made up of *companion* apps, which exist only to complement other products or services. By studying Productivity apps I am able to focus my attention on apps where the app itself is the product, and thus where we would expect to see profit-maximizing behavior from the developers. See Section B.1 for a more detailed discussion of this point, and see Section 4 for a full discussion of how my sample is created.

[4]E.g., Baker, Bloom, and Davis (2016) develop a new index of economic policy uncertainty using the text of news reports, Wu (2017) analyzes sexism on a popular Economics message board website, and Aryal, Ciliberto, and Leyden (2018) find evidence of tacit collusion in the airline industry using earnings call transcripts. Gentzkow, Kelly, and Taddy (2017) provide a broad overview of economics research that uses text as data.

these texts for use in estimating my model.

First, using text descriptions of every app update, called release notes, I classify every update as either a minor, bug-fixing update or a major, feature-adding update. To do so, I classified by hand a set of release notes, which are then used to train a support-vector machine (SVM). The SVM is able to process every update's release notes and determine whether that update is a Minor or Major update. This distinction serves as a proxy for the quality of an update, and allows my model to account for the heterogeneity in updates.

Second, I develop a method for defining precise categorical markets by again applying NLP and ML techniques to the text app descriptions. In this case, I map the text descriptions to a vector space, and then identify apps by clustering the descriptions in that space using the k-Means clustering algorithm, a form of unsupervised machine learning. This approach should prove useful for other researchers studying long-tail product industries, such as online retailers, or peer-to-peer transaction markets such as Craigslist.[5]

I find that digital product updates increase extensive-margin demand, but that consumers do not appear to differentiate based on the content of updates. That is, there is no statistical evidence that consumers are more responsive to Major updates than Minor updates. While the exact mechanism behind this result is unclear, it suggests that consumers may be poorly informed about the quality of digital goods prior to purchase. On the supply side, I estimate the fixed costs of production for Major and Minor updates and find that Major updates are 22% more expensive to produce than Minor updates. These seemingly contradictory results, that consumers are not more responsive to Major updates, yet Major updates are more costly to produce, points to the need for future work understanding consumer behavior on the intensive, or product use, margin of demand. In Section 6.3 I discuss the role of the intensive margin of demand, where consumers are choosing whether to use previously purchased apps each period, in explaining this apparent contradiction.

Finally, using counterfactual simulations where I simulate developer behavior in the absence of digitization, I find that the availability of use monetization and post-purchase updating both generally increases the frequency of updates *and* the likelihood that an update is a Major, feature-adding

[5]This approach generalizes to other forms of descriptive information on products. E.g., product photographs, which are collectable from most online marketplaces, could be used.

updated, as opposed to Minor, incremental update. These results suggest that we should expect not only to see an increased rate of product updating in many other consumer goods industries, but also an increase in the quality of these updates as quality competition among firms accelerates as a result of digitization.

This paper proceeds as follows: In Section 2 I provide an overview of Apple's App Store marketplace and of the current state of research on the app industry. Given that background, I develop a model of consumer and firm behavior for the app industry in Section 3. Section 4 outlines the data I use to estimate this model, and the data processing steps that must be taken in order to estimate the model. Section 4.4 provides preliminary evidence of the effect of digitization on product updating. Section 5 outlines the estimation procedure for both the demand and supply models, and Section 6 discusses the results of that estimation. Finally, in Section 7 I analyze how developers' updating behavior changes under digitization by considering counterfactual simulations that "turn off" the digital aspects of the industry. Section 8 concludes.

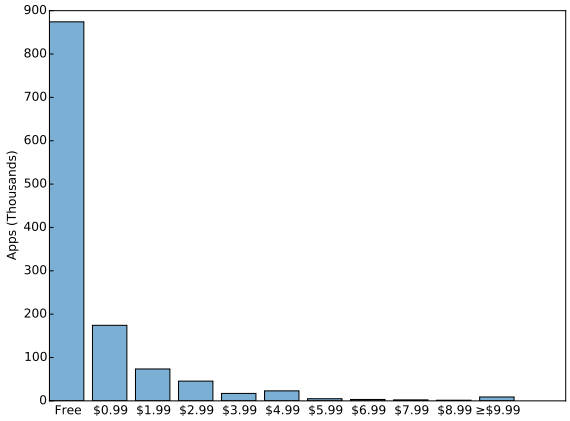
2 The Smartphone Application Industry

The mobile application (app) industry is characterized by a small number of distinct platforms, or mobile operating systems. The most prominent platforms in the United States are Apple's iOS, Google's Android, Microsoft's Windows, and Blackberry. Apple serves 43.3% of the U.S. mobile platform market as of March of 2016 (ComScore, 2016). The industry is a textbook example of a multi-sided market. Consumers purchase hardware, which is exclusively tied to one platform.[6] By purchasing the appropriate hardware, a consumer gains access to the platform's store, which is the primary (and in some cases only) way to download or purchase apps for use on the device. Consumers in this market generally single-home, though some consumers purchase a phone associated with one platform and a tablet associated with another platform.

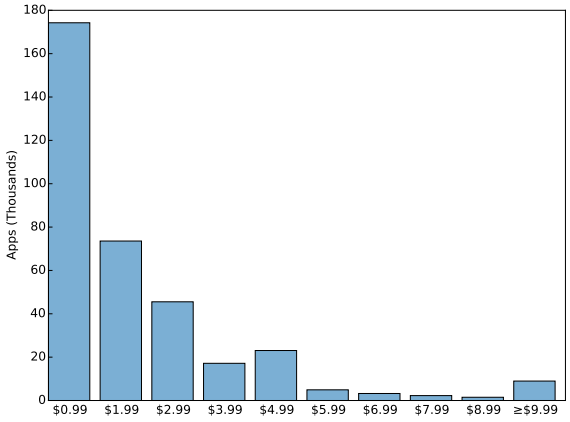
For the remainder of this paper, I limit discussion to Apple's mobile platform, called iOS, and its corresponding App Store marketplace. While the discussion below is specific to Apple, the general structure of the platform is applicable to the other mobile platforms.

[6] Each platform runs on a corresponding set of hardware devices, primarily mobile phones and tablet computers, though some also run on other devices such as televisions.

Figure 1: App Store Prices



(a) Distribution App Store Prices (12/31/2014)



(b) Distribution of App Store Prices, Excluding Free Apps (12/31/2014)

2.1 Apple's iOS Platform

Apple's App Store marketplace is the only approved platform for selling apps for its mobile platform.[7] After being submitted to the App Store by a developer apps are made available to consumers for purchase after a review process conducted by Apple to ensure that the app does not violate a set of regulations Apple maintains for its platform. This same approval process is applied to all app updates.

Developers set the price of their app, and are able to change that price at any time. Developers are restricted to a finite, discrete set of 94 possible prices. Apps can be free, or priced as low as $0.99 and as high as $999.99. Apple collects 30% of all sales revenue, which is similar to other platforms (Spencer, 2015).[8] Figs. 1a and 1b show a snapshot of the distribution of prices in the App Store.

In addition to setting a price, developers can choose to include advertisements and/or in-app purchases (IAPs). IAPs are opportunities within an app to buy additional content or functionality. As an example, the task management app OmniFocus offers an IAP for the "Pro" version of the product, which includes additional task organization capabilities. As with app sales revenue, Apple

[7] Apps can be acquired illegally through online piracy. Piracy is relatively limited in this industry, particularly for the iOS platform and for the genre of apps considered in this paper, and is not considered here.

[8] The one exception to this policy, introduced in September 2016, reduces Apple's share of subscription revenue to 15% on all subscriptions older than a year (Goode, 2016). This change occurred after the sample period considered in this paper.

collects 30% of all IAP revenues. Apps are also allowed to offer subscriptions through Apple's payment system or through other means (e.g., a website), but any subscription bought *within* the app must be purchased through Apple's system.

In the early years of the App Store, many developers offered two versions of the same product, a free and a "paid" version. The free version would either be limited in some way, or would contain advertisements. This was, in general, either an attempt to engage in price discrimination or an attempt to create a rudimentary "Free Trial" option for users. This has largely fallen out of fashion, particularly among non-game apps.

App production is primarily characterized by the fixed costs associated with designing and programming the app. In almost all cases, app developers face zero marginal costs when selling their app. This is both a function of the digital nature of these products, and the fact that Apple handles all of the "back end" requirements of selling an app: managing the computer servers that host the files that consumers download, processing all transactions on the platform, and managing the (digital) storefront.

2.2 Existing Research on Smartphone Applications

This paper contributes to a small, but growing literature on smartphone apps. Early work was largely descriptive, but research has since expanded to studying cross-platform entry decisions (Bresnahan, Orsini, and Yin, 2014; Liu, 2017; Nekipelov, Park, and Liu, 2013), consumer search (Ershov (2017)), firm strategy (Bresnahan, Li, and Yin (2016), Davis, Muzyrya, and Yin (2014), and Yin, Davis, and Muzyrya (2014)), and the relationship between mobile platforms and developers (Gans (2012))

There has been some research on the demand for apps and on app updates, though to my knowledge, no paper has explicitly considered the effect of updates on consumer utility as this paper does. Yin, Davis, and Muzyrya (2014) find that the level of updating by the most successful apps varies by category. Specifically, highly successful game apps update infrequently, while the most successful non-game apps update more frequently. Ghose and Han (2014) estimate a demand model in the style of Berry, Levinsohn, and Pakes (1995) across the App Store and the Google Play Store. They use a nested-logit model, where consumers first choose whether to download a Free or Paid app, then a category of apps, and finally which app to buy. Using this model, Ghose and Han

find evidence that consumer demand is higher the more recently an app has been updated.

Of particular relevance to the issues studied in this paper is Comino, Manenti, and Mariuzzo (2016), which considers the effect of app updates on the growth rate of downloads. They find that the growth rate of downloads is positively affected by updates on Apple's platform, while there is no effect in the Google Play Store – a difference they attribute to Apple's strict app review process, which is interpreted as having a positive effect on the overall quality of the apps and app updates on Apple's platform compared to Google's. My paper builds on this idea of app updates affecting demand by directly estimating how updates affect consumer utility, and by modeling the supply-side decision of when and how to update an app.

3 A Structural Model of App Demand and Updating

In this section, I develop a model of consumer and developer behavior in the smartphone application (app) industry. Each period developers choose whether, and if so, how, to update their app. Developers earn revenue through both an extensive margin of demand, which captures consumers' app purchase decisions, and an intensive margin of demand, which captures consumers' app use decisions. Extensive-margin revenue comes from new purchases, while intensive-margin revenue can come from three possible sources: in-app purchases (IAPs), advertisements (ads), and subscriptions.

On the demand side, consumers choose each period whether to purchase an app in a given market (extensive margin demand). Consumers make static decisions, but are forward-looking in their purchase decisions, and consider the expected lifetime value of owning an app. At the same time, consumers choose each period whether to use any of their previously purchased apps (intensive margin demand). This additional margin of demand is important because so many apps earn revenue (in whole or in part) from consumers' intensive-margin *use* decisions. Due to a lack of available data on consumers intensive-margin behavior I am unable to estimate a structural model of the intensive margin of demand.[9] Instead, I use a reduced form approximation of an app's total revenue when estimating the supply-side app updating model.[10]

There is relatively little within-app variation in the monetization strategies used by developers,

[9] In Appendix C I outline a basic model of consumer behavior on the intensive margin, which is used to calculate consumers' lifetime expected utility from purchasing an app.

[10] See Section 4.1 for a discussion of how I approximate apps' per-period revenue, and Section 5.2 for a how I estimate the supply-side model.

and so I assume the choice of how to monetize an app, including the price of the app, is determined when the app enters the App Store, and is otherwise fixed. This paper abstracts from the monetization strategy entry decision. Additionally, multi-app developers are assumed to treat the development of each app separately. The cost of this assumption is that this model will fail to account for any economies of scope in the production of apps. Only 8% of developers in the sample have more than one app in a given market, and of the developers that do have multiple apps in a market, 98% have only two apps.

3.1 Timing

Time in this model is discrete, and I index each period t. In my empirical application of this model a period is a week. The timing for each period is as follows: At the beginning of a given period t, developer j observes the current state of the market, S_{jt}, and chooses whether, and if so, how, to update their apps. Developers face a choice of whether to not update their app, or to create a Major or Minor product update. As discussed in Section 2, Apple then reviews the update submission, and propagates the update to all previous purchasers of app j following that review.[11]

Following the updating decision and propagation of the update to consumers, market-level demand-shocks are realized, and consumers observe their individual consumer-app match values. Observing these demand shocks, consumers make their extensive margin decisions of what app, if any, to purchase, and their intensive margin decisions of what previously purchased apps, if any, to use. Following consumers' purchase and use decisions, developers receive the revenue they earned that period, and the next period begins.

3.2 A Model of App Demand

As discussed above, consumers face two separate decisions each period: whether to purchase a new app, and whether to use any already owned apps.

Extensive Margin of Demand On the extensive margin of demand, consumer i receives utility u_{ijt} from purchasing app j in period t, and chooses the app that provides the greatest utility. In doing so, the consumer considers the value of future use of the app and the potential for any future

[11] I ignore the possibility that Apple might reject an update.

updates to the app. Specifically, the utility from app j, u_{ijt} is

$$\begin{aligned} u_{ijt} &= X_{jt}\beta + \alpha p_{jt} + \xi_{jt} + \Lambda_{ijt} + \epsilon_{ijt} \\ u_{i0t} &= \epsilon_{i0t} \end{aligned} \tag{1}$$

and u_{i0t} is the utility from the outside option of not purchasing an app. X_{jt} is a vector of observable app characteristics, which includes the app's file size and age-appropriateness rating, whether or not the app includes IAPs, ads, and subscriptions, and month and iOS version fixed effects. p_{jt} is the price of the app. ξ_{jt} is the mean unobservable consumer utility from the app and represents the utility that results from unobserved (to the econometrician) characteristics of the app. Λ_{jt} is the expected, discounted future utility from app j, capturing the fact that consumers anticipate both future use of the app, and future updates to the app, and account for these values when making a purchase decision. Finally, ϵ_{ijt} is a period-specific consumer-app match value. I assume that $\epsilon_{ijt} \sim$ i.i.d. Type I Extreme Value, which admits a closed form solution for the share of consumers purchasing app j at time t, s_{jt}.

$$s_{jt} = \frac{\exp(X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \xi_{jt})}{1 + \sum_k \exp(X_{kt}\beta + \alpha p_{kt} + \Lambda_{kt} + \xi_{kt})} \tag{2}$$

Mean Unobservable Consumer Utility (ξ) Even within the same market, apps are highly differentiated, and, while there are some app characteristics that are observable to the econometrician, many key features are not. These unobserved characteristics are captured in the demand model by ξ_{jt}. ξ_{jt} captures a wide variety of unobserved characteristics including primary functions of an app, subtle features that differentiate it from its competitors, as well as other, broader characteristics that might affect a consumer's demand decision, such as whether it is particularly easy to find through the App Store's user interface.

Some of the unobserved characteristics captured by ξ_{jt}, such as specific functionality of the app, will not vary period-to-period, or, at a minimum, will not vary period-to-period in an i.i.d. way. For example, a note-taking app would not offer a search feature one period, remove it the next, and then reinstate it in a third period. Because of this, I assume ξ_{jt} follows an AR-1 process, expressed as

$$\xi_{jt} = \rho\xi_{jt-1} + \eta_{jt}$$

where the unobserved quality of the app is correlated over time according to ρ, but is still subject to i.i.d., mean-zero shocks η_{jt}.

While the process above captures the fact that unobserved product characteristics will be fairly consistent period-to-period, it does not account for the fact that when a developer updates an app, ξ_{jt} is likely to change in a larger way than it typically would. To capture this, I model app updates as directly affecting ξ_{jt} by assuming that ξ_{jt} undergoes a one-time, vertical shift as the result of an update. Namely, assume ξ_{jt} follows the law of motion

$$\xi_{jt} = \rho\xi_{jt-1} + \mu_{Minor}1(a_{j,t} = \texttt{Minor}) + \mu_{Major}1(a_{j,t} = \texttt{Major}) + \eta_{jt} \tag{3}$$

where $\mu = (\mu_{Minor}, \mu_{Major})$ captures the effect of the developer's update decision a_{jt}.

Note that unpacking the variety of possible mechanisms captured by ξ, while potentially interesting, is not necessary in order to answer the questions considered in this paper. All that is necessary in order to see whether consumers respond to updates is that the net effect of updates on consumer behavior is captured.

Expected, discounted future utility (Λ) When making purchase decisions, consumers have expectations over the likelihood of future updates, and over future consumer-app match values which are relevant to the consumer's period-specific, intensive margin, use decision. In order to construct a measure of consumers' lifetime expected utilities of owning an app I assume that consumers make independent, binary choices of whether to use each of their previously purchased apps. In each period after purchasing an app the consumer will receive utility based on either using the app or choosing the outside option of not using that app. Therefore, I define the lifetime expected utility of owning app j, purchased in period t, Λ_{jt}, as the discounted sum of the per-period expected utility of owning an app, with expectations over the developer of app j's future updating decisions. This takes the form

$$\Lambda_{ijt} = E\left[\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \ln\left(1 + e^{X_{j\tau}\beta+\xi_{j\tau}}\right)\right] \tag{4}$$

where δ is the discount factor. See Appendix C for the derivation of Λ_{jt}.

3.3 A Model of App Updating

The revenue an app earns each period is a function of the app's monetization strategy, which is assumed to be fixed. Monetization strategies consist of some combination of setting a non-zero price, and including advertisements, subscriptions, and/or in-app purchases (IAPs) in the app. It is possible that some apps earn revenue in additional ways, such as through accompanying hardware or services that are not sold through the App Store, and/or by selling users' information to third parties (see Kesler, Kummer, and Schulte (2017)).

Apps in market m are indexed $j \in J_m$. Developers face a choice each period $t = 0, \ldots, \infty$ regarding whether, and if so, how, to update their app. There is a discrete set of potential updates so that the developer's per-period choice set is $A = \{\emptyset, \texttt{Minor}, \texttt{Major}\}$, with $\emptyset$ representing the choice to not update the app. Denote state k in period t from the perspective of firm j as S_{kjt}.

Developers choose $a_{jt} \in A$ for each period $t = 0, \ldots, \infty$ in order to maximize

$$\sum_{t=0}^{\infty} \beta^t \Big(\pi(a_{j,t}, S_{j,t}) + \theta^{\epsilon} \epsilon_{j,t}(a_{j,t}) \Big) \tag{5}$$

where the per-period profit function $\pi(a_{j,t}, S_{j,t})$ is

$$\pi_{j,t}(a_{j,t}, S_{k,j,t}; \theta^D) + \theta^{\epsilon} \epsilon(a_{jt}) = R_j(S_{k,j,t}, \theta^D)(1-\phi) - \theta^{Minor} 1(a_{j,t} = \texttt{Minor}) - \theta^{Major} 1(a_{j,t} = \texttt{Major}) + \theta^{\epsilon} \epsilon_{j,t}(a_{j,t}) \tag{6}$$

R_j is the revenue the app earns, θ^D is the parameters from the demand model developed in Section 3.2, θ^{Minor} and θ^{Major} represent the fixed costs of updating, and ϵ_{jt} is an i.i.d. Type I Extreme Value error, which captures the fact that apps may earn additional revenue in the given period that is not captured by the model and available data. ϕ is the share of revenues that the platform collects. In the case of Apple, $\phi = 0.3$. Note that marginal costs are assumed to be 0, as discussed in Section 2.

R_j is defined as

$$R_j(S_{kjt}, \theta^D) = \overbrace{p_j^{Retail} s_{j,t} M_{mt}}^{\text{Direct Sales Revenue}} + (\texttt{Intensive Margin Revenue}) \tag{7}$$

where intensive margin revenue can come from advertisements, in-app purchases, subscriptions, or any combination of the three. M_{mt} represents the market size of market m in period t.

Equilibrium When deciding whether to update their app, developers must weigh the current-period fixed cost of updating against present- and future-period revenues that would result from updating. I assume a Markov Perfect Nash Equilibrium (Ericson and Pakes, 1995). In such an equilibrium, strategy σ_j maps from a given state $(S_{k,j}, \epsilon_j)$ to an action a_j without dependence on t. Then, following Bellman's principle of optimality, the value to a firm in a given state can be expressed

$$V_j^{\sigma}(S_{k,j}, \epsilon_j) = \max_{a_j \in A}[\pi(a_j, S_{k,j}) + \theta^{\epsilon}\epsilon_j(a_j) + \beta \int V_j^{\sigma}(S_{k',j}) g(S_{k',q}|a, \sigma_{-j}, S_{k',j}) dS_{k',j}] \tag{8}$$

and, since ϵ is i.i.d. Type I Extreme Value, I can express the optimal strategy for j as a conditional choice probability:

$$P^{\sigma_j}(a, S_{k,j}, \sigma_{-j}) = \frac{\exp\left(\frac{\nu_j^{\sigma}(a, S_{k,j}, \sigma_{-j})}{\theta^{\epsilon}}\right)}{\sum_{a' \in A} \exp\left(\frac{\nu_j^{\sigma}(a', S_{k,q}, \sigma_{-j})}{\theta^{\epsilon}}\right)} \tag{9}$$

where $\nu_j(\cdot)$ is the choice-specific value function:

$$\nu_j^{\sigma}(a, S_{k,q}, \sigma_{-j}) = \pi(a, S_{k,j}) + \beta \int V_j^{\sigma}(S_{k',j}) g(S_{k',j}|a, \sigma_{-j}, S_{k',j}) dS_{k',j} \tag{10}$$

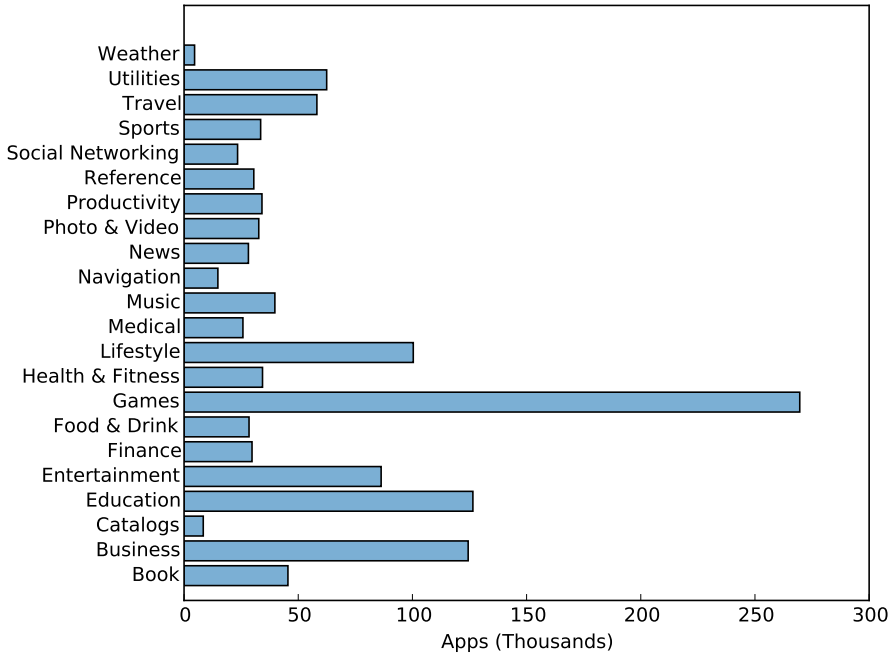
I assume a Markov Perfect Nash Equilibrium exists. Making this assumption is required due to the fact that this model has continuous state variables (e.g., download size, market size, and ξ_{jt}).[12]

4 Data

I collect daily app characteristic and ranking data from Apple's App Store, and sales data directly from iOS app developers. The full sample consists of all apps in the App Store from December 31, 2014 to June 29, 2016. I aggregate this daily data to the app-week level. In total, there were 1,241,910 iOS apps in the App Store on December 31, 2014 and 1,823,735 iOS apps in the App

[12] As noted in Sweeting (2013), it would be possible to discretize these continuous state variables in order to then apply the proof of Doraszelski and Satterthwaite (2010) that a pure strategy equilibrium exists.

Figure 2: Number of Apps by Category (12/31/2014)



Store on June 29, 2016, distributed across twenty-two categories (see Fig. 2).

I observe a number of characteristics for each app, including the app's name, price, file size (in megabytes), age-appropriateness rating, the day the app was first released, the version number of the app, the category classification of the app, and what devices the app is compatible with. I also observe a text description of each app and app update, both written by the developer. An example of an app description and an app update description, called release notes, are shown in Figs. 3 and 4, respectively. Finally, I use data on the presence of advertisements for apps in the Productivity category that has been collected by the app analytics company AppFigures.[13]

In addition to the characteristic data, I observe three daily ranking lists for each category, which order apps by sales. In particular, I observe the Top Free, Top Paid, and Top Grossing lists, which rank apps by either the quantity sold (Top Free and Top Paid) or by the apps' gross revenue (Top Grossing). I limit the full sample for this paper to those apps that rank 200^{th} or better on the Top Grossing list for at least 10% of my sample period. There are two reasons for this restriction.

First, apps can be broadly classified as either "companion" or "product" apps, where the former are apps that are companions to other products or services outside of the App Store (e.g., the Wells

[13]Section 4.2 describes the Major and Minor updates distinction that is referenced in Fig. 4

Figure 3: Example of an App Description

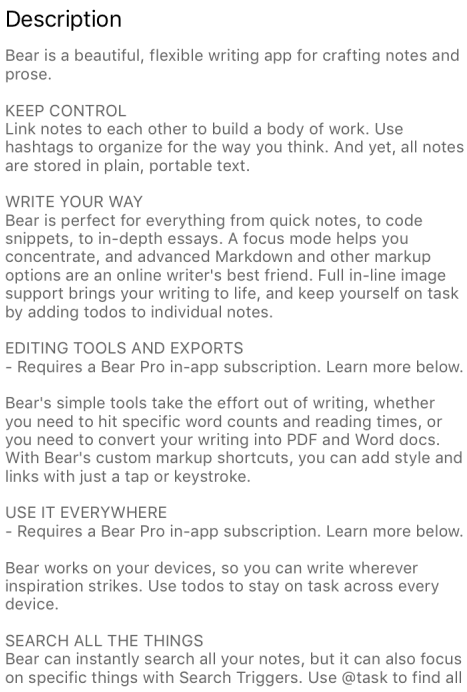
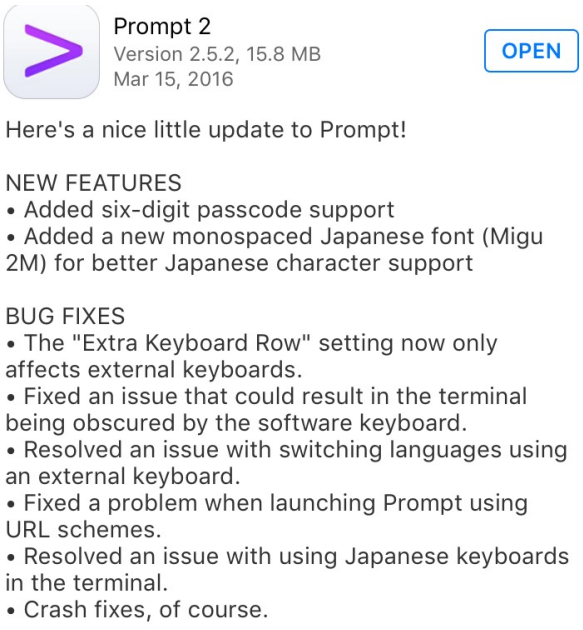
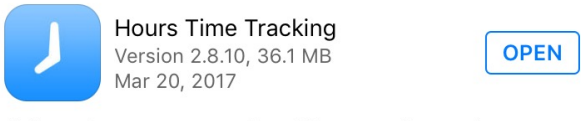


Figure 4: Update Type Examples



(a) Major Update



(b) Minor Update

Fargo or United Airlines apps), and the latter are apps that are a primary product being sold by the developer. Companion apps are excluded because in many cases their development is not profit-maximizing. For example, United Airlines may not make any revenue from its app, but is willing to invest in its development because the app supports United's ticket sales business. Second, because entry into the app store is relatively inexpensive, and the cost of keeping a poorly selling app on the app store is negligible, there are a large number of apps that have limited, or even zero, sales during the sample period. I limit my sample to apps that reach a ranking of at least 200^{th} on the Top Grossing list at least once allows me to avoid both companion and abandoned apps as much as is possible. See Sections B.1 and B.2 for further discussion of these issues.

For the estimation of the structural model, I further limit the sample to those apps that rank at least 200^{th} for at least 10% of the sample period. These restrictions result in a sample of 352 apps. A description of this sample is given in Section 4.4.

In order to estimate the model in this paper I must first address three issues in the data. First, in Section 4.1 I account for the fact that sales data is, in general, unavailable for this industry, then, in Section 4.2, I use natural language processing (NLP) and machine learning (ML) techniques to classify updates based on their content. Finally, in Section 4.3 I again use NLP and ML techniques to develop a method for defining markets in online product markets.

4.1 Estimating App Sales

As is common when estimating logit demand models, estimating my demand model (Eq. (1)) requires data on period-specific market shares. Here, as in many industries, sales data is proprietary, and direct data on market shares for my entire sample is unavailable. In studying apps and other online markets, this issue has been dealt with in two ways. Bajari, Fox, and Ryan (2008) develop a method for estimating demand directly from sales ranking data, but their method requires that observable product characteristics are sufficient for determining a consumer's preference ordering within a market. Given the importance of the unobservable characteristics represented by ξ in my model, and, in particular, the manner in which product updates are incorporated into the ξ process, adopting the estimation approach developed by Bajari, Fox, and Ryan is not appropriate in my case.

The second approach, which has been widely used in research on apps, is to first estimate a

relationship between an app's sales ranking and its level of sales using a limited source of information about app sales. Then this estimated relationship is used to predict sales for all products in all periods, and the demand model is estimated using market shares that are computed using these predicted sales values. This approach for mapping sales ranking data to sales quantity data in online markets was pioneered by Chevalier and Goolsbee (2003) and Brynjolfsson, Hu, and Smith (2003) in studying online book sales, and applied to app markets by Garg and Telang (2012), Ghose and Han (2014), and Ershov (2017), among others.[14] In most applications, the relationship between sales ranking and sales quantity is estimated using publicly available ranking data and a small (relative to the sample of apps being studied) amount of sales data either collected from various developers or from small, independent app marketplaces.[15]

Apple provides three types of sales ranking lists, at both the storewide and the category level. The exact algorithms for the ranking lists are unknown, but they roughly rank apps by sales.[16,17] The Top Free list ranks apps with a price of \$0.00 by sales quantity, the Top Paid list ranks apps with a non-zero price by sales quantity, and the Top Grossing list ranks apps with zero and non-zero prices by revenue.

I use the Top Free and Top Paid ranking lists at the category level to estimate sales because they directly account for apps' period-specific downloads, which is the necessary variable for computing the market shares used in the demand estimation. Following Chevalier and Goolsbee (2003), I assume sales ranking data follows a Pareto distribution. Specifically, given a set of apps $j \in J$ and some threshold level of sales $\bar{S}$,

$$\Pr[\text{sales}_{jt} > \bar{S}] = \left(\frac{k}{\bar{S}}\right)^{\beta} \tag{11}$$

[14] This approach has also been applied by app developers, see, e.g., Perry (2015)

[15] Instead of using rankings, Liu (2017) uses app ratings and/or reviews as a proxy for sales. An app rating is an integer between 1 and 5 that the consumer assigns an app, whereas a review is a rating accompanied by a textual comment on the product. This approach can be unreliable, though, as some developers actively prompt users for ratings and/or reviews while others do not.

[16] While Apple's internal policies regarding the ranking algorithm are not known, the view within the industry is that the lists do follow some explicit algorithm, which is not altered to benefit any particular app (e.g., for promotional reasons).

[17] Bresnahan, Li, and Yin (2016) study apps – primarily games – that attempt to game the ranking lists, though not by altering the ranking algorithm.

Table 1: Ranking-Sales Relationship Estimates (Top Paid List)

α	5.231*** (0.182)
β	0.424*** (0.025)

Includes month-year fixed effects. Standard errors are in parenthesis. R-squared=0.760.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Given a sufficient number of apps, this can be rewritten as

$$\frac{\text{rank}_{jt}}{|J|} = \left(\frac{k}{\bar{S}}\right)^{\beta} \tag{12}$$

Taking logs, the assumption of a Pareto sales ranking distribution provides the estimable relationship

$$\ln(\text{rank}_{jt}) = \overbrace{\big(\beta \ln(k) + \ln(|J|)\big)}^{\alpha} - \beta \ln(\text{sales}_{jt}) \tag{13}$$

I estimate Eq. (13) using publicly observable ranking data, and sales quantity data I have collected directly from some of the developers in the sample. Table 1 shows the results of estimating the rank-sales relationship for the Top Paid ranking list. Due to data limitations, I am only able to estimate Eq. (13) for the Top Paid ranking list in the Productivity category. In order to also calculate sales for the Free apps in the sample, I follow Ghose and Han (2014) in assuming that the shape, β, of the relationship between an app's ranking and its sales is the same for both the Free and Paid list, but that the scale parameter, α, can differ. I calibrate $\alpha_{Free} = 7.3$ using sales data collected from developers' that produce Free apps.[18]

My finding of $\beta = 0.424$ is lower than what is typically found in the literature. Garg and Telang (2012) summarize seven studies of this relationship in non-app industries, and find that estimates range from 0.613 to 1.2. In app markets, Ghose and Han (2014) estimate $\beta = 1.09$, and Ershov (2017) estimates the parameter to be 1.168 for games and 0.926 for non-games.

There are two possible reasons for the difference between my estimates and those found in the literature, particularly those of Ghose and Han (2014) and Ershov (2017). First, my estimates are the only ones calculated using sales data from Apple's App Store, and the shape of the ranking-

[18]The results of this paper are robust to using different values for this parameter within a reasonable range of 7.3.

Table 2: Ranking-Revenue Relationship Estimates (Top Grossing List)

α	7.1016 (0.204)
β	0.4039 (0.031)

Includes month-year fixed effects. Standard errors are in parenthesis. R-squared=0.760.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

sales relationship may differ between platforms. Ghose and Han (2014) use data from a third-party (i.e., not Google nor Apple) app marketplace, and Ershov (2017) estimates the relationship for the Google Play Store using the lower bound of sales ranges the store provides publicly.[19]

A second reason is that the shape of the ranking-sales relationship may be changing over time. Using data on Amazon's book sales, Brynjolfsson, Hu, and Smith (2010) find evidence that the shape parameter has decreased in magnitude over time, as weight has shifted to the tail of the distribution. A similar change may be occurring in app markets. The sample period for this paper (12/31/2014 to 6/29/2016) is more recent than other papers that have estimated this relationship, so a smaller estimate of $|\beta|$ would be consistent with the findings of Brynjolfsson, Hu, and Smith (2010). Currently, data limitations restrict my ability to further investigate whether the shape of the ranking-sales relationship has in fact changed over time in the App Store.

Given the results in Table 1, sales quantities are estimated for all apps in the sample based on their ranking each period. For the remainder of this paper, any reference to an app's sales quantity (or market share) should be viewed as a reference to that app's estimated sales quantity (or market share). Additionally, I estimate this model for apps' per-period revenues using the Top Grossing list, which are shown in Table 2. These estimates are used in the estimation of the supply model (see Section 5.2).

4.2 Classifying App Updates

In order to understand how consumers respond to app updates, it is important to account for the differences between specific updates. Updates serve a variety of purposes – from totally re-inventing an app to fixing a small typo in a rarely-used part of an app. Developers provide text descriptions

[19] For example, an app with cumulative sales of 45,000 is publicly listed as having "10,000 to 50,000" sales. See (Ershov, 2017, Table 15) for the sales ranges provided.

of each update, called release notes, which outline the specific changes included in a particular update.

Based on these descriptions, I classify each update as either a minor, bug-fixing update, or a major, feature-adding update. Conceptually, Minor updates include changing the app so that it no longer crashes whenever the user performs a certain action, updating the app so that it is able to run on new devices or operating systems, and implementing "under-the-hood" changes, which involve cases where the developer makes changes primarily for the developer's own benefit – for example, by changing the underlying database system to something that the developer finds easier to work with, but that the consumer would be unaware of. Major updates are those that add additional, user-salient functionality or content to the app. For example, adding voice dictation to a note-taking app, or adding audio pronunciations to a dictionary app would be Major updates.

Prior to classifying the updates, I preprocess the updates' release notes using NLP techniques to prepare them for the SVM algorithm. Each update's release notes is tokenized and lemmatized, which reduces each word to its dictionary form, and all punctuation and non-letter characters (e.g., bullet points and emojis) are removed.[20] I remove a standard list of common English "stop words," such as "a" and "the," as well as a list of words related to Apple and its products, such as "iPhone" and "iOS," from the descriptions. Removing these words allows me to focus my analysis on the words that indicate the purpose of app updates, rather than, for example, the frequency with which a developer mentions Apple's platform.

In order to categorize updates using their release notes, I train and use a support vector machine (SVM). An SVM is a form of supervised machine learning that trains a model using a "training set" of pre-classified app updates. Once trained, the model can take other update release notes as an input and return an estimated update type. In order to train the SVM, I classified 782 app updates by hand. Fig. 5 outlines the set of rules under which the training set was built.

Additionally, I optimally select a number of other options, including allowing for multi-word phrases ("n-grams"), removing words that only appear a few times across all descriptions (such as proper nouns), and removing any words that appear in a large percentage of the descriptions. The parameters for these options are chosen by iterating over a parameter grid, and choosing the best

[20] Tokenization is the practice of reducing a document to a set of individual word "tokens," and lemmatization is the practice of reducing tokens, i.e., words, to their dictionary form. E.g., "drinks", "drank", and "drinking" all reduce to "drink."

Figure 5: Update Classification Training Rules

Major Update

- Adds additional functionality.
- Adds additional content.
- Adds support for a new language.

Minor Update

- Any direct mention of fixing bugs.
- Performance improvements or anything indicating "under-the-hood improvements."
- Adjustments to maintain compatibility with the latest version of iOS.
- Changes to existing functionality that imply minor improvements to existing functionality, not the addition of more functionality.

Table 3: Update Classification Precision and Recall

	Precision	Recall	N
Minor	0.91	0.91	185
Major	0.78	0.77	73
Average/Total	0.87	0.87	258

Precision indicates the ratio of the number of correctly predicted cases for a given update type to the total number of cases predicted to be of that type. Recall is the ratio of the number of correctly predicted cases for a given update type to the total number of cases of that type.

vector of parameters by maximizing a scoring metric. Ultimately, this process produces a "bag of words" for each description.

Given the bag of words for each description, I map each document to a vector space where each element of a vector represents a particular word or n-gram. Rather than simply using an indicator for which words are present in a document, I use term frequency, inverse-document frequency weighting. This weighting procedure places greater emphasis on words that appear frequently in a document relative to the word's overall frequency in the set of release notes, under the assumption that such words are a more valuable signal for differentiating a particular set of release notes than words that appear frequently across all documents. The weight for word i in release notes j is

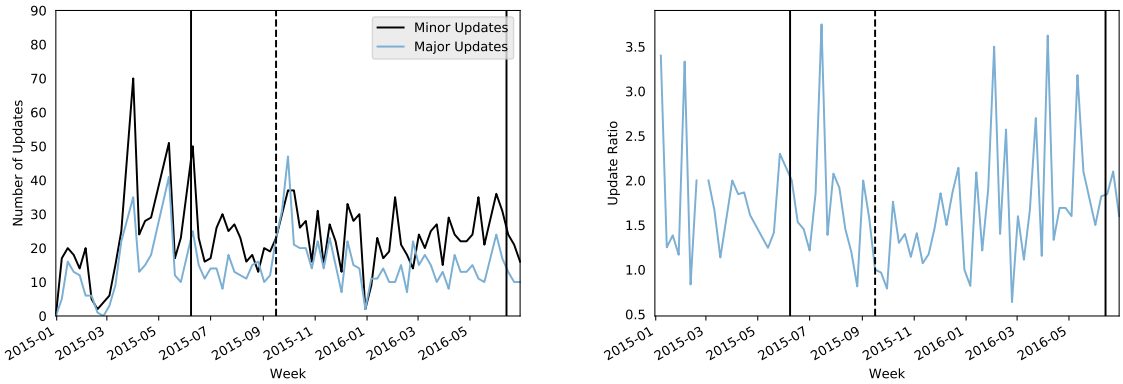
$$weight_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \times \ln\left(\frac{N}{n_i}\right) \tag{14}$$

where f_{ij} is the frequency of word i in document j, N is the total number of release notes (i.e., the total number of updates), and token i appears in n_i of the release notes. This process results in a set of sparse, high-dimensional vectors representing each release note.

Two-thirds of the trained release notes are used to train the SVM, and the remaining third is used to test the performance of the machine. The results of these tests are shown in Table 3. Fig. 6a shows the number of Major and Minor updates over the sample period, and Fig. 6b shows the ratio of Major to Minor updates. In general, Minor updates outnumber Major updates, but there is time variation in the ratio of the two types of updates and at times Major updates outnumber Minor updates. For example, just following the release of a new model of the iPhone (indicated by the dashed vertical line), there is a spike in the number of Major updates, which outnumbers Minor

Figure 6: Updates By Type

SVM Classification



(a) App Updates by Type

(b) Ratio of Minor to Major Updates

The vertical lines mark the beginning (solid) and end (dashed) of the period in which Apple offers developers access to a pre-release version of their next major platform upgrade. Many developers hold off on updating their apps in anticipation of the release of the new version of the platform.

updates in that period.

4.3 Defining Markets

In addition to accounting for app sales, estimating the model developed in Section 3 requires defining markets within the Productivity category of the App Store. Properly defining the relevant market for a product has long been a challenge in empirical market analyses. Markets for digital goods provide additional challenges. First, the existence of a "long tail" of products in, for example, app markets, online book markets such as Amazon.com, and online music markets such as iTunes, dramatically expands the potential choice set for consumers. Second, the wide variety of monetization methods used in a particular app market make it difficult to develop and employ systematic methods of market definitions.

Traditional, systematic methods for defining markets such as the commonly employed "small but significant and non-transitory increase in price" (SSNIP) test are especially difficult to develop and apply due to the challenge of calculating the elasticity of demand when many products are priced at $0.00 Evans (2011). Approaches that do not involve price elasticity calculations are also difficult given the variety of ways products are monetized in this industry. Examples of this include the recently suggested "small but significant and non-transitory increase in *(exchanged) costs*"

(SSNIC) test, where exchanged costs represent exchanges of information or attention in return for a product, and the "small but significant and not-transitory decline of quality" (SSNDQ) test, which was employed in a 2013 case before the Chinese Supreme People's Court Newman (2015). It is not clear, for example, how to define a product's exchanged costs when the firm employs multiple forms of monetization, or even when a firm employs a single form of monetization, but has the *ability* to employ multiple forms of monetization.[21] That is, estimating the SSNIC associated with a paid-upfront price increase of x would have to account for the endogenous response of the product's advertisement level, number and prices of in-app purchases, and/or its subscription price. This would still be abstracting from the endogeneity of *which* forms of monetization a particular product uses. While such an abstraction is reasonable in a typical, non-digital industry, where all firms charge for a product at the point of purchase, they are less reasonable for app markets given the varied manner in which the products are monetized. Finally, accounting for all of this would be difficult in terms of modeling, and also computationally difficult given the large number of potential products available.

Previous Approaches for Defining Markets in the App Industry In the literature on smartphone apps, markets have generally been defined by adopting the categories used within the marketplaces themselves. That is, in Apple's App Store, each of the 22 consumer-facing categories would each be considered a market.[22]

This approach of adopting the platforms' categories is overly broad and can suggest competition where there is none. For example, within Apple's Productivity category, there are a number of apps focused on helping a user manage their photos on the photo-sharing app Instagram, and there are a number of apps focused on writing computer code on mobile devices. While some consumers may use both, the apps serving one of these two purposes are unlikely to be in direct competition with each other. In order to properly characterize consumer choice, and model firms' strategic behavior in these markets, it is important to define markets in such a way as to properly account for demand-side substitution that occurs within markets.

[21] See Newman (2015), particularly pages 66 and 67 for a discussion of implementing the SSNIC test. Notably, though, this discussion implicitly assumes that the monetization strategy used by firms is fixed, whereas in many industries, including apps, it is not.

[22] There are, to some extent, sub-genres defined within the App Store, but these are primarily used for Game apps, which are not considered in this paper.

One improvement on this market definition approach is that of Ghose and Han (2014), who offer a more detailed approach by using a nested structure for markets, where consumers first choose whether to buy a paid or free app, and then which category to shop in. However, this too fails to account for the large amount of heterogeneity within product categories.

I define markets by combining apps' categorical classification with additional information provided by the apps' developers. Within the Productivity category I cluster apps using an unsupervised machine learning algorithm based on the text descriptions provided by the developers. These descriptions are a prominent part of apps' listings on the App Store, and thus developers have a strong motivation to provide a clear description of their product.

Defining Markets via Clustering I use a combination of natural language processing techniques and unsupervised clustering algorithms to define markets. First, all app descriptions undergo the same text pre-processing process outlined in Section 4.2. The specific parameters of this process are separately determined for the market definition algorithm.

Once the app descriptions have been processed and vectorized, I define markets using k-means clustering. Given a set of description vectors $X = \{x_1, \ldots, x_N\}$, and an exogenously determined number of clusters k, k-means clustering allocates each app description $x \in X$ to one of the k clusters, characterized by a centroid in the vector space defined by X. Broadly, x_i is in cluster j if and only if it is more similar to the x's in j than to those in any other cluster.

More precisely, define

$$\gamma_{ij} = \begin{cases} 1, & x_i \text{ is in cluster } j \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

and let δ_j be the centroid of cluster j. Letting $\gamma = \{\gamma_{ij}\}$ and $\delta = \{\delta_j\}$, the k-means clustering algoritm then solves

$$\min_{\gamma,\delta} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{k} \gamma_{ij} ||x_i - \delta_j||^2 \tag{16}$$

Eq. (16) is solved using a recursive, two-step process. In the first step it optimally selects γ given δ, and in the second step it optimally selects δ given the γ determined in the first step. Specifically, the objective in each step is

$$\text{Step 1:} \qquad \min_{j'} ||x_i - \delta_{j'}||^2 \ \forall i \tag{17}$$

$$\text{Step 2:} \qquad \sum_{i=1}^{N} \gamma_{ij}(x_i - \delta_j) = 0, \ \forall j \tag{18}$$

This two-step process is repeated until the convergence of γ and δ. The optimal number of markets (i.e., clusters) and the optimal parameters for the text pre-processing stage are determined by repeating the clustering algorithm across a parameter grid and then selecting the vector of parameters that maximize a scoring metric. In particular, I use the silhouette score, which provides an average measure of how well each app matches with its market, compared to how it matches with the other markets (Rousseeuw, 1987).

4.4 Descriptive Evidence

Restricting the sample to apps that ranked at or better than 200^{th} for at least 10% of the sample periods results in 352 apps across 39 markets. Fig. 7 shows four of the markets determined by the k-means clustering algorithm. There are some cases where human judgement might be at odds with the algorithm's market choices, but the market definitions generally appear to be reasonable.[23]

Fig. 8 shows update levels by market. Importantly, there is a wide level of variation across markets in both the level of updating and the relative level of Major and Minor updates. Table 4 provides summary statistics for each market. While in some markets all apps use the same monetization strategy (see markets 5 and 30 as examples), most markets show a mix of approaches. Prices in the sample range from Free to $119.99, with a mean price of $4.66, and a median price of $0.99. The distribution of prices for this sample is shown in Fig. 9. Just under half (49%) of the apps in the sample are free to purchase.

Finally, while many apps in the App Store engage in use monetization, many others do not. This provides natural treatment and control groups for developing an initial understanding of how the ability to monetize a product's use affects developers' updating behavior. Of course, a natural

[23] One example of this is market 0, which consists of one app, "Tody." Tody is a "smarter to-do list for managing household cleaning routines," and while the specific emphasis on household cleaning (as opposed to more general to-do list purposes) appears to be why it has been allocated its own market, it's likely that defining markets by hand would result in it being grouped in with other task management apps. Note that market 0 is dropped from the analysis when estimating the structural model in Section 5 because Tody has a monopoly on its market.

Figure 7: Example Markets

Drawing Desk
Paint
Doodle
Sketch
Tayasui Sketches
Graphic

(a) Market 36 - Drawing

VPN Express
TunnelBear VPN
Astrill VPN Client
VyprVPN
HideMyAss! Pro VPN
VPN Unlimited
VPN Seed4.Me
Unlimited Free VPN
VPN Master

(b) Market 5 - Virtual Private Network

5000 Followers Pro
FBLikes
5,000 Followers for Instagram
TwitterBoost
Followers Jackpot
Gain Followers
Ins Followers
Mega Followers
Follower Boost for Instagram

(c) Market 34 - Instagram Management

Groups
Simpler Contacts Pro
Smart Merge Pro
Cleaner Pro
Text 2 Group Pro
Easy Backup
Simpler Contacts
Cleaner
Smart Merge
Text 2 Group
Cleaner Master for iOS

(d) Market 4 - Text Messaging

Figure 8: Update Types, By Market

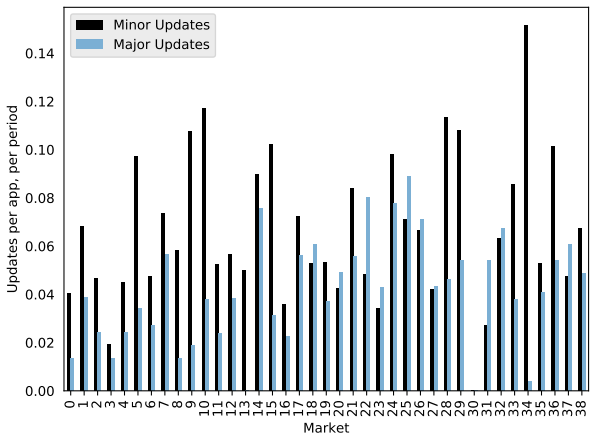


Figure 9: Distribution of Prices in Estimation Sample (On 12/31/14)

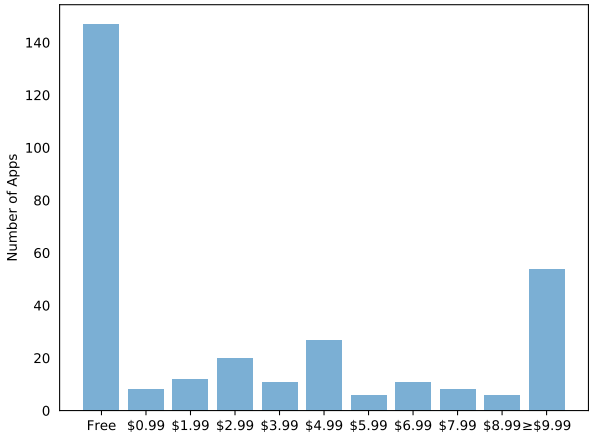


Table 4: Summary Statistics

		Monetization												**Updates**		**Characteristics**				
		Price				Price (Paid apps)								SVM			Age Appropriateness Rating			
Market	Apps	Min	Mean	Median	Max	Min	Mean	Median	Max	Free	IAP	Ad	Subs	Minor	Major	ln(size)	4+	9+	12+	17+
0	1	0.00	3.68	3.99	4.99	2.99	3.84	3.99	4.99	0.041	1.000	0.000	0.000	0.041	0.014	2.539	1.000	0.000	0.000	0.000
1	32	0.00	6.61	3.99	49.99	0.99	9.37	4.99	49.99	0.295	0.552	0.000	0.075	0.070	0.039	2.375	0.885	0.066	0.017	0.032
2	5	0.00	12.50	11.99	29.99	3.99	14.32	11.99	29.99	0.127	0.177	0.000	0.000	0.048	0.025	2.753	1.000	0.000	0.000	0.000
3	7	0.00	3.16	2.99	9.99	1.99	4.40	3.99	9.99	0.282	0.424	0.000	0.000	0.019	0.014	2.633	0.720	0.144	0.136	0.000
4	11	0.00	1.26	0.00	4.99	0.99	3.07	2.99	4.99	0.591	0.876	0.000	0.003	0.045	0.024	3.027	1.000	0.000	0.000	0.000
5	9	0.00	0.00	0.00	0.00	—	—	—	—	1.000	1.000	0.000	0.389	0.100	0.034	2.271	0.867	0.013	0.000	0.120
6	23	0.00	8.60	6.99	119.99	0.99	13.73	9.99	119.99	0.374	0.542	0.000	0.103	0.048	0.028	3.376	0.914	0.000	0.000	0.086
7	30	0.00	11.31	9.99	49.99	0.99	12.14	9.99	49.99	0.069	0.272	0.000	0.000	0.075	0.057	3.535	0.890	0.034	0.000	0.076
8	8	0.00	0.93	0.00	3.99	1.99	2.93	2.99	3.99	0.683	0.764	0.165	0.000	0.055	0.012	2.621	0.406	0.000	0.358	0.236
9	8	0.00	0.00	0.00	0.00	—	—	—	—	1.000	1.000	0.131	0.822	0.109	0.020	2.041	0.760	0.000	0.000	0.240
10	13	0.00	0.38	0.00	4.99	2.99	3.38	2.99	4.99	0.887	0.978	0.087	0.309	0.118	0.039	2.856	0.839	0.000	0.161	0.000
11	8	0.00	5.78	4.99	12.99	2.99	7.72	6.99	12.99	0.251	0.376	0.000	0.007	0.053	0.024	3.256	0.731	0.000	0.000	0.269
12	5	0.00	2.76	0.00	9.99	3.99	6.27	4.99	9.99	0.560	0.465	0.000	0.233	0.060	0.038	2.493	0.975	0.000	0.025	0.000
13	2	0.00	0.00	0.00	0.00	—	—	—	—	1.000	1.000	0.000	0.000	0.051	0.000	2.673	1.000	0.000	0.000	0.000
14	19	0.00	3.50	0.00	16.99	0.99	7.95	8.99	16.99	0.560	0.669	0.000	0.396	0.091	0.076	3.820	0.730	0.000	0.000	0.270
15	7	0.00	0.00	0.00	0.00	—	—	—	—	1.000	1.000	0.589	0.000	0.101	0.031	2.505	0.271	0.000	0.729	0.000
16	3	0.00	1.00	0.00	2.99	2.99	2.99	2.99	2.99	0.667	0.667	0.000	0.000	0.036	0.023	2.104	1.000	0.000	0.000	0.000
17	11	0.00	0.31	0.00	3.99	2.99	3.31	2.99	3.99	0.907	0.907	0.000	0.610	0.073	0.057	3.325	1.000	0.000	0.000	0.000
18	9	0.00	7.93	4.99	39.99	1.99	12.09	6.99	39.99	0.344	0.783	0.000	0.261	0.054	0.061	2.715	1.000	0.000	0.000	0.000
19	6	0.00	6.59	4.99	15.99	0.99	7.99	4.99	15.99	0.176	0.176	0.000	0.000	0.055	0.038	2.739	1.000	0.000	0.000	0.000
20	13	0.00	4.93	4.99	9.99	0.99	5.96	4.99	9.99	0.172	0.172	0.000	0.077	0.042	0.049	3.016	0.733	0.000	0.089	0.178
21	19	0.00	3.05	2.99	8.99	0.99	4.92	4.99	8.99	0.380	0.646	0.000	0.254	0.084	0.057	2.959	1.000	0.000	0.000	0.000
22	6	0.00	18.84	4.99	99.99	0.99	22.84	6.99	99.99	0.175	0.510	0.000	0.009	0.047	0.082	3.312	1.000	0.000	0.000	0.000
23	8	0.00	0.43	0.00	19.99	0.99	7.58	4.99	19.99	0.943	0.975	0.000	0.983	0.034	0.042	2.738	0.687	0.156	0.000	0.156
24	4	0.00	1.27	0.00	9.99	0.99	5.07	4.99	9.99	0.750	1.000	0.000	0.750	0.098	0.078	2.995	1.000	0.000	0.000	0.000
25	4	0.00	6.00	0.00	24.99	2.99	12.16	2.99	24.99	0.507	0.507	0.000	0.075	0.072	0.089	3.111	1.000	0.000	0.000	0.000
26	15	0.00	3.25	0.00	9.99	1.99	7.15	4.99	9.99	0.545	0.618	0.000	0.043	0.066	0.072	2.827	0.254	0.000	0.000	0.746
27	5	0.00	4.43	0.00	19.99	3.99	9.01	7.99	19.99	0.508	0.593	0.000	0.319	0.041	0.044	3.091	0.992	0.000	0.008	0.000
28	11	0.00	0.15	0.00	1.99	0.99	1.56	1.99	1.99	0.901	1.000	0.027	0.199	0.114	0.047	3.162	0.799	0.000	0.000	0.201
29	2	7.99	8.99	8.99	9.99	7.99	8.99	8.99	9.99	0.000	1.000	0.000	0.628	0.108	0.054	3.631	1.000	0.000	0.000	0.000
30	3	7.99	12.66	9.99	19.99	7.99	12.66	9.99	19.99	0.000	0.000	0.000	0.000	0.000	0.000	-0.029	1.000	0.000	0.000	0.000
31	2	0.00	2.33	0.00	4.99	1.99	4.87	4.99	4.99	0.521	1.000	0.000	0.000	0.028	0.056	3.054	1.000	0.000	0.000	0.000
32	6	0.00	2.03	2.99	4.99	0.99	3.96	3.99	4.99	0.487	0.487	0.000	0.005	0.049	0.073	1.721	1.000	0.000	0.000	0.000
33	10	0.00	3.42	0.99	19.99	0.99	6.79	7.99	19.99	0.497	0.698	0.000	0.378	0.086	0.038	2.935	0.899	0.000	0.000	0.101
34	9	0.00	0.00	0.00	0.00	—	—	—	—	1.000	1.000	0.495	0.146	0.152	0.004	2.283	0.507	0.000	0.493	0.000
35	6	0.00	3.22	1.99	9.99	0.99	4.27	2.99	9.99	0.246	0.507	0.000	0.101	0.053	0.041	2.034	1.000	0.000	0.000	0.000
36	3	0.00	0.59	0.00	2.99	0.99	2.94	2.99	2.99	0.800	0.800	0.000	0.000	0.097	0.054	4.071	1.000	0.000	0.000	0.000
37	4	0.00	2.28	0.00	8.99	8.99	8.99	8.99	8.99	0.747	1.000	0.000	1.000	0.048	0.062	2.842	1.000	0.000	0.000	0.000
38	5	0.00	2.21	0.00	9.99	9.99	9.99	9.99	9.99	0.779	0.798	0.000	0.596	0.068	0.049	3.028	0.617	0.000	0.000	0.383
Total	352	0.00	4.66	0.99	119.99	0.99	9.14	6.99	119.99	0.490	0.638	0.029	0.207	0.070	0.045	2.902	0.838	0.016	0.041	0.105

Observations are app-weeks. $N = 23,897$

Table 5: Updating Behavior and Use Monetization

	Update	Major
Use Monetization	0.0223***	-0.0416
	(0.0062)	(0.0227)
Price	0.0005	0.0024*
	(0.0003)	(0.0009)
App Age	-0.0000**	-0.0000
	(0.0000)	(0.0000)
Version Age	-0.0096***	0.0056*
	(0.0007)	(0.0027)
Version Age Squared	0.0001***	-0.0000
	(0.0000)	(0.0001)
Size	0.0002***	0.0009***
	(0.0000)	(0.0002)
Constant	0.2229***	0.1373**
	(0.0153)	(0.0495)
Recommended Age	Yes	Yes
N	17642	2615

concern with such an analysis is that there is likely selection on unobservables into use or non-use monetization. Furthermore, analyzing how use- and non-use monetizing apps differ does not capture the effects of firms' abilities to update products past the point of sale on their updating decisions. Thus, estimating the structural model developed in Section 3 and the counterfactual analysis conducted in Section 7 will be necessary to more definitively determine the effect of digitization on firm behavior.

In the sample of apps used for estimating the structural model, I find that apps that employ use monetization are 2.23 percentage points (17%) more likely to update in a given period. These results persist when controlling for the app's price, age, the age of the current version of the app, and the file size of the app (see the first column of Table 5). I also find that apps that employ use monetization are 7.23 percentage points less likely to submit a Major update conditional on updating. As the second column of Table 5 shows, I fail to reject the null hypothesis that use monetization has no effect on a developer's choice of whether to submit a Major or Minor update.

5 Estimating the Structural Model

I estimate the model in Section 3.2 using the sample and markets defined in Section 4.3.

5.1 Demand Estimation

Recall that the demand model (Eq. (1)) implies the market share of app j in period t can be expressed as

$$s_{jt} = \frac{\exp(X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \xi_{jt})}{1 + \sum_k \exp(X_{kt}\beta + \alpha p_{kt} + \Lambda_{kt} + \xi_{kt})}$$

In order to derive an estimating equation for the demand model, I transform market shares by taking the log of both sides Berry (1994). This gives

$$\ln(s_{jt}) - \ln(s_{0t}) = X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \xi_{jt} \tag{19}$$

In a typical application of a logit demand model, Eq. (19) can be estimated using ordinary least squares or two-stage least squares. However, because ξ follows the law of motion defined in Eq. (3), I first difference out $\rho\xi_{j,t-1}$. This step is necessary because app j's period t updating decision is likely to be a function of ξ_{jt-1}.

For clarity, let $y_{jt} = \ln(s_{jt}) - \ln(s_{0t})$ be the left-hand side of Eq. (19). Given this, and plugging in the ξ law of motion, the standard estimating equation, Eq. (19), can be re-written as

$$y_{jt} = X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \rho\xi_{jt-1} + \mu_{Minor}1(a_{j,t} = \texttt{Minor}) - \mu_{Major}1(a_{j,t} = \texttt{Major}) + \eta_{jt} \tag{20}$$

In order to estimate the parameters of the model, I *quasi-first-difference* this model, by subtracting ρy_{jt-1} from y_{jt}, giving

$$\begin{aligned} y_{jt} - \rho y_{jt-1} = (X_{jt} - \rho X_{jt-1})\beta + \alpha(p_{jt} - \rho p_{jt-1}) + (\Lambda_{jt} - \Lambda_{jt-1}) \\ + \mu_{Minor}1(a_{j,t} = \texttt{Minor}) - \mu_{Major}1(a_{j,t} = \texttt{Major}) + \eta_{jt} \end{aligned} \tag{21}$$

I estimate Eq. (21) using non-linear least squares (NLLS). In estimating the demand model, Λ_{jt} is calculated using forward simulation.[24].

Non-linear Price Disutility The model assumes that price has a linear effect on a consumer's utility. This is not necessarily the case. Anecdotal evidence suggests that while consumers may strongly dislike paying for an app, they may not be highly price sensitive conditional on paying for

[24]This process is discussed in more details in Section C.1

an app. In light of these concerns, I consider three specifications of p_{jt} when I estimate the model in order to account for any possible non-linearities in consumers' price sensitivity. Specification (1) includes the price p_{jt} of the app, and specification (2) uses an indicator variable for whether the price is greater than zero or not. Specification (3) includes both the "paid" indicator and the app's price.

Timing and Endogeneity As discussed in Section 2, Apple maintains a strict review process for all app updates. This process creates a delay between the creation of the update (and its submission to Apple), and the release of the update to consumers of about a week. Because I aggregate my data to the week level, I assume that the decision of whether, and if so, how, to update an app is made prior to the realization of η_{jt}. That is, η_{jt} is assumed to be independent of period-t updating decisions.

Price changes can be made independently from app updates, and are not subject to review, so the standard price endogeneity concern exists. This is dealt with using a two-stage least squares approach, where period t prices are predicted in a first-stage regression using current and one-period lagged covariates.[25] This is similar to the approach taken in Doraszelski, Lewis, and Pakes (2016).

5.2 Fixed Costs Estimation

I estimate the app updating model using the methodology of Bajari, Benkard, and Levin (2007), henceforth referred to as BBL. BBL propose a two-stage estimator. In the first stage, state transition probabilities and the conditional choice probabilities for firms' updating decisions are estimated. These are then used to approximate the firms' value functions under both the observed equilibrium policies and a variety of alternative policies using forward simulation. In the second stage, the approximated value functions are used to estimate the dynamic parameters of the model.

First Stage In the first stage of the supply estimation, I estimate the state transition probabilities for the state variables that vary over time, estimate a reduced form model for approximating apps' revenues, and estimate reduced form policy functions for apps' updating decisions. Table 6 provides a list of state variables in the model, and a description of their evolution process.

[25] More precisely, let the set of covariates $X_{jt} = (X^1_{jt}, X^2_{jt})$ where X^1 consists of variables that vary over time for a given app, and the elements of X^2 do not. The first stage regression used is $p_{jt} = \gamma_0 X_{jt} + \gamma_2 X^1_{jt} + \epsilon_{jt}$.

Table 6: State Variables

State Variable	Evolution	Level
Paid (i.e., $1(p > 0)$)	Fixed	Firm-level
Retail Price	Fixed	Firm-level
IAP	Fixed	Firm-level
Ad	Fixed	Firm-level
Subscription	Fixed	Firm-level
Download Size	AR-1, when updated	Firm-level
Last-period Update Status	Firm's action	Firm-level
Competitors' Last-period Update Statuses	Competitors' actions	Firm-level
ξ_j	Eq. (3)	Firm-level
ξ_{-j}	Eq. (3)	Firm-level
Number of Firms	Fixed	Market-level
Market Average Download Size	(See note)	Market-level

The average download size for the market is computed as the average of the download sizes for all apps in the market. Those download sizes follow an AR-1 process, changing only when a given app has been updated.

Following BBL, I rewrite Eq. (8) to express the value function as a linear function of the parameters to be estimated.

$$\begin{aligned} V(S_{kjt}|\sigma_j,\sigma_{-j}) =& E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t R_j(S_{kjt},\theta^D) \\ &-\theta^{Minor} E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t 1(a_{jt} = \texttt{Minor}) \\ &-\theta^{Major} E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t 1(a_{jt} = \texttt{Major}) \\ &+\theta^{\epsilon} E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t \epsilon_{jt}(a_{jt}) \\ =& \mathbf{R}_{j,\sigma_j,\sigma_{-j}} - \theta^{Minor} \mathbf{F}^{Minor}_{j,\sigma_j,\sigma_{-j}} - \theta^{Major} \mathbf{F}^{Major}_{j,\sigma_j,\sigma_{-j}} + \theta^{\epsilon} \boldsymbol{\epsilon}_{j,\sigma_j,\sigma_{-j}} \end{aligned} \tag{22}$$

As discussed in Section 3, the intensive margin of demand cannot be reliably estimated due to limited data availability. Thus, it is not possible to develop a structural model for app j's revenue, $R_j(S_{k,j};\theta^D)$, since many apps will depend on intensive- as well as extensive-margin demand for revenue. Instead, I estimate a flexible model of revenue on the current state and a large number of additional covariates that are each a function of the current state.[26] Using the results of this

[26] Revenue data for the firms is calculated using the Top Grossing ranking list. See Section 4.1 for more details.

model I am able to predict the revenue an app will earn given the state, and this prediction is used in the supply estimation. Thus, revenue is treated as observed in the estimation procedure. Letting $\tilde{R}(S_{k,j}, \theta^R)$ represent the estimated reduced form revenue model, the value function used in estimating this model is

$$V(S_{kjt}|\sigma_j, \sigma_{-j}) = \tilde{\mathbf{R}}_{j,\sigma_j,\sigma_{-j}} - \theta^{Minor}\mathbf{F}^{Minor}_{j,\sigma_j,\sigma_{-j}} - \theta^{Major}\mathbf{F}^{Major}_{j,\sigma_j,\sigma_{-j}} + \theta^{\epsilon}\boldsymbol{\epsilon}_{j,\sigma_j,\sigma_{-j}} \tag{23}$$

where $\tilde{\mathbf{R}}_{j,\sigma_j,\sigma_{-j}}$ is equivalent to calculating $\mathbf{R}_{j,\sigma_j,\sigma_{-j}}$ using the reduced form revenue model in place of a full structural model of revenue.

As outlined in BBL, the linearity of the firms' value functions in $\theta = (\theta^{BF}, \theta^F, \theta^{\epsilon})$ significantly reduces the computational burden of estimating this model, because it is only necessary to calculate the value functions once, as opposed to re-calculating the values for each candidate parameter during the optimization process. I calculate approximations of $\boldsymbol{R}$, $\boldsymbol{F}^{Minor}$, $\boldsymbol{F}^{Major}$, and $\boldsymbol{\epsilon}$ using forward simulation. Out of concern for simulation error I set the approximated values of these terms equal to their average value across a large number of simulations.

Given the estimated policy functions and the reduced form revenue model. I use the following process to calculate a single simulated path for $\boldsymbol{R}$, $\boldsymbol{F}^{Minor}$, $\boldsymbol{F}^{Major}$, and $\boldsymbol{\epsilon}$. In these simulations I assume the discount factor $\delta = 0.95$.

1. Initialize $\boldsymbol{R}$, $\boldsymbol{F}^{Minor}$, $\boldsymbol{F}^{Major}$, and $\boldsymbol{\epsilon}$ equal to 0.

2. Given the state, calculate the conditional choice probabilities (CCPs) for all firms in the market, and for each firm select an action based on that firm's set of CCPs. $\boldsymbol{F}^{Minor}$ or $\boldsymbol{F}^{Major}$ are updated as appropriate (dependent on the chosen action for firm j), as well as $\boldsymbol{\epsilon}$.

3. Update the state variables to reflect changes due to the firms' actions.

4. Given the updated state, calculate the firms' revenues using the revenue model $\tilde{R}$ and update the value of $\boldsymbol{R}$.

5. Repeat steps 2-4 for 50 periods.

In order to estimate the model I simulate $\boldsymbol{R}_*$, $\boldsymbol{F}^{Minor}_*$, $\boldsymbol{F}^{Major}_*$, and $\boldsymbol{\epsilon}_*$ using the equilibrium CCPs estimated from observed data, as well as $|P|$ sets of $\boldsymbol{R}_p$, $\boldsymbol{F}^{Minor}_p$, $\boldsymbol{F}^{Major}_p$, and $\boldsymbol{\epsilon}_p$ under

alternative policies $p \in P$. In practice, I use nine alternative policies. For each term I average across 250 simulated paths for each initial state that is considered.

Stage 2 Given the results of Stage 1, in equilibrium the following should be true:

$$V(S_{kjt}|\sigma_j^*, \sigma_{-j}^*, \theta^{Supply}) \geq V(S_{kjt}|\sigma_j^p, \sigma_{-j}^*, \theta^{Supply}) \; \forall p \in P \tag{24}$$

Following this, I estimate $\theta^S = (\theta^{Minor}, \theta^{Major}, \theta^{\epsilon})$ using the BBL objective function,

$$\theta^S = \arg\min_{\theta} \sum_{p \in P} \left\{ V_{j,\sigma_j^p,\sigma_{-j}^*} - V_{j,\sigma_j^*,\sigma_{-j}^*})\theta, 0 \right\}^2 \tag{25}$$

6 Results from the Structural Model

6.1 Demand Results

Table 7 shows the results of estimating the demand model (Eq. (21)). The model is estimated using non-linear least squares (NLLS) under three specifications: specification (1) includes the price of the app, specification (2) uses an indicator variable for whether the app charges a non-zero price or not, and specification (3) includes both the price and the "paid" indicator. For the reasons discussed below, (3) is the preferred specification. The first set of results groups all updates together, and the second set of results classifies the updates as either Major or Minor updates according to the Support Vector Machine (SVM) classification method discussed in Section 4.2.[27]

Updates I find that updates increase the demand for apps, and that there is a non-zero correlation in the quality of apps over time, with estimates of ρ close to 0.86. While updates don't have a large direct effect on quality, their effect persists, via the ξ_{jt} AR-1 process, across multiple periods. These results are consistent with the finding in Foerderer and Heinzl (2017), Ghose and Han (2014), and Comino, Manenti, and Mariuzzo (2016) who all find evidence that updates increase demand.

As shown in the right half of Table 7, consumers respond to both Major and Minor updates, but

[27] See Section B.3 for results from using apps' version numbers as an alternative means means for classifying updates. The results are qualitatively similar.

Table 7: NLLS Demand Estimates

	Update			Update Types		
	(1)	(2)	(3)	(1)	(2)	(3)
Update	0.0927***	0.0867***	0.0897***			
	(0.0264)	(0.0264)	(0.0263)			
Major				0.0706*	0.0648	0.0682*
				(0.0402)	(0.0402)	(0.0402)
Minor				0.1066***	0.1005***	0.1033***
				(0.0326)	(0.0326)	(0.0326)
Price	-0.0349***		-0.0310***	-0.0349***		-0.0310***
	(0.0048)		(0.0049)	(0.0048)		(0.0049)
Paid		-0.9162***	-0.6854***		-0.9156***	-0.6849***
		(0.1723)	(0.1730)		(0.1723)	(0.1730)
IAP	0.4176***	0.0322	-0.0184	0.4163***	0.0312	-0.0194
	(0.1334)	(0.1738)	(0.1705)	(0.1334)	(0.1737)	(0.1704)
Ad	0.2910	0.1422	0.1454	0.2809	0.1321	0.1357
	(0.3660)	(0.3714)	(0.3647)	(0.3664)	(0.3717)	(0.3650)
Subscription	-0.3049**	-0.3360***	-0.3665***	-0.3054**	-0.3365***	-0.3670***
	(0.1217)	(0.1229)	(0.1217)	(0.1217)	(0.1229)	(0.1217)
$\ln(size)$	0.1936***	0.1769***	0.1952***	0.1954***	0.1787***	0.1970***
	(0.0486)	(0.0487)	(0.0479)	(0.0487)	(0.0487)	(0.0480)
ρ	0.8609***	0.8623***	0.8598***	0.8608***	0.8623***	0.8598***
	(0.0033)	(0.0033)	(0.0033)	(0.0033)	(0.0033)	(0.0033)

Observations are app-weeks. Standard errors are in parentheses.
All specifications include controls for apps' age appropriateness ratings, iOS platform version, and month.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

they do not appear to react differently to the different types of updates.[28] The fact that consumers do not respond differently between Major and Minor updates shows that, in general, the content of an update does not matter to potential new consumers. In fact, this suggests that in general updates may simply serve as a signal of continued development or as a purely promotional tool. That is, rather than considering how an update has improved the quality of an app, consumers may simply take note that an app has been updated recently when making a purchase decision.

Monetization As expected, I estimate a negative relationship between price and demand. Further, comparing specifications (1) and (2), I find that consumers are far more sensitive to an increase in price from free to \$0.99 (the minimum non-zero price allowed by the platform), than to an equivalent increase from any non-zero price. Specification (3) captures both the consumer response to the price of a product, and the response to whether the product is free or not. Noticeably, the coefficient on the paid indicator in specification (3) is much larger in magnitude than the coefficient on price. This suggests a non-linear effect of price on utility. In particular, consumers appear to be highly reticent to pay for an app, but conditional on paying, they are not particularly price sensitive. Using the results from specification (3), the additional disutility of increasing the price from \$0.00 to \$0.99 is approximately the same as increasing the price from \$0.99 to \$15.99. Specification (3) is the preferred specification from these results, as it is able to capture this non-linearity in price sensitivity.

Table 8 shows the results when the model is estimated using non-linear two-stage least squares (NLTSLS) with $\Lambda_{jt} = 0$.[29] The price coefficient in specification (1) is larger in magnitude compared to the NLLS results, but the disparity between the NLLS and NLTSLS results is smaller in specification (3), which suggests that controlling for whether or not an app is a paid app is the primary cause for bias in specification (1).

Finally, I find that the inclusion of subscriptions in an app is associated with lower utility. IAPs and ads, on the other hand, do not have a statistically significant effect on utility, which may be due to the relatively limited information consumers receive about the inclusion of and content associated with IAPs and the fact that developers may attempt to hide the presence of advertisements in their

[28] Using a Wald test, I cannot reject the hypothesis that Major and Minor updates have the same effect at the 5% level in Specification (3).

[29] Results without restricting Λ_{jt} to be 0 are forthcoming.

Table 8: NLTSLS Demand Estimates
$\Lambda_{jt} = 0$

	Update		Update Types	
	(1)	(3)	(1)	(3)
Update	0.0940***	0.0917***		
	(0.0264)	(0.0264)		
Major			0.0734*	0.0718*
			(0.0402)	(0.0402)
Minor			0.1069***	0.1043***
			(0.0326)	(0.0326)
Price	-0.0553***	-0.0463***	-0.0552***	-0.0462***
	(0.0080)	(0.0084)	(0.0080)	(0.0084)
Paid		-0.6177***		-0.6169***
		(0.1776)		(0.1776)
IAP	0.2871**	-0.0755	0.2862**	-0.0759
	(0.1420)	(0.1755)	(0.1420)	(0.1755)
Ad	0.2732	0.1485	0.2638	0.1396
	(0.3746)	(0.3740)	(0.3748)	(0.3742)
Subscription	-0.3450***	-0.3948***	-0.3452***	-0.3950***
	(0.1241)	(0.1245)	(0.1241)	(0.1245)
$\ln(size)$	0.2074***	0.2060***	0.2091***	0.2076***
	(0.0495)	(0.0492)	(0.0496)	(0.0493)
ρ	0.8603***	0.8595***	0.8603***	0.8595***
	(0.0033)	(0.0033)	(0.0033)	(0.0033)

Observations are app-weeks. Standard errors are in parentheses.
All specifications include controls for apps' age appropriateness ratings, iOS platform version, and month.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Table 9: Dynamic Parameter Estimates

θ^{Minor}	3,207.10
	(197.27)
θ^{Major}	3,900.05
	(173.98)
θ^{ϵ}	423.95
	(169.36)

Bootstrapped standard errors are reported in parenthesis.

App Store listings.

Interpreting the monetization estimates as causal effects requires abstracting away from developers' selection into a particular method (or methods) of monetization – which is unlikely to be a reasonable abstraction. Additionally, the potential bias caused by this selection effect is difficult to sign. For example, it could be that apps that use ads are on average lower quality than apps that use IAPs because viewing an ad does not require the consumer to take an explicit action (purchasing the IAP). On the other hand, it could be that ad-financed apps are on average higher quality because consumers are more likely to spend extended periods of time using higher-quality products and therefore monetizing the time spent viewing is optimal relative to trying to sell product add-ons to the consumer. The topic of how developers choose how to monetize their apps is not currently well understood, and is left to future work. Since apps rarely change the form(s) of monetization they use, understanding this selection effect would require modeling apps' entry decisions into the available forms of monetization.

6.2 Fixed Cost Results

Table 9 shows the results of estimating the dynamic app updating model. In line with the fact that updating is relatively uncommon, I find that app updates are quite costly. The cost of a Minor update, θ^{BF}, is \$3,207.10, and the cost of a Major updates is 21.6% higher at \$3,900.05. While these costs may at first seem negligible, and the difference between them small, it's important to view them in context. Across the sample, the average weekly gross revenue is \$1,445, with the median slightly lower at \$1,175.00. Thus, a Minor update costs approximately the same as two weeks of revenue, and a Major update costs an additional half-week's revenue.

Of course, as shown in Section 6.1, updates increase demand, so developers likely earn their revenue back in less time than average revenue figures would suggest. In fact, average revenue is approximately $600 higher on average immediately following an update.

6.3 Understanding Intensive Margin Demand

Taken together, the demand and supply results appear to present a contradiction: In Section 6.2, I show that Major updates are more expensive to produce than Minor updates, yet Table 7 shows that demand is not differentially affected by Major updates. Despite these two facts, as evidenced by Fig. 4 and Table 4, developers produce a relatively large number of Major updates. The fact that developers do regularly produce Major updates suggests that the different types of updates may indeed serve different strategic purposes, but that those purposes are not focused on spurring new *extensive margin* demand. Instead, Major updates may have an outsized effect on consumer behavior through the intensive margin of demand. That is, Major updates may increase the likelihood that a consumer uses an already-owned app in a given period by more than a Minor update, which would bring a greater return to the developer via use-based revenue sources.

This possibility is consistent with the idea that consumers' failure to differentiate between different types of updates may result from a ex-ante lack of information about the full functionality and user experience of apps. Under this view, so much of the quality of an app is unknown that the specifics of how it has been updated are uninformative about the degree to which overall quality has changed. Of course, other explanations — e.g., rational inattention to the details of updates — are possible. Whether and how consumers respond to Major versus Minor updates on the intensive margin remains an open research question — though, one that is currently stymied by significant data collection challenges.

7 Calculating Firm Behavior without Digitization through Counterfactual Analysis

Having constructed and estimated a model of consumers' demand for apps and developers' app updating decisions, I now return to my primary research question: How the digitization of consumer goods — characterized by the addition of two new strategic tools — affects firms' updating behavior.

In particular, I investigate how both the frequency and content of updates changes in the presence of these tools. Since these tools have been available to developers since the introduction of the App Store, I am unable to take advantage of some form of a natural experiment in order to answer this question. Instead, I conduct counterfactual simulations wherein I "turn off" these two aspects of digitization and simulate how developers would choose to update in their absence. The results of this counterfactual exercise serve as a proxy for how developers would behave in a "traditional" durable goods industry. Thus, I compare the results of this counterfactual exercise to observed behavior in order to estimate an effect of digitization on firms' updating behavior.

Model Restrictions In practice, turning off the two aspects of digitization I study means making two specific restrictions in the model. First, I artificially restrict the revenue developers can earn to revenue earned via the extensive, or purchase, margin of demand. Revenue from advertisements, subscriptions, and in-app purchases is all fixed at zero in the counterfactual regime. Put another way, firms no longer consider the effect of updates on consumers' intensive margin behavior. Second, I remove the ability for consumers to receive product updates to apps they have already purchased.

In terms of the model developed in Section 3, these two restrictions are represented by first changing the firm's revenue function, R_j from

$$R_j(S_{kjt}) = \overbrace{p_j^{Retail} s_{j,t}^{ext} M_{mt}}^{\text{Direct Sales Revenue}} + (\texttt{Intensive Margin Revenue})$$

to

$$R_j^{counter}(S_{kjt}) = \overbrace{p_j^{Retail} s_{j,t}^{ext} M_{mt}}^{\text{Direct Sales Revenue}}, \tag{26}$$

and by changing consumers lifetime expected utility from purchasing an app from

$$\Lambda_{ijt} = E\left[\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \ln\left(1 + e^{X_{j\tau}\beta + \xi_{j\tau}}\right)\right],$$

where consumers have expectations over future updates and future consumer-app match values, to

$$\Lambda_{ijt}^{counter} = \sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \ln\left(1 + e^{X_{j\tau}\beta + \xi_{j\tau}}\right), \tag{27}$$

where the consumers' expectations are only over future consumer-app match values, and not over

future updates, and $\xi_{j\tau}$ is not affected by the developer's updating decisions in periods $\tau' > \tau$.

Solving the Model and Forward Simulation For computational reasons, I assume firms make static product updating decisions. That is, in the counterfactual simulations, firms do not consider how a decision in period t will affect the state of the market in period $t+1$. Because of this, I expect that this counterfactual will *understate* the level of updating in the counterfactual regime, and thus will result in my potentially *overstating* the effect of digitization on the frequency of updating.

Accounting for Counterfactual Prices It is necessary to account for the fact that prices are likely to differ between the observed and counterfactual regimes. As discussed in Section 3.3, my model of app updating focuses on developers' product updating decisions, and assumes that prices are fixed. In order to account for the fact that prices will differ in the counterfactual regime, I assume firms play a two-stage game, where in the first stage they set prices, and in the second stage they engage in the infinite-horizon updating game defined in Section 3.3 and estimated in Section 5.2. In the interest of simplicity I assume firms play a complete information game in the first stage, where all firms know the value of all other firms under each possible combination of prices. In practice, I solve this two-stage game backwards, first by calculating each firms' value under each possible pricing equilibrium, and then by determining the Nash equilibrium in the first-stage pricing game. In order to make this exercise computationally feasible I restrict firms to selecting a price from the set $\{0.99, 9.99, 19.99, 29.99, 39.99\}$.

Counterfactual Results Table 10 presents the results of the six counterfactual simulations. For the static version of the counterfactual, I simulate updating in six markets, each with between two and four apps. Table 10 presents the unconditional likelihood of an app in the given market updating in a given period for both the counterfactual regime and the observed, digital regime. The "Counterfactual" columns present the frequency of updates when products are treated as traditional, non-digital consumer goods, and the "Digital" column presents the frequency of app updates under digitization. For each type of updating, Overall, Minor updates, and Major updates, I present the difference between the level of updating for digital products and traditional products, Δ. Positive values of Δ indicate an increased level of updating in the Digital regime compared to the Counterfactual regime.

Table 10: Counterfactual Results (Static Updating)

	Overall Updates			Minor Updates			Major Updates			Major Conditional on Updating		
Market	Counterfactual	Digital	Δ	Counterfactual	Digital	Δ	Counterfactual	Digital	Δ	Counterfactual	Digital	Δ
16	7.8%	5.9%	-1.9pp	5.5%	3.6%	-1.9pp	2.3%	2.3%	0.0pp	0.29	0.39	0.10
24	9.6%	17.6%	8.0pp	5.8%	9.8%	4.0pp	3.8%	7.8%	4.0pp	0.39	0.44	0.05
25	8.3%	16.1%	7.8pp	4.0%	7.2%	3.2pp	4.3%	8.9%	4.6pp	0.52	0.55	0.03
29	6.7%	16.2%	9.5pp	4.1%	10.8%	6.7pp	2.6%	5.4%	2.8pp	0.39	0.33	-0.05
31	3.7%	8.5%	4.7pp	1.5%	2.8%	1.3pp	2.2%	5.6%	3.4pp	0.60	0.67	0.07
36	9.3%	15.1%	5.9pp	6.5%	9.7%	3.2pp	2.8%	5.4%	2.6pp	0.30	0.36	0.06

I find that overall updating is higher under digitization in all but one market. These increases cover a range of 4.7 to 9.5 percentage points, representing an increase in the rate of updating of between 64% and 142% relative to a baseline of the "non-digital" counterfactual level. These increases are driven by an increase in both Major and Minor updates, though the rate of Major updates increases by slightly more than Minor updates. Indeed, in five of the six markets, the chance of a given update being a Major update increases by between 6% and 31%. That said, as with the increase in the frequency of updating, I find that in one of the simulated markets the relative frequency of Major updates falls.

These results speak to the importance of accounting for intensive-margin behavior when studying both digital and digitizing industries. The fact that digitization is found to result in a 64% to 142% increase in the rate of product innovation shows that any analysis of firm behavior in this industry must account for the intensive-margin incentives firms face. This presents both new opportunities — a more nuanced understanding of non-price competition — and new difficulties — data on relevant intensive-margin prices and quantities can be difficult to collect — for economic analysis.

8 Conclusion

As consumer goods undergo a process of digitization, firms gain access to new strategic tools, include the abilities to monetize the use of their products and to continue to update their product after a consumer has purchased it. I find that the availability of these two tools leads to more frequent product updating in Apple's App Store marketplace, and, in general, a higher relative frequency in Major, feature-adding updates compared to more minor, incremental updates. There is, however, a large level of variation in these results across markets, which suggests that the specifics of an

individual market play an important role in determining how digitization affects firms' incentives to provide more substantial updates, and is an area for future research.

In answering the question of how digitization affects firms' updating behavior, I have contributed a new method for defining markets, which allows economists to use high-dimensional product descriptions (e.g., text, images) in order to group products into markets. In addition, this work has contributed to a growing body of research that uses text as a primary form of data.

As an increasing number of consumer durable good industries begin to undergo digitization it is important to understand how the changes brought on by digitization affect firm behavior. While the specific effects estimated in this paper are specific to the smartphone application industry, the qualitative results are relevant to other industries. In particular, we should expect to see more frequent product updating via digital updates in other industries, an expectation that is already coming to fruition in some markets, such as in the market for electronic cars.

This paper also identifies a possible concern as products become increasingly digital. My finding that consumers do not differentiate between Major and Minor updates when making their extensive-margin (purchase) decisions suggests that the saliency of product functionality and, more generally, quality will be an important issue for firms.

As for future economic research on this and related industries, there remains substantial work to be done on understanding consumer behavior on the intensive margin of demand (i.e., product use behavior). Understanding how consumers behave *after* they have purchased a product will be increasingly important as more and more products earn a flow revenue from product use via advertisements, subscriptions, and within-product micro-transactions.

References

Anderson, Simon P., André de Palma, and Jacques-François Thisse. 1992. *Discrete Choice Theory of Product Differentiation*. The MIT Press. 58

Apple. 2016. "Apple - WWDC 2016 Keynote." URL https://www.youtube.com/watch?v=n5jXg{_}NNiCA. 3

———. 2017. "Developer earnings from the App Store top $70 billion." URL https://www.apple.com/newsroom/2017/06/developer-earnings-from-the-app-store-top-70-billion/. 3

Aryal, Gaurab, Federico Ciliberto, and Benjamin T. Leyden. 2018. "Public Communication and Tacit Collusion in the Airline Industry." 5

Bajari, Patrick, Lanier Benkard, and Jonathan Levin. 2007. "Estimating dynamic models of imperfect competition b." *Econometrica* 75 (5):1331–1370. 34

Bajari, Patrick, Jeremy T. Fox, and Stephen P. Ryan. 2008. "Evaluating wireless carrier consolidation using semiparametric demand estimation." *Quantitative Marketing and Economics* 6 (4):299–338. 18

Baker, Scott R., Nicholas Bloom, and Steven J. Davis. 2016. "Measuring Economic Policy Uncertainty." *The Quarterly Journal of Economics* 131 (4):1593–1636. 5

Berry, Steven T. 1994. "Estimating Discrete-Choice Models of Product Differentiation." *The RAND Journal of Economics* 25 (2):242. 33

Berry, Steven T., James Levinsohn, and Ariel Pakes. 1995. "Automobile Prices in Market Equilibrium." *Econometrica* 63 (4):841–890. 9

Bresnahan, T, J Orsini, and Pl Yin. 2014. "Platform Choice By Mobile Apps Developers." *NBER working paper* . 9

Bresnahan, Timothy, Xing Li, and Pai-ling Yin. 2016. "Paying Incumbents and Customers to Enter an Industry: Buying Downloads." *Working Paper* 94305. 9, 19

Brynjolfsson, Erik, Yu (Jeffrey) Hu, and Michael D. Smith. 2003. "Consumer Surplus in the Digital Economy: Estimating the Value of Increased Product Variety at Online Booksellers." *Management Science* 49 (11):1580–1596. 19

———. 2010. "The Longer Tail: The Changing Shape of Amazon's Sales Distribution Curve." *Social Science Research Network* (September):1–13. 21

Chevalier, Judith and Austan Goolsbee. 2003. "Measuring Prices and Price Competition Online: Amazon.com and BarnesandNoble.com." *Quantitative Marketing and Economics* 1 (2):203–222. 19

Comino, Stefano, Fabio Maria Manenti, and Franco Mariuzzo. 2016. "Updates Management in Mobile Applications : iTunes vs Google Play." (May):1–33. 10, 37

ComScore. 2016. "U.S. Smartphone Subscriber Market Share." URL https://www.comscore.com/Insights/Rankings/comScore-Reports-January-2016-US-Smartphone-Subscriber-Market-Share. 7

Davis, Jason P, Yulia Muzyrya, and Pai-Ling Yin. 2014. "Experimentation Strategies and Entrepreneurial Innovation: Inherited Market Differences in the iPhone Ecosystem." *INSEAD Working Papers Collection* (24):1–40. 9

Doraszelski, Ulrich, Gregory Lewis, and Ariel Pakes. 2016. "Just Starting Out: Learning and Equilibrium in a New Market." *NBER Working Paper* . 34

Doraszelski, Ulrich and Mark Satterthwaite. 2010. "Computable Markov-perfect industry dynamics." *RAND Journal of Economics* 41 (2):215–243. 15

Eizenberg, Alon. 2014. "Upstream innovation and product variety in the U.S. home PC market." *Review of Economic Studies* 81 (3):1003–1045. 5

Ericson, Richard and Ariel Pakes. 1995. "Markov-perfect industry dynamics: A framework for empirical work." *The Review of Economic Studies* 62 (1):53–82. 15

Ershov, Daniel. 2017. "The Effect of Consumer Search Costs on Entry and Quality in the Mobile App Market." 9, 19, 20, 21

Evans, David S. 2011. "The antitrust economics of free." *Competition Policy International* 7 (1):70–89. 25

Fan, Ying. 2013. "Ownership Consolidation and Product Characteristics: A Study of the US Daily Newspaper Market." *American Economic r* 103 (5):1598–1628. 5

Foerderer, Jens and Armin Heinzl. 2017. "Product Updates: Attracting New Consumers versus Alienating Existing Ones." *SSRN Electronic Journal* URL https://www.ssrn.com/abstract=2872205. 4, 37

Gandhi, Amit, Zhentong Lu, and Xiaoxia Shi. 2013. "Estimating Demand for Differentiated Products with Error in Market Shares." 56

Gans, Joshua S. 2012. "Mobile Application Pricing." :1–24. 9

Garg, R and Rahul Telang. 2012. "Estimating App Demand from Publicly Available Data." *Available at SSRN 1924044* 37 (4):1–22. 19, 20

Gentzkow, Matthew, Bryan T. Kelly, and Matt Taddy. 2017. "Text as Data." *NBER Working Paper Series* 23276:53. URL http://www.nber.org/papers/w23276. 5

Ghose, A.a and S.P.b Han. 2014. "Estimating demand for mobile applications in the new economy." *Management Science* 60 (6):1470–1488. 9, 19, 20, 21, 27, 37

Goettler, Ronald L and Brett R Gordon. 2011. "Does AMD Spur Intel to Innovate More?" *Journal of Political Economy* 119 (6):1141–1200. 5

Goode, Lauren. 2016. "Apple's new subscription offerings are now available to App Store developers." URL https://www.theverge.com/2016/9/2/12774758/apple-developers-app-store-new-subscription-rules. 8

Gowrisankaran, Gautam and Marc Rysman. 2007. "Dynamics of Consumer Demand for New Durable Goods." *Journal of Political Economy* 120 (6):1173–1219. 4

Hoctor, Kevin. 2013. "In-App Purchase - The Future is Here." URL http://blog.hoctor.com/in-app-purchase-the-future-is-here/. 58

Kesler, Reinhold, Michael Kummer, and Patrick Schulte. 2017. "Access to User Data , Market Power and Innovation in Online Markets: Evidence from the Mobile App Industry." 14

Lambert, Fred. 2017. "Tesla pushes new Autopilot 2.0 update with truly "silky smooth" control algorithm." 2

Lee, Robin S. 2013. "Vertical Integration and Exclusivity in Two-Sided Markets." *The American Economic Review* 103 (7):2960–3000. 4

Liu, Yongdong. 2017. "Mobile App Platform Choice." :1–66. 9, 19

Matthews, Lee. 2015. "Tesla P85D software update reduces the car's already insane 0-60 time." URL https://www.geek.com/apps/tesla-p85d-software-update-reduces-the-cars-already-insane-0-60-time-1614741/. 2

Nekipelov, Denis, Minjung Park, and Yongdong Liu. 2013. "Timely versus quality innovation: The case of Mobile Applications on iTunes and Google Play." 9

Nelson, Phillip. 1970. "Information and Consumer Behavior." *Journal of Political Economy* 78 (2):311–329. 58

Newman, John M. 2015. "Antitrust in Zero-Price Markets: Foundations." *University of Pennsylvania Law Review* 164 (1). URL http://heinonline.org/HOL/Page?handle=hein.journals/pnlr164{&}id=153{&}div={&}collection=. 26

Perry, Charles. 2015. "The Shape of the App Store." URL dazeend.org/2015/01/the-shape-of-the-app-store/28/. 19

Quan, Thomas W. and Kevin R. Williams. 2015. "Product variety, across market demand heterogeneity and the value of online retail." *Working Paper* November. 56

Rousseeuw, Peter J. 1987. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis." *Journal of Computational and Applied Mathematics* 20 (C):53–65. 28

Spencer, Graham. 2015. "A Beginner's Guide to App Store Pricing Tiers." URL https://www.macstories.net/stories/a-beginners-guide-to-app-store-pricing-tiers/. 8

Sweeting, Andrew. 2013. "Dynamic Product Positioning in Differentiated Product Markets: The Effect of Fees for Musical Performance Rights on the Commercial Radio Industry." *Econometrica* 81 (5):1763–1803. 5, 15

Wu, Alice H. 2017. "Gender Stereotyping in Academia : Evidence from Economics Job Market Rumors Forum." (August). 5

Yin, Pai Ling, Jason P. Davis, and Yulia Muzyrya. 2014. "Entrepreneurial innovation: Killer apps in the iPhone Ecosystem." *American Economic Review* 104 (5):255–259. 9

Appendices

A Additional App Store Background

A.1 Consumers and the App Store

Hardware To access the iOS platform and its associated store, a consumer must first purchase either an iPhone, an iPad, an iPod Touch, or an Apple TV (Apple's mobile phone, tablet, touch-screen MP3 player, and television-based digital media player product lines, respectively).[30] These devices can be purchased either directly from Apple, through an electronics retailer (e.g., Best Buy or Amazon), or through a cellular service provider (e.g., AT&T or Verizon). iPhones must be purchased with a cellular service plan, while iPads come in two versions, one with and one without cellular service. The iPod Touch and Apple TV do not have cellular service. Unlike most other platforms, Apple exclusively designs and manufactures the devices compatible with its platform.

App Categories Given ownership of an iOS-compatible device, a consumer is able to directly access the App Store, Apple's mobile app marketplace. Within the store, apps are organized into 22 categories. Examples of categories include Games, Entertainment, and Productivity. Fig. 2 shows a snapshot of the number of apps in each category.

Consumers register a credit card with the App Store, thus facilitating quick and easy purchases of apps. Having purchased an app, the consumer has access to that app for life, even if the app's developer chooses to remove the app from the App Store. However, without occasional product updates from the developer, a given app will likely not work with future versions of the platform. The iOS platform is typically given a major update once a year, with a variable, but generally small number of additional, incremental updates throughout the year.

Cross-device Use Apps in the App Store are clearly marked as either being for the iPhone, for the iPad, or they are marked as "Universal" apps, which run on either type of device.[31,32]

[30] The App Store also includes apps for the Apple Watch product line, but the watch requires a connection to an iPhone or iPad in order to work.

[31] Developers are relatively limited in their ability to restrict their apps to particular devices within a given product line.

[32] Apps that run on iPhone are able to run on iPod Touch devices.

For a number of years following the introduction of the iPad, consumers would have to purchase separate iPhone and iPad versions of a particular app (if available) if they wished to run the apps on both devices. This has become less common in recent years as Apple has changed the underlying structure of the iOS platform to better accommodate Universal apps.

A.2 App Development

To develop for the iOS platform, developers must join Apple's developer program, which costs $99/year per member. Developers create iOS apps primarily using some combination of the Objective-C and Swift programing languages. Any app characteristics the developer wishes to include in the product must be implemented via code. Developers are aided by a collection of public application programming interfaces (APIs), which are pre-written methods for interacting with the primary features of the platform and corresponding devices.[33] In addition, developers can have their app interface with other apps and web services, though limits exist on what types of interactions are permissible. These interactions are conducted through the use of that app's or service's API.

Given the fact that apps are fully digital products, and because Apple handles the requisite data center for hosting apps and propagating them to consumers' devices after purchase, the marginal cost of an additional sale is nearly zero, and in many cases, is in fact zero. Thus, app production is primarily characterized by the fixed costs of writing the app's code, and then additional fixed costs associated with writing the code for any future updates to the product. Indeed, as the capital requirements for producing apps are very low (all that is needed is a computer), entry costs are very low – which in part explains the large number of apps within the store.

B Data Appendix

B.1 Companion versus Product Apps

Apps can be classified as either companion or product apps. Companion apps are those that serve as a companion to an existing product or service. For example, the American Airways app, which

[33] As an example, Apple provides a set of APIs for accessing the basic camera functionality. Thus, a developer does not have to write the code that make the device take a photo. Instead, a developer need only to "call" the relevant API and the phone will take and return a photo to the developer's application.

offers detailed flight information, flight check-in, and a consumer's boarding pass on the day of the flight, serves as a companion to American Airways flights. On its own, the app provides no value. Similarly, the brick-and-mortar retailer Target's app serves as a companion to shopping at Target either in the store, where it provides information about what aisle certain products are available in, or online.

Product apps are those that are themselves the primary product being offered. The weather app, Dark Sky, which provides forecasts and weather radar, is produced as a standalone product. This distinction is important, because while product apps can reasonably be expected to be developed in a standard profit-maximizing way, there is no such expectation for a companion app as it is just a (possibly small) piece of a much larger product offering.[34] Further, from a demand perspective, consumer preferences for a companion app will be highly dependent on products, pricing, and competition outside of the app industry – demand for companion apps is largely a derived demand. For example, demand for the American Airways app will almost entirely be driven by demand for American Airways flights.

B.2 Accounting for Abandoned Apps

A second issue in defining a sample in the app store stems from the fact that, as discussed in Section 2, entry into the app store is relatively inexpensive. Thus, many apps have such a low level of sales (never or rarely achieving a sales ranking associated with non-zero sales) that it is hard to consider them as behaving strategically in the marketplace. It is important to re-emphasize the fact that, since app development primarily consists of fixed costs, developers face little to no costs in keeping an abandoned product in the store. Unlike the production of physical goods, continuing to sell an abandoned app does not, in general, result in continued variable expenses.

Considering, for a moment, only the apps in the Productivity category, only 10.6% ever reach a sales ranking better than 500^{th} in the sample period, and only 3.2% ever reach a ranking better than 200. Fig. 10 shows the distribution of apps' best rankings for those that reach a ranking of at least 500 on at least one occasion. However, even among apps that do reach a particular ranking

[34] Accounting for the pricing and updating of a companion app would require modeling the larger product offering. For example, decisions made regarding Target's apps could be viewed as profit maximizing from the perspective of Target's retail operation, but directly observing the revenue collected through the app would likely not rationalize the costs of development.

Figure 10: Distribution of Productivity Apps' Best Ranking

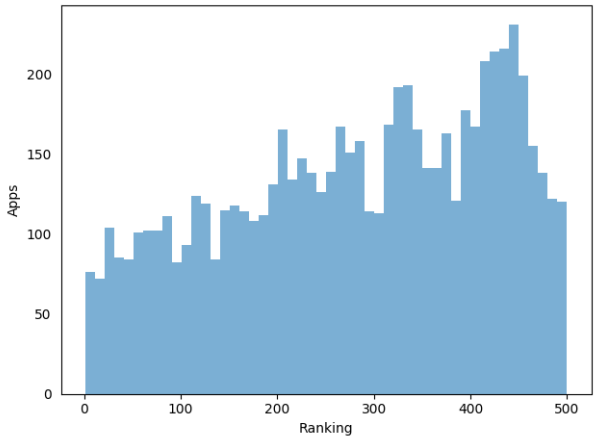
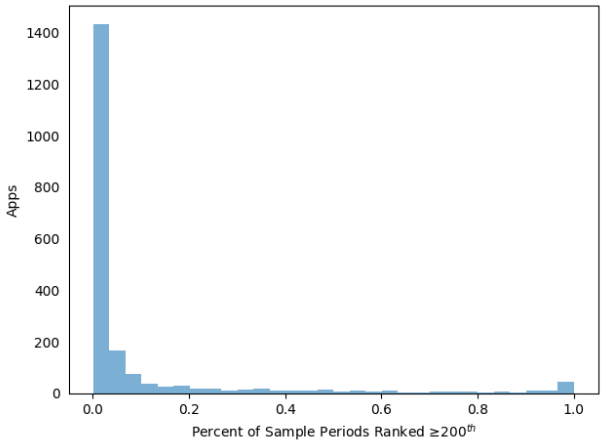
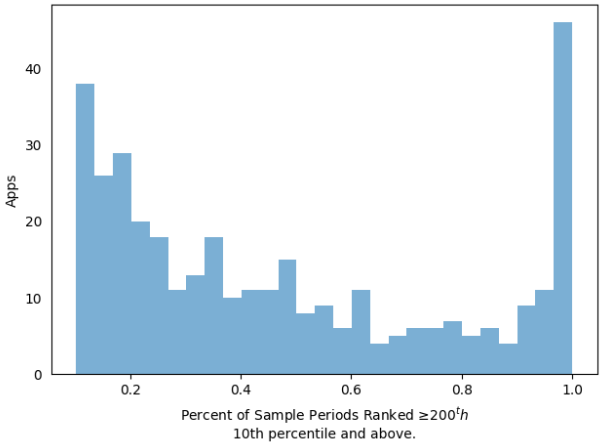


Figure 11: Persistence Above Ranking Threshold of 200



(a) Periods ranked above 200



(b) Periods ranked above 200 (10^{th} percentile and above.)

threshold, few apps are able to repeatedly maintain sales above that threshold. Fig. 11 shows that among apps that achieve at least a ranking of 200, nearly all spend less than 20% of the sample period at or exceeding that threshold.

There are several possible reasons for why so many apps enter the store, yet never achieve a meaningfully high ranking. One possibility is that there is a high degree of randomness in app store success, and so the nearly 90% of Productivity apps that entered the store but never reached a ranking of at least 500 had an ex-ante positive expected profit, but upon entry, "drew" a bad shock. Note that this disparity between expected and realized outcomes could be due to a high level of volatility in the marketplace, or to a general high degree of uncertainty regarding consumer

demand on the part of the developer. Another possibility, which explains *some* of the consistently low-ranking apps in the store, though certainly not all, is that many apps are hobbies or passion projects for developers. That is, some apps may only exist to scratch a particular developer's own itch.

This issue would be especially problematic when estimating the structural model in Section 3, because keeping abandoned apps in the sample would introduce an enormous number of zeroes in the market-share data, which is inconsistent with the logit demand model developed in Section 3.2. While methods do exist for allowing this, they generally require an assumption of sampling error in shares in order to explain zero-shares (see Gandhi, Lu, and Shi (2013)). This is inappropriate in the current context, as many products on the App Store actually do have zero sales throughout much of, and in many cases, all of, the sample period.[35]

B.3 Classifying Updates by Version Number

Every time a developer updates an application, they change the version number, which serves as a developer-defined index of the development of an app. Among software developers, it is customary to use the change in the version number to indicate the magnitude or importance of an update. For example, an update from version 4.3.1 to 5.0.0 typically indicates a major revision, whereas an update from version 4.3.1 to 4.3.2 typically indicates a smaller update. That said, there are no formal rules regarding version numbering, so updates where the version number changes from, for example, 4.3.1 to 5.0.0 may only include minor changes to the app despite the "large" change in the version number.

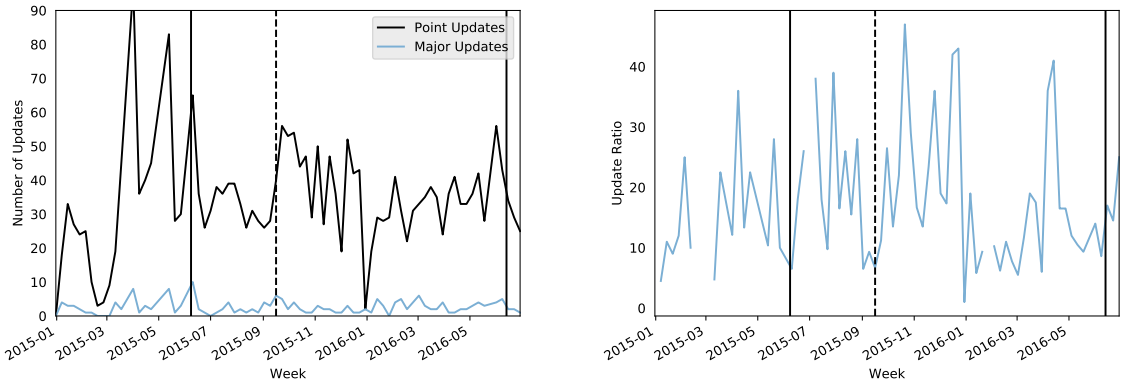
Borrowing vocabulary from the developer community, I code any update where the first number in the version number changes (e.g., 4.3.1 to 5.0.0) as a Major update, and any update where any of the subsequent numbers change (e.g., 4.3.1 to 4.3.2) as a Point update (so-called because the numbers after the first "point" are changed).

Estimating the demand model (with $\Lambda = 0$) using the version number update distinction instead of the SVM classification results in nearly identical results (tables forthcoming). As in my primary results presented in the paper, I cannot reject the hypothesis that Major and Bug Fix updates have

[35] Quan and Williams (2015) develop an alternative method for handling zero-shares, but it requires cross-market products. As there is no distinction between local app markets and a national app market, applying their method is not possible in this case.

Figure 12: Updates By Type

Version Number Classification



(a) App Updates by Type

(b) Ratio of Point to Major Updates

Table 11: Cross Tabulation of Update Classifications

	Major (VN)	Point (VN)	Total
Minor (SVM)	90	1,591	1,681
Major (SVM)	135	939	1,074
Total	225	2530	2,755

the same effect, though standard errors are smaller than in the SVM case.

C Intensive Margin of Demand

Developers face two margins of demand, the extensive and intensive margins. On the extensive margin consumers consider whether to purchase an app in a given market, on the intensive margin consumers choose whether or not to use their previously purchased apps, and whether or not to purchase subscriptions and in-app purchases.

In this appendix section I outline a model of intensive-margin demand for apps. This model is useful for both understanding the incentives firms face regarding the intensive margin, and the model is used in order to simulated consumers' lifetime expected utility from owning an app, which is used in the estimation of the demand model. See Sections 3.2 and 5.1 for more details on the extensive margin model and estimation procedure.

Each period consumers choose whether to use any of the apps they already own. That is,

consumer i, owning apps $j \in J_i$ makes $|J_i|$ independent decisions of whether to use each app or not. Specifically, consumers earn utility w_{ijt} from using app j at time t where

$$\begin{aligned} w_{ijt} &= X_{jt}\tilde{\beta} + \tilde{\xi}_{jt} + \tilde{\epsilon}_{ijt} \\ w_{i0t} &= \epsilon_{i0t} \end{aligned} \tag{28}$$

and the fraction of previous purchasers who choose to use app j in period t is s_{jt}^{int}, which is derived in the same way as s_{jt}^{ext}.

Eq. (28) differs from Eq. (1) in three ways. First, consumers do not consider the price of the app when making a use decision, as they have already paid that one-time cost. Second, $\Lambda_{jt} = 0$, reflecting the fact that making period-specific use decisions is independent across periods.[36]

Finally, a primary concern developers have about selling entirely digital goods is that consumers may not be well-informed about the quality of apps prior to purchase.[37] In light of this, I do not restrict the intensive-margin preference parameter $\tilde{\beta}$ and unobserved product quality $\tilde{\xi}$ to be equal to their extensive-margin counterparts. Furthermore, I assume that any difference in $(\tilde{\beta}, \tilde{\xi})$ is unknown to the consumer prior to making a purchase, which fits with the notion that apps may in some ways be experience goods à la Nelson (1970).

Derivation of Lifetime Expected Utility ($\Lambda_{jt}sp$) Consumers anticipate that apps will be updated in future periods, and thus consider the future utility of using the product when making their purchase decision. Specifically, consumers have expectations over future updates, and over future consumer-app match values ϵ_{ijt} (see Eq. (28)), which are relevant to the consumer's period-specific, intensive margin, use decision. Thus, conditional on the developer's updating choices, the expected value of owning previously purchased app j in period τ, when the consumer can either choose to use app j or not is the expected utility of Eq. (28). Since the error in the intensive margin demand model is assumed to be i.i.d. Extreme Value Type 1, this expectation, is defined by the standard "logsum" formula for the double exponential (Anderson, de Palma, and Thisse, 1992),

[36] It may be the case that in some app markets period t use affects the utility received in periods $\tau > t$. This is not considered here due in to data limitations. Games, the primary category where one might imagine this matters is not considered in this paper.

[37] E.g., "The limited information a prospective customer has prior to a purchase is one of the problems with Apple's App Store. People are supposed to fork over money for apps, but only get to see five screenshots and a few paragraphs of text before making a decision — that just doesn't cut it." (Hoctor, 2013)

which is

$$Ew_{ij\tau} = \ln\left(1 + e^{X_{j\tau}\beta + \xi_{j\tau}}\right) \tag{29}$$

Therefore, the expected future, discounted value of owning app j, Λ_{jt} is the discounted sum of Eq. (29) with expectations over the developer of app j's future updating decisions. Namely,

$$\begin{aligned} \Lambda_{ijt} &= E\left[\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} Ew_{ijt}\right] \\ &= E\left[\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \ln\left(1 + e^{X_{j\tau}\beta + \xi_{j\tau}}\right)\right] \end{aligned} \tag{30}$$

where δ is the discount factor. The symmetry of the model implies, $\Lambda_{ijt} = \Lambda_{jt}\ \forall i$.

Note that the formation of the expected future value Λ is entirely a function of consumers' extensive margin preferences β and the extensive-margin unobserved product quality ξ. While the values of β and ξ may differ between the two margins, consumers are only able to form expectations over what they know prior to purchasing a given app. Since the intensive-margin values, if they differ at all, are only learned *after* the point-of-sale, Λ cannot account for these differences.

Given w_{ijt} it is possible to define the number of "active users" of app j, AU_{jt}. Specifically,

$$AU_{jt} = s_{jt}^{ext} M_{m\tau} + s_{jt}^{int} \sum_{\tau=t_0}^{t-1} s_{j\tau}^{ext} M_{m\tau} \tag{31}$$

That is, AU_{jt} is the number of consumers who purchased the product in period t, plus the number of customers who have previously purchased the app and have chosen to use it in period t.

Given this measurement of an app's *intensive-margin market size*, one can imagine models similar to the intensive use model that characterize the share of an app's active users that purchase an IAP (s_{jt}^{IAP}) and that purchase, or make a continued payment for, a subscription ((s_{jt}^{sub}), as appropriate depending on the app's monetization strategy. In most cases it would be reasonable to assume all active users view any ads presented in the app, so $s_{jt}^{ads} = 1$. In the interest of brevity, and since such models do not enter into the estimation approach used in this paper, I omit a more precise discussion of these models.

Revenue R_j is defined as

$$R_j(S_{kjt}) = \overbrace{p_j^{Retail} s_{j,t}^{ext} M_{mt}}^{\text{Sales Revenue}} + \overbrace{\left(\sum_l (p_{j,l}^{IAP} s_{j,l,t}^{IAP}) + p_j^{Ads} + p_j^{sub} s_{j,t}^{sub} \right) AU_{j,t}}^{\text{Use Revenue}} \tag{32}$$

where, as noted above, s_{jt}^x is the share making a purchase of $x \in \{IAP, Sub\}$, as appropriate, and $AU_{j,t}$ is the set of active users for app j in period t. l indexes the number of IAPs offered by the app, which in many cases is greater than one. Note that marginal costs are assumed to be 0, as discussed in Section 2.

C.1 Calculating Λ_{jt}

I calculate Λ_{jt} using forward simulation.

Broadly, the process is similar to what's used in the supply model, in the estimation of the demand model, for each candidate parameter I do the following: For each observation, simulate forward 50 periods, calculating $\tilde{u}$ for each period. Then take the discounted (β) of each $\tilde{u}$, which gives a single simulation. I repeat this process 5 times, and, again for each observation, average across all of those simulations to get Λ_{jt}. In light of the potentially high computational cost of these simulations, I calculate and save each simulated path for each observation. Then for each candidate parameter in the estimation procedure I calculate $\tilde{u}$ for each period of each path, take the discounted sum for each path, and the average across all of the simulations.