There's an App (Update) for That
A Structural Model of Product Updating Under Digitization

Benjamin Thomas Leyden
Greensboro, North Carolina

B.A. Economics and Mathematics, University of North Carolina at Chapel Hill, 2012
M.A. Economics, University of Virginia, 2013

A Dissertation presented to the Graduate Faculty
of the University of Virginia in Candidacy for the Degree of
Doctor of Philosophy

Department of Economics

University of Virginia
May 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 between 63% to 142%, and, in particular, lead to an increase in the relative frequency of major, feature-adding updates compared to 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: D12, L13, L15, L86, O30
Keywords: Product innovation, Endogenous product characteristics, Dynamic oligopoly, Software, Digitization

Acknowledgements

This dissertation would not be possible without the support and guidance of many people. Thank you to my advisors, Simon Anderson, Federico Ciliberto, and Gaurab Aryal for your guidance, patience, and support. I am proud to call myself a student of these remarkable teachers.

Thank you to the many other members of the UVa Department of Economics who have supported me. Ken Elzinga and Lee Coppock, you are model teachers, and I hope to live up to the standards you have set. To Elliott Isaac, Zach Sullivan, Brett Lissenden, and Bill Johnson, thank you for your continued friendship and support — it has been a pleasure to go through this together. And to Zhou Zhang, thank you for your friendship and continual guidance — I could not ask for a better role model to follow through this program. Additionally, this research would not have been possible without financial support from the Bankard Fund for Political Economy, the Radulovacki Summer Research Fund, and the UVa Quantitative Collaborative.

I am indebted to the many app developers — particularly the participants at the Release Notes conferences — who have spoken with me about their industry, and, in some cases, provided data that has been essential to this project. I am grateful to Alice Zhao and Yono Mittlefehldt, who have been especially supportive in both welcoming me to the developer community, and in helping to make it possible for me to attend these conferences.

I was fortunate to develop strong, lifelong friendships while at the University of North Carolina, and I am grateful for the continued support of those friends. Thank you, Clint, Darvin, David, Dutra, Evan, Frank, H-dot, Jeremy, Michael, Patrick, Rob, Stilwell, and Zack.

I am grateful for the guidance and friendship of Anna Lea Johnson. You taught me to be diligent and hardworking, but more importantly, you taught me to be compassionate in everything I did. Mrs. J., thank you so much for all you did.

I have benefited greatly from the never-ending support and encouragement provided by my parents, Peggy and Dennis Leyden, and my sister, Sarah Leyden. I am grateful for everything you have done for me from the earliest days, as I dragged your unplugged phone room-to-room, to the present day.

Finally, I would like to thank my remarkable wife, Helen Chandler, who has been a constant source of support. Your encouragement and love has kept me going through even the most challenging of times, and I am certain I would not have been able to complete this dissertation without you.

This dissertation is dedicated to my father, Dennis P. Leyden, and my late grandfather, Dennis R. Leyden. I am proud to be following in your footsteps, and strive to study economics with the same thoughtfulness and dedication as you both have.

Contents

Chapter 1

Introduction

Consumer durable 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 behavior. 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 digital 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 due to digitization as defined above. To be clear, no sin-

gle characteristic of the modern app industry is completely new, rather, it is the confluence of these changes that distinguishes modern apps 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 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 a standard model of competition in durable goods, a firm competes first against other firms to make a sale, and then it competes *against itself*, as it updates its product 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 emphasis 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

[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 Chapter 3, developers keep 70% of the total revenue their app earns.

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 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 less substantial 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 purchase decisions. On the intensive margin, consumers make binary, period-by-period decisions of whether to use their already-purchased apps. 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.

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 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.

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] These data have 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 joins 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 natural language processing (NLP) and

[3]See Section 5.1.2 for a discussion of the reason for focusing on Productivity apps, Chapter 5 for a full discussion of how the 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 (2017) 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.

machine learning (ML) techniques to process 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 and content 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

[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.

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, point to the need for future work understanding consumer behavior on the intensive, or product use, margin of demand. In Section 7.3 I discuss the possible 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 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 good 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 Chapter 2, I discuss existing economic research that is relevant to this project. In Chapter 3, I provide an overview of Apple's App Store marketplace. Given that background, I develop a model of consumer and firm behavior for the app industry in Chapter 4. Chapter 5 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 5.5 provides preliminary evidence of the effect of digitization on product updating. Chapter 6 outlines the estimation procedure for both the demand and supply models, and Chapter 7 discusses the results of that estimation. Finally, in Chapter 8 I analyze how developers' updating behavior changes under digitization by considering counterfactual simulations that "turn off" the digital aspects of the industry. Chapter 9 concludes.

Chapter 2

Literature Review

The research presented here contributes to four strands of economic research: research regarding the effect of technological innovations on the economy, research regarding the demand for and supply of consumer durable goods, empirical research on how firms choose the characteristics of their products, and finally, research on the smartphone application industry.

2.1 Innovation and General Purpose Technologies

Economists have studied what drives innovative behavior dating back to work by Schumpeter (1947) and Arrow (1962). Much of the early work on this issue focused on the role of competition in affecting innovation. Indeed, starting with the early work of Schumpeter and Arrow, there has been a longstanding debate about whether increased concentration results in increased or decreased innovation. Schumpeter argued that increased concentration would increase innovation, as firms would be better able to reap the benefits of their innovative work, while Arrow argued that

the opposite was true. Namely, that firms in highly concentrated industries had little incentive to innovate given their already strong ability to exert market power.

This foundational work on what drives innovation continues to be debated to this day, with Aghion et al. (2005) finding modern empirical evidence for an argument first put forward by Scherer (1967) that the relationship between competition and innovation follows an inverted-U shape, with innovation being highly unlikely under both highly concentrated and highly competitive markets. Goettler and Gordon (2011) employ a structural model of product innovation in order to conduct counterfactual simulations to determine how changing the market structure of a single industry affects innovative behavior. In doing so they find evidence in support of the Schumpeterian view.

At the same time, particularly over the last twenty-five years, economists have begun to look beyond the competitiveness of an industry in order to understand what drives firms' innovative behavior. In particular, there has been an ongoing effort to understand how specific technological innovations induce follow-on innovation. The most prominent example of this work is Bresnahan and Trajtenberg (1995)'s notion of "General Purpose Technologies," which are seminal technological innovations that have large, aggregate effects on economic behavior (e.g., the steam engine, electricity, and information technology). Jovanovic and Rousseau (2005) provide a broad overview of the research into General Purpose Technologies, and a comparison of the roles electricity and information technology played in driving subsequent innovative activity.

The work presented in this paper is motivated by the work of Bresnahan and Trajtenberg, and the subsequent research on how technological changes affect future innovations. While the digitization of consumer goods is not a technological change on

the scale or scope of the introduction of the steam engine, it is a technological change that has the potential to affect many industries in an economically meaningful way. Today, an increasingly large swath of industries are undergoing product digitization, and it is important to understand how these changes affect the product innovation behavior of firms so that we can both anticipate what changes are to come, and properly evaluate firm conduct as policymakers work to regulate affected industries.

2.2 Durable Goods

A second, long-standing literature this research contributes to is the economic literature on durable goods. Early work in this area was largely theoretical, and focused on the pricing decision of firms in light of the durable nature — the consumer can repeatedly re-use the product — of such products. In studying the demand for durable goods, economists have highlighted the importance of capturing consumers' dynamic considerations. First, as highlighted by Coase (1972), consumers have the option of delaying a purchase. Coase's famous "conjecture," shows that in a monopoly setting, this forward-looking behavior by consumers will result in competitive pricing even in a monopoly market.

Gowrisankaran and Rysman (2007), present the now-canonical empirical model of dynamic consumer demand for durable goods. In their paper, consumers exhibit dynamic behavior in that they anticipate that the quality and price of available products may be better in future periods, thus delaying purchase. Further, once a purchase has been made, the consumer will take into account these dynamic considerations relative to the quality of the currently owned product.

The demand model developed in this paper differs from the approach developed

in Gowrisankaran and Rysman (2007). The digitization of consumer goods provides firms with the ability to continue to update already-sold products, which means that consumers no longer face the dynamic problem of optimally timing their purchase. Rather, consumers purchase products after considering the current quality of the product *and* what they expect the future quality of the product to be — a function of both expectations regarding future product innovations, and expectations about the stochastic depreciation of the product's quality.

More closely related to the industry studied in this paper, Lee (2013) and Zhou (2014) both study consumer demand for video game software. In both cases, software is considered to be a product of fixed quality. For example, in Lee (2013), period-to-period changes in the lifetime expected utility of a software purchase occur only as a result of changes in the composition of available products, temporal preference changes, and idiosyncratic match values. Consumers in his model have expectations over future values of the lifetime expected utility purchasing a video game, thus reducing the consumer's dynamic purchase problem to an optimal stopping problem. This paper differs from these papers, and other recent empirical papers on durable goods, as it allows for quality changes to an existing, already-purchased product. That is, unlike the software products in Lee (2013) and Zhou (2014), the inherent quality or characteristics of a product are not fixed. And, unlike in Gowrisankaran and Rysman (2007), firms are able to update products without requiring the consumer to purchase the newer version.

On the supply side, research regarding software development has primarily focused on the tradeoffs traditional software developers face when introducing new versions. Here, I distinguish between traditional software, such as the video games in the context of Lee (2013) and Zhou (2014), and modern "digitized" software markets where

firms have access to use monetization and post-purchase updating technologies. In the traditional case, firms must account both for consumers' dynamic purchase behavior — as in any other durable goods industry — and they must choose whether or not to make each version forward and/or backward compatible. That is, they must decide whether the new version of the product be capable of working with files produced with either prior or future versions of the product.[1]

Research on this issue has been primarily theoretical. Ellison and Fudenberg (2000) find that a monopoly developer may upgrade too frequently when providing backward-, but not forward-compatible software. Dogan et al. (2011) study how developers' upgrade choices, and their "upgrade design effort," or how much work firms will engage in in the first period to lower the cost of upgrading in the second period, respond to demand uncertainty. More recently, Brecko (2017) studies why software firms often do not price discriminate when releasing an updating version of their product.

2.3 Endogenous Product Characteristics

This work contributes to a growing empirical literature on endogenous product characteristics. Much of the empirical research in industrial organization over the last 30 years has taken product characteristics as exogenous in order to focus on pricing behavior. More recently, however, this assumption has been loosened.

Much of the recent empirical work on endogenous product characteristics has employed static models. This work is often focused on understanding how multi-product

[1] Any researcher who has had to request that a coauthor use the `saveold` command in Stata will be familiar with this issue.

firms choose what product assortments to offer. For example, Draganska, Mazzeo, and Seim (2009) study the product sets offered by ice cream manufactures, who routinely change their product offerings over the course of a year. Eizenberg (2014) studies the welfare effects of upstream innovation in the home computer industry by endogenizing the set of products computer manufactures choose to sell in light of the set of CPU technologies available to them.

Given the key role that assuming the exogeneity of product characteristics plays in estimating demand models (see Berry, Levinsohn, and Pakes (1995) and the multitude of papers that follow the approach developed therein), identification remains a significant challenge in estimating models with endogenous product characteristics. Papers in this literature often employ an ad hoc approach to identification, and a general approach to addressing the endogeneity issues that arise has not been developed.

For example, Eizenberg (2014) relies on a timing assumption — that firms choose which products to offer before observing demand shocks — in order to get point identification. This paper relies similarly on the timing of the model in order to identify the effect of updates on demand, though the timing in this case follows directly from institutional details of Apple's App Store marketplace. In particular, the propagation of app updates from a developer to consumers is delayed by a required review process, resulting in a delay between a developer's decision to update an app and the realization of the relevant demand shocks.

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), who studies radio stations' choices about when to change format. The model developed in Sweeting (2013)'s research on radio station format changes is closely related to the model developed in this paper. In both

cases, firms face a discrete menu of options — format choice in Sweeting's work, and how to update a software product in this paper. Additionally, in both cases, firms face somewhat atypical profit functions where marginal costs are zero, and revenue does not come from (in whole or in part) consumers' purchase decisions. In the case of Sweeting (2013), firms earn the entirety of their revenue through advertisements, while in the case of this paper, firms earn revenue through direct sales and through a variety of other means, such as advertisements and in-app, add-on purchases.

This paper differs from Sweeting (2013) in a number of important ways. First, demand in this model is bipartite: Developers face both an extensive margin of demand, where consumers make purchase decisions, and an intensive margin of demand, where consumers choose whether to use previously purchased products. It is through this intensive margin of demand that firms are able to earn revenue through various use-monetization methods. Beyond that, this model captures the fact that consumers are forward-looking in their decisions, a fact that is not relevant to studying consumer demand for radio stations.

2.4 Smartphone Applications

Finally, this paper contributes to a small, but growing literature on smartphone apps. Early work on this industry studied a large variety of topics including 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; Yin, Davis, and Muzyrya, 2014), and the relationship between mobile platforms and developers (Gans, 2012).

In recent years, there has been some work 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. This structure assumes a higher degree of competition between the Free apps in two different categories than between the Free and Paid apps in a single category. While I find strong evidence that consumers strongly prefer free to paid apps (beyond what would be expected given a difference in price of $0.99), it seems unlikely that a consumer considering a free note-taking app would consider a free app for playing solitaire to be a closer substitute than a note-taking app with a non-zero price.

Ghose and Han find evidence that consumer demand is higher the more recently an app has been updated, but their findings cannot be interpreted as a causal effect of updates on demand. This is primarily because they make use of standard instruments developed by Berry, Levinsohn, and Pakes (1995), which rely on the exogeneity of product characteristics. Thus, their finding could simply reflect the fact that higher quality products are updated more frequently.

Comino, Manenti, and Mariuzzo (2016) take a reduced form approach to studying how app updates affect 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.

Chapter 3

Industry Background

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). Table 1 shows the market shares for each of the major platforms in 2016.

The industry is a textbook example of a multi-sided market. Consumers purchase hardware, which is exclusively tied to one platform.[1] 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.b

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

[1]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.

Table 1: U.S. Market Shares of Mobile Platforms

Platform	Share (%)
Android	52.9
Apple	43.3
Microsoft	2.7
Blackberry	1.0
Other	0.1

Source: ComScore (2016). Data current as of March, 2016.

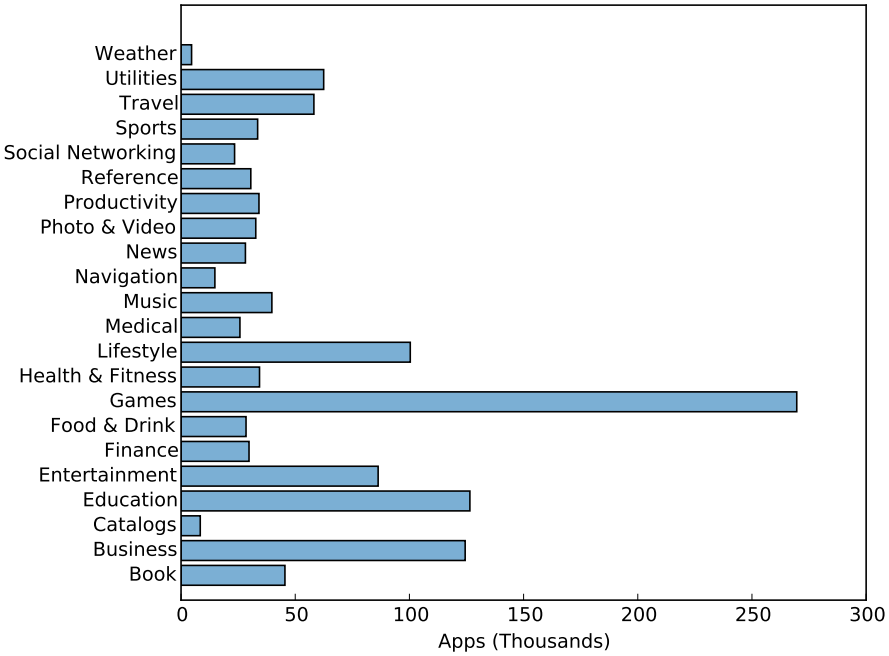
is specific to Apple, the general structure of the platform is applicable to the other mobile platforms.

3.1 Consumers and the App Store

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).[2] 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.

[2]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.

Figure 1: Number of Apps by Category (12/31/2014)



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. Figure 1 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 incremental updates throughout the year.

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.[3,4] 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.

3.2 App Development

To develop for the iOS platform, developers must join Apple's developer program, which costs $99/year per person. 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.[5] 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.

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

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

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

[5]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.

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. 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.

Apple's App Store marketplace is the only approved platform for selling apps for its mobile platform.[6] After a developer submits an app or app update to the App Store, Apple conducts a review process to ensure that the app does not violate a set of regulations Apple maintains for its platform. Once the app or app update is approved, the app is made available to consumers for purchase, or the version already available for sale is updated.

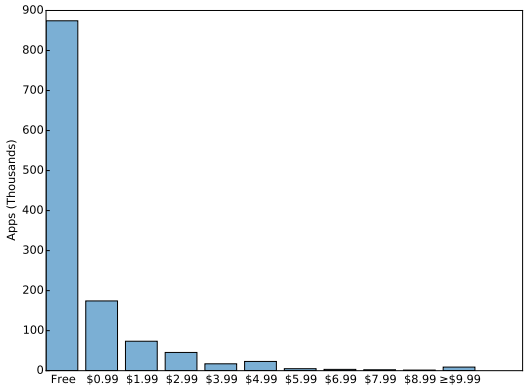
Developers set the price of their app, and are able to change that price at any time. Developers are restricted to a finite, discrete set of 94 possible prices. Apps can be free, or priced as low as \$0.99 and as high as \$999.99. Apple collects 30% of all sales revenue, which is similar to other platforms (Spencer, 2015).[7] Figures 2a and 2b 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

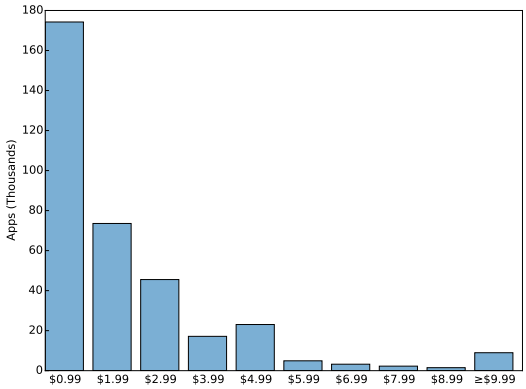
[6]Apps can be acquired illegally through online piracy. Piracy is relatively limited on the iOS platform, and is not considered here.

[7]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.

Figure 2: App Store Prices



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



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

offers an IAP for the "Pro" version of the product, which includes additional task organization capabilities. As with app sales revenue, Apple 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 practice has largely fallen out of fashion, particularly among non-game apps.

Chapter 4

Empirical Model

In this section, I 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 many apps earn revenue (in whole or in part) from consumers' intensive-margin *use* decisions. Due to a lack of data on consumers' intensive-margin behavior I am unable to estimate the structural model

of the intensive margin of demand developed here. Instead, I use a reduced form approximation of an app's total revenue when estimating the supply-side portion of this model.[1]

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

4.1 Timing

Time in this model is discrete, indexed by t. In the estimation 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 app. Developers face a choice of whether to not update their app, or to create a Major or Minor product update. As discussed in Chapter 3, Apple then reviews the update submission, and propagates the update to all previous purchasers of app j following that review.[2]

Following the updating decision and propagation of the update to consumers,

[1] See Section 5.2 for a discussion of how I approximate apps' per-period revenue, and Section 6.2 for how I estimate the supply-side model.

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

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.

4.2 A Model of the Demand for Smartphone Apps

Consumers face two separate decisions each period: whether to purchase a new app (extensive margin), and whether to use any already owned apps (intensive margin).

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

$$\begin{aligned} u_{ijt} &= X_{jt}\beta + \alpha p_{jt} + \xi_{jt} + \Lambda_{ijt} + \epsilon_{ijt} \\ u_{i0t} &= \epsilon_{i0t} \end{aligned} \tag{4.2.1}$$

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}.

$$s_{jt} = \frac{\exp(X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \xi_{jt})}{1 + \sum_k \exp(X_{kt}\beta + \alpha p_{kt} + \Lambda_{kt} + \xi_{kt})} \tag{4.2.2}$$

4.2.2 Intensive Margin of Demand

Simultaneously, consumers choose each period 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

$$\begin{aligned} w_{ijt} &= X_{jt}\tilde{\beta} + \tilde{\xi}_{jt} + \tilde{\epsilon}_{ijt} \\ w_{i0t} &= \epsilon_{i0t} \end{aligned} \tag{4.2.3}$$

and the fraction of previous purchasers who choose to use app j in period t is s_{jt}^{int}.

$$s_{jt} = \frac{\exp(X_{jt}\tilde{\beta} + \Lambda_{jt} + \tilde{\xi}_{jt})}{1 + \sum_k \exp(X_{kt}\tilde{\beta} + \Lambda_{kt} + \tilde{\xi}_{kt})} \tag{4.2.4}$$

Equation (4.2.3) differs from Equation (4.2.1) in three ways. First, consumers do not consider the price of the app when making an intensive-margin use decision, as they have already paid that one-time cost. Second, on the intensive margin $\Lambda_{jt} = 0$,

reflecting the fact that making period-specific use decisions is independent across periods.[3]

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. For example, the app developer Kevin Hoctor has noted that, "The limited information a prospective customer has prior to a purchase is one of the problems with Apples 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). 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 differences between (β, ξ) and $(\tilde{\beta}, \tilde{\xi})$ are unknown to the consumer prior to making a purchase, reflecting the idea that apps may in some ways be experience goods à la Nelson (1970).

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

$$AU_{jt} = s_{jt}^{ext} M_{m\tau} + s_{jt}^{int} \sum_{\tau=t_0}^{t-1} s_{j\tau}^{ext} M_{m\tau} \tag{4.2.5}$$

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. AU_{jt} can be interpreted as the "market size" a firm faces when trying to sell IAPs or subscriptions to its existing user-base.

[3]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. Game apps, the primary category where one might imagine this matters, are not considered in this paper.

4.2.3 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} and $\tilde{\xi}_{jt}$ (without loss of generality, I refer only to ξ_{jt} going forward). 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

$$\xi_{jt} = \rho\xi_{jt-1} + \mu_{Minor}1(a_{j,t} = \texttt{Minor}) + \mu_{Major}1(a_{j,t} = \texttt{Major}) + \eta_{jt} \tag{4.2.6}$$

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.

4.2.4 Expected, Discounted Future Utility (Λ_{jt})

Consumers consider the future utility of using the product when making their purchase decision. Consumers have expectations over future updates, and over future consumer-app match values ϵ_{ijt} (see Equation (4.2.3)), which are relevant to the consumer's period-specific, intensive margin, use decision. Thus, conditional on the developer's updating choice, 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 Equation (4.2.3). 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 "logsum" formula for the double exponential (Anderson, de Palma, and Thisse, 1992).

$$Ew_{ij\tau} = \ln\left(1 + e^{X_{j\tau}\tilde{\beta} + \tilde{\xi}_{j\tau}}\right) \tag{4.2.7}$$

Therefore, the expected, discounted future value of owning app j, Λ_{jt} is the dis-

counted sum of Equation (4.2.7) with expectations over the developer of app j's future updating decisions. Namely,

$$\begin{aligned}\Lambda_{ijt} &= E\left[\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} E w_{ijt}\right] \\ &= E\left[\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \ln\left(1+e^{X_{j\tau}\tilde{\beta}+\tilde{\xi}_{j\tau}}\right)\right]\end{aligned} \tag{4.2.8}$$

where δ is the discount factor, and t is the period of purchase. 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 extensive and intensive 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* purchasing an app, Λ cannot account for these differences.

4.3 A Dynamic Model of App Updating

4.3.1 App Developer's Profit Function

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.

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 $S_{k,j,t}$. In period t, developers choose an action $a_{jt} \in A$, and in doing so earn per-period profits

$$\begin{aligned}\pi_{j,t}(a_{j,t}, S_{k,j,t}; \theta^D) + \theta^{\epsilon}\epsilon(a_{jt}) = R_j(S_{k,j,t}, \theta^D)(1-\phi) \\ - \theta^{Minor}1(a_{j,t} = \texttt{Minor}) - \theta^{Major}1(a_{j,t} = \texttt{Major}) + \theta^{\epsilon}\epsilon_{j,t}(a_{j,t})\end{aligned} \tag{4.3.1}$$

R_j is the revenue the app earns, θ^D is the parameter from the demand model developed in Section 4.2, θ^{Minor} and θ^{Major} represent the fixed costs of updating, and 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 zero, as discussed in Chapter 3.

I assume that ϵ_{jt} is distributed i.i.d. Type I Extreme Value. This term captures the fact that apps may earn additional revenue in the given period that is not captured by the model and available data. For example, it was revealed in mid-2017 that every time users checked the weather using the weather tracking app AccuWeather, their location was passed on to a ad-targeting firm (Strafach, 2017). Such sources of information are not observable using data on the app store, as as such would not be captured by the revenue function $R_j(S_{k,j,t}, \theta^D)$.

I define $R_j(S_{k,j,t}, \theta^D)$ as

$$R_j(S_{kjt}) = \overbrace{p_j^{Retail} s_{j,t}^{ext} M_{mt}}^{\text{Sales Revenue}} + \overbrace{\left(\sum_l (p_{j,l}^{IAP} s_{j,l,t}^{IAP}) + p_j^{Ads} + p_j^{sub} s_{j,t}^{sub}\right) AU_{j,t}}^{\text{Use Revenue}} \tag{4.3.2}$$

where M_{mt} is the market size of market m in period t, s_{jt}^{x} is the share of active users making a purchase of $x \in \{IAP, Sub\}$, as appropriate, p^{Ads} is the per-user ad-price, 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.

Since I am unable to estimate the intensive margin model presented in Section 4.2.2, I use a reduced form approximation of an app's intensive margin revenue for estimating the supply model. Given this, we can simplify Equation (4.3.2) to be

$$R_j(S_{kjt}, \theta^D) = \overbrace{p_j^{Retail} s_{j,t} M_{mt}}^{\text{Direct Sales Revenue}} + (\texttt{Intensive Margin Revenue}) \tag{4.3.3}$$

4.3.2 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. Thus, a firm facing a discount factor of δ has expected future profits of

$$\sum_{t=0}^{\infty} \beta^t \Big(\pi(a_{j,t}, S_{j,t}) + \theta^{\epsilon} \epsilon_{j,t}(a_{j,t}) \Big) \tag{4.3.4}$$

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. Following Bellman's principle of optimality, the value to an optimally behaving firm in a given state, when its competitors are playing the set of

strategies σ, can be expressed as

$$V_j^{\sigma}(S_{k,j}, \epsilon_j) = \max_{a_j \in A}[\pi(a_j, S_{k,j}) + \theta^{\epsilon}\epsilon_j(a_j) + \beta \int V_j^{\sigma}(S_{k',j}) g(S_{k',q}|a, \sigma_{-j}, S_{k',j}) dS_{k',j}] \tag{4.3.5}$$

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

$$P^{\sigma_j}(a, S_{k,j}, \sigma_{-j}) = \frac{\exp\left(\frac{\nu_j^{\sigma}(a, S_{k,j}, \sigma_{-j})}{\theta^{\epsilon}}\right)}{\sum_{a' \in A} \exp\left(\frac{\nu_j^{\sigma}(a', S_{k,q}, \sigma_{-j})}{\theta^{\epsilon}}\right)} \tag{4.3.6}$$

where $\nu_j^{\sigma}(\cdot)$ is the choice-specific value function:

$$\nu_j^{\sigma}(a, S_{k,q}, \sigma_{-j}) = \pi(a, S_{k,j}) + \beta \int V_j^{\sigma}(S_{k',j}) g(S_{k',j}|a, \sigma_{-j}, S_{k',j}) dS_{k',j} \tag{4.3.7}$$

Returning to the equilibrium concept, a given set of strategies $\sigma^* = \{\sigma_j^*\}_{j \in J_m}$ is a Markov Perfect Nash Equilibrium if each firm j faces no incentive to deviate from σ_j^* given its competitors' strategies σ_{-j}^*. That is, letting $V(S_{jt}|\sigma_j^*, \sigma_{-j}^*, \theta^{Supply})$ denote the value to firm j of following σ_j^* while its competitors follow σ_{-j}^*, the equilibrium condition is

$$V(S_{jt}|\sigma_j^*, \sigma_{-j}^*, \theta^{Supply}) \geq V(S_{jt}|\sigma_j', \sigma_{-j}^*, \theta^{Supply}) \;\; \forall \sigma' \in \Sigma \tag{4.3.8}$$

Importantly, I assume that a Markov Perfect Nash Equilibrium exists. This assumption is required due to the fact that this model has continuous state variables

(download size, market size, and ξ_{jt}). Sweeting (2013) notes that it is possible to discretize these continuous state variables in order to then apply the proof of Doraszelski and Satterthwaite (2010) to show that a pure strategy equilibrium exists. Still, uniqueness would not be guaranteed. Rather than discretize these state variables, I follow Sweeting (2013) and others in the empirical dynamic games literature in assuming the existence of the equilibrium.

Chapter 5

Data

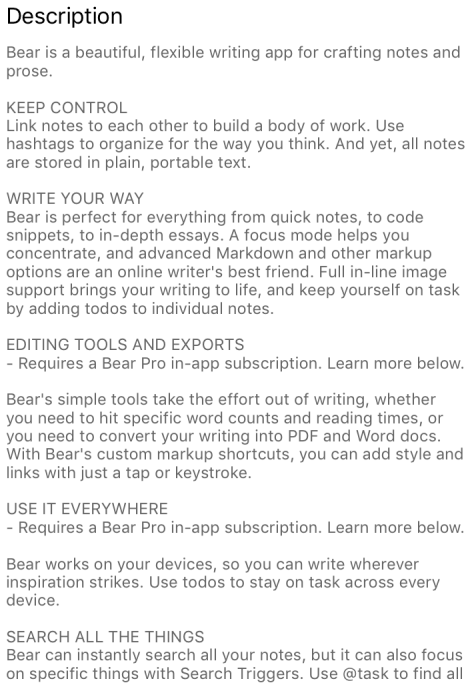
5.1 Data Collection and Sample Definition

5.1.1 Data Collection

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 Store on June 29, 2016, distributed across twenty-two categories (see Figure 1).

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 of each app update, both written by the developer. An example of an app

Figure 3: Example of an App Description



Description

Bear is a beautiful, flexible writing app for crafting notes and prose.

KEEP CONTROL
Link notes to each other to build a body of work. Use hashtags to organize for the way you think. And yet, all notes are stored in plain, portable text.

WRITE YOUR WAY
Bear is perfect for everything from quick notes, to code snippets, to in-depth essays. A focus mode helps you concentrate, and advanced Markdown and other markup options are an online writer's best friend. Full in-line image support brings your writing to life, and keep yourself on task by adding todos to individual notes.

EDITING TOOLS AND EXPORTS
- Requires a Bear Pro in-app subscription. Learn more below.

Bear's simple tools take the effort out of writing, whether you need to hit specific word counts and reading times, or you need to convert your writing into PDF and Word docs. With Bear's custom markup shortcuts, you can add style and links with just a tap or keystroke.

USE IT EVERYWHERE
- Requires a Bear Pro in-app subscription. Learn more below.

Bear works on your devices, so you can write wherever inspiration strikes. Use todos to stay on task across every device.

SEARCH ALL THE THINGS
Bear can instantly search all your notes, but it can also focus on specific things with Search Triggers. Use @task to find all

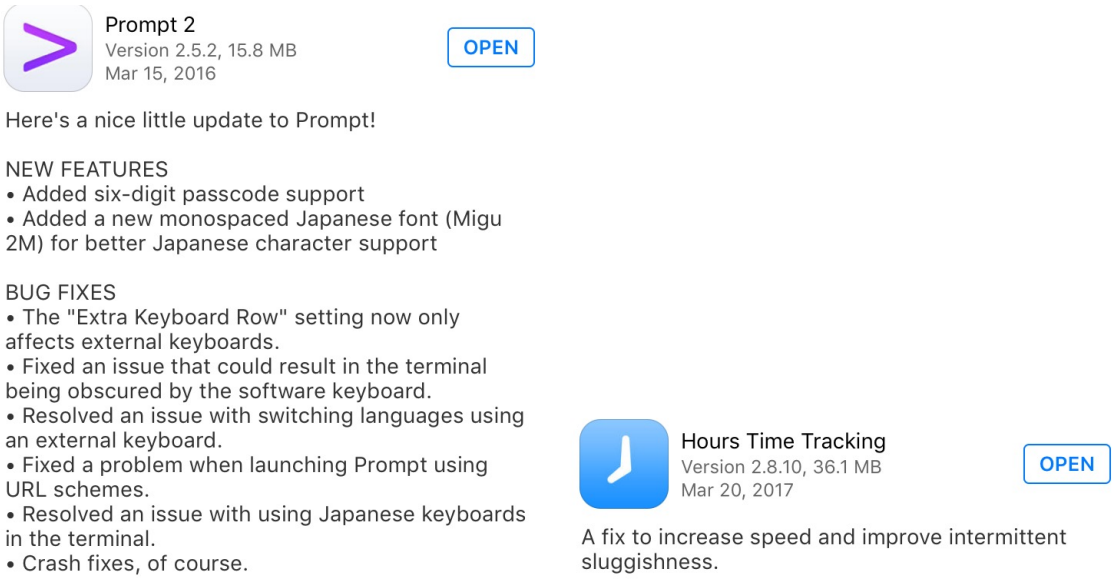
description and an app update description, called release notes, are shown in Figures 3 and 4, respectively. Finally, I use data from the app analytics company AppFigures on the presence of advertisements for apps in the Productivity category.[1]

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). In all cases, I observe the top 500 apps for each day in the respective lists.

I face two challenges arising from the unusual mix of products that are sold on the App Store. First, the development of many apps is not profit maximizing, which can make the application of standard empirical economic techniques inappropriate.

[1]Section 5.3 describes the Major and Minor updates distinction that is referenced in Figure 4

Figure 4: Update Type Examples



(a) Major Update

(b) Minor Update

Second, many products on the App Store have been effectively abandoned by their developers, and are no longer competitive in the marketplace. In what follows, I discuss these two issues and how I account for them.

5.1.2 Product versus Companion Apps

Apps can be broadly 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 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 little value to the consumer. 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, on the other hand, are those that are themselves the product being offered. The weather app, Weather Atlas, which provides forecasts and a weather radar, is produced as a standalone product. This distinction is important, because while Product apps can be reasonably 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.[2] 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.

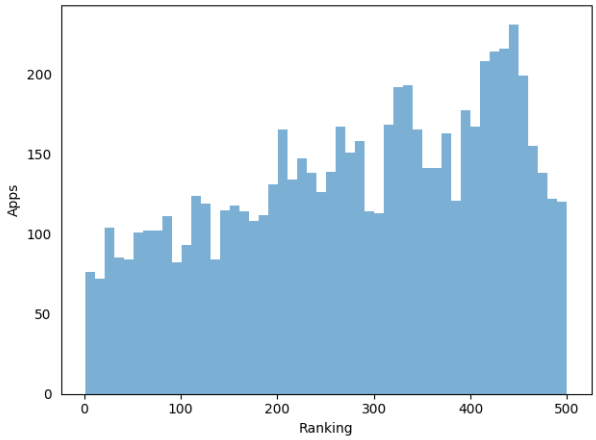
In light of this, I limit my attention to the Productivity category of apps. While many categories within the App Store consist of a high proportion of Companion apps, such as the Social Networking and Catalogs categories, the Productivity category is predominantly made up of Product apps.

5.1.3 Abandoned Apps

A second issue in defining a sample in the App Store stems from the fact that, as discussed in Chapter 3, 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 strategic actors in the marketplace. It is important to re-emphasize the fact that, since app development

[2]Accounting for the pricing and updating of a companion app would require modeling the larger product offering. For example, decisions make regarding Target's app 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 5: Distribution of Productivity Apps' Best Ranking

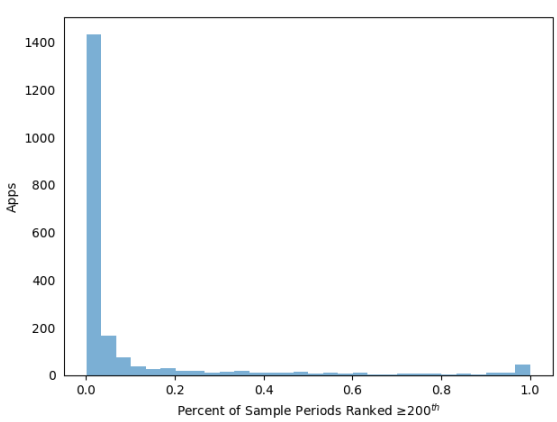


primarily consists of fixed costs, developers face little to no costs in keeping an abandoned product in the store. Unlike the production of physical goods, continuing to sell an abandoned app does not, in general, result in continued costs.

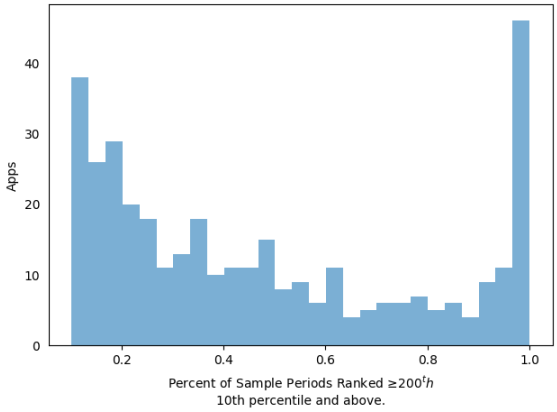
In the Productivity category, only 10.6% apps ever reach a sales ranking better than 500^{th} in the sample period, and only 3.2% ever reach a ranking better than 200. Figure 5 shows the distribution of apps' best rankings for those that reach a ranking of at least 500 on at least one occasion. However, even among apps that do reach a particular ranking threshold, few apps are able to repeatedly maintain sales above that threshold. Figure 6 shows that among apps that achieve at least a ranking of 200, nearly all spend less than 10% 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

Figure 6: Persistence Above Ranking Threshold of 200



(a) Periods ranked above 200



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

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.[3] 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 Chapter 4, 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 4.2. 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

[3]See Boudreau (2018) for a discussion of the effects of amateurs on mobile app platforms.

throughout much of, and in many cases, all of, the sample period.[4]

5.1.4 Sample Definition

In light of these challenges I make a number of restrictions to the sampled used when estimating the structural model. First, I focus my analysis on apps in the Productivity category, as apps a large majority of the apps in that category can be classified as Product, as opposed to Companion, apps. In order to address the abandoned app issue, 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 5.5.

In order to estimate the model in this paper I must first address three issues in the data. First, in Section 5.2 I account for the fact that sales data is, in general, unavailable for this industry, then, in Section 5.3, I use natural language processing (NLP) and machine learning (ML) techniques to classify updates based on their content. Then, in Section 5.4 I again use NLP and ML techniques to develop a method for defining markets in online product markets.

5.2 Estimating App Sales

Estimating the demand model developed in Section 4.2 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,

[4] 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.

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 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.[5] 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.[6]

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

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

[6]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.

roughly rank apps by sales.[7,8] 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 $\bar{S}$,

$$\Pr[\text{sales}_{jt} > \bar{S}] = \left(\frac{k}{\bar{S}}\right)^{\beta} \tag{5.2.1}$$

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

$$\frac{\text{rank}_{jt}}{|J|} = \left(\frac{k}{\bar{S}}\right)^{\beta} \tag{5.2.2}$$

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

$$\ln(\text{rank}_{jt}) = \overbrace{\big(\beta \ln(k) + \ln(|J|)\big)}^{\alpha} - \beta \ln(\text{sales}_{jt}) \tag{5.2.3}$$

I estimate Equation (5.2.3) using publicly observable ranking data, and sales quantity data I have collected directly from some of the developers in the sample. Table 2

[7] 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).

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

Table 2: 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.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

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 Equation (5.2.3) 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.

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-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.[9]

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 2, 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 3. These estimates are used in the estimation of the supply model (see Section 6.2).

[9]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.

Table 3: 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.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

5.3 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. 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.

I use two methods to classify app updates as Major or Minor updates. The first approach, which has been used previously in research on smartphone apps, uses

changes in an app's version number to identify the degree to which an update changes an app. In light of concerns about using version numbers to classify app updates, which I discuss below, I propose a new approach for classifying the contents, which takes advantage of the fact that developers write text descriptions of each app update, called release notes.

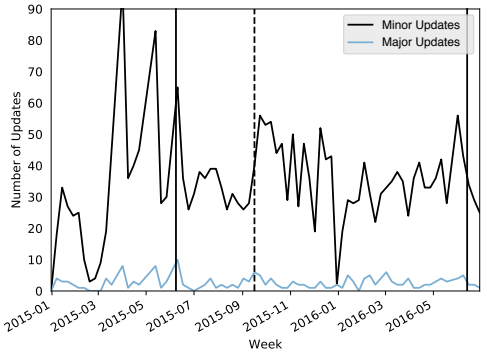
5.3.1 Version Number Classification

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 to 5.0.0 typically indicates a major revision, whereas an update from version 4.3 to 4.4 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 to 5.0 may only include minor changes to the app despite the "large" change in the version number. I code any update where the first number in the version number changes (e.g., 4.3 to 5.0) as a Major update, and any update where any of the subsequent numbers change (e.g., 4.3 to 4.4) as a Minor update. Figure 7a shows the number of Major and Minor updates over the sample period, and Figure 7b shows the ratio of Major to Minor updates.

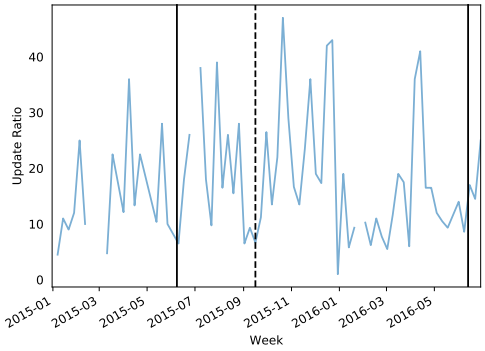
While changes in an app's version number can be informative about the importance of an update, the lack of a clearly defined set of rules for version numbering means they don't provide a clear sense of the *content* of an update. In fact, many apps in the App Store have Major version number changes where only small adjust-

Figure 7: Updates By Type

Version Number Classification

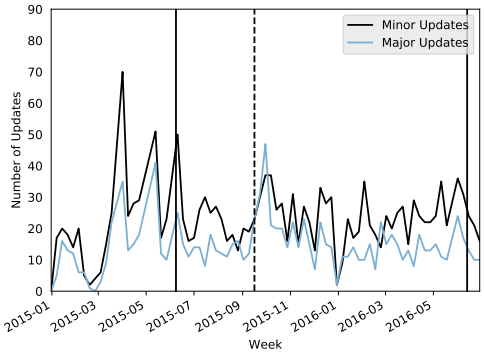


(a) App Updates by Type

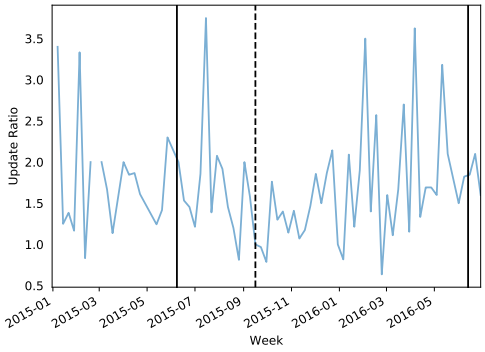


(b) Ratio of Minor to Major Updates

SVM Classification



(c) App Updates by Type



(d) 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.

ments have been made, while many Minor version number changes correspond to the addition of important new features. To circumvent this issue, and to get a more precise sense of the content of an update, I make use of each update's release notes in order to classify updates solely by their content.

5.3.2 SVM Classification

Developers provide text descriptions of each update, called release notes, which outline the specific changes included in a particular update. 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. Figure 8 outlines the set of rules under which the training set was built.

Prior to classifying the updates, I preprocess the updates' release notes using natural language processing (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.[10] 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

[10]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 8: 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.

than, for example, the frequency with which a developer mentions Apple's platform.

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

Table 4: 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.

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

$$weight_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \times \ln\left(\frac{N}{n_i}\right) \tag{5.3.1}$$

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

Table 5: Cross Tabulation of Update Classifications

		Version Number		
		Minor	Major	Total
SVM	Minor	1,591	90	1,681
	Major	939	135	1,074
	Total	2530	225	2,755

tests are shown in Table 4. Figure 7c shows the number of Major and Minor updates over the sample period, and Figure 7d 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 updates in that period.

Figure 7 provides an easy way to visually assess the differences between the Version Number and SVM classification methods. The clearest difference is the fact that the Version Number classification approach identifies significantly fewer Major updates than the SVM approach. Thus, the Version Number classification system appears to understate the intensity of product innovation taking place in the industry. Table 5 shows that 939, or 34% of the updates classified as Minor by the Version Number approach, add new functionality or content, as identified by the SVM approach. Additionally, there are some cases (accounting for 3.3% of total updates) where the Version Number approach would indicate a major improvement to the app when only minor changes had been made. As a result of the fact that changes in an app's version number misclassifies the content of an app update over a third of the time, the SVM classification approach is the preferred approach in the analysis that follows.

5.4 Defining Markets

5.4.1 Challenges in Defining Markets in Digital Industries

In addition to accounting for app sales, estimating the model developed in Chapter 4 requires defining markets within the Productivity category of the App Store. While 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).

In recent years, new approaches that do not involve price elasticity calculations have been suggested. 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).

However, these new methods are difficult to implement given the wide variety of

ways products are monetized in this industry. 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.[11] 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.

5.4.2 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.[12]

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,

[11] 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.

[12] 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.

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.

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.

5.4.3 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 5.3. 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

$$\gamma_{ij} = \begin{cases} 1, & x_i \text{ is in cluster } j \\ 0, & \text{otherwise} \end{cases} \tag{5.4.1}$$

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

$$\min_{\gamma,\delta} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{k} \gamma_{ij} ||x_i - \delta_j||^2 \tag{5.4.2}$$

Equation (5.4.2) 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

$$\text{Step 1:} \qquad \min_{j'} ||x_i - \delta_{j'}||^2 \ \forall i \tag{5.4.3}$$

$$\text{Step 2:} \qquad \sum_{i=1}^{N} \gamma_{ij}(x_i - \delta_j) = 0, \ \forall j \tag{5.4.4}$$

This two-step process is repeated until the convergence of γ and δ. The op-

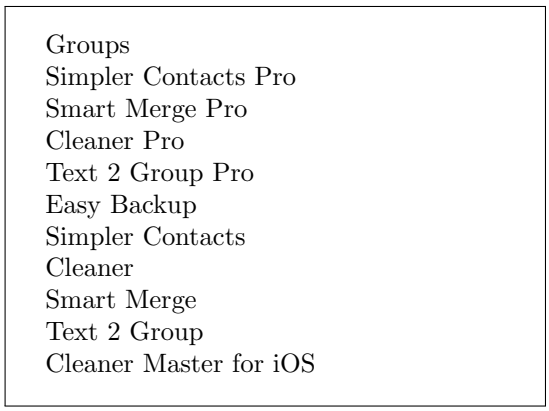
timal number of markets (i.e., clusters) and the optimal parameters for the text pre-processing stage are determined by repeating the clustering algorithm across a parameter grid and then selecting the vector of parameters that maximize a scoring metric. In particular, I use the silhouette score, which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

5.5 Descriptive Evidence

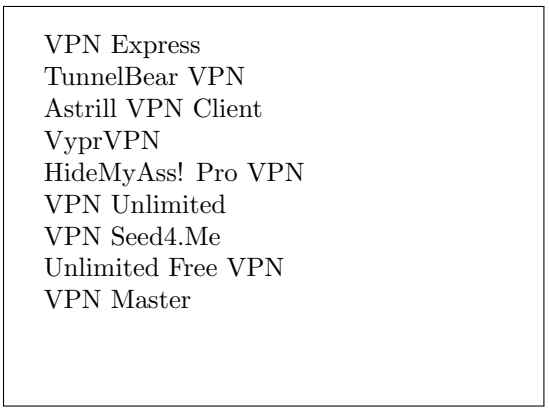
Restricting the sample to apps that ranked at or better than 200^{th} for at least 10% of the sample periods results in a sample of 352 apps. The market definition technique proposed in Section 5.4 identifies 39 markets, with between 1 and 32 apps per market. Figure 9 shows four of the markets determined 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.[13] Table 6 provides summary statistics for each market.

[13]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 Chapter 6 because Tody has a monopoly on its market.

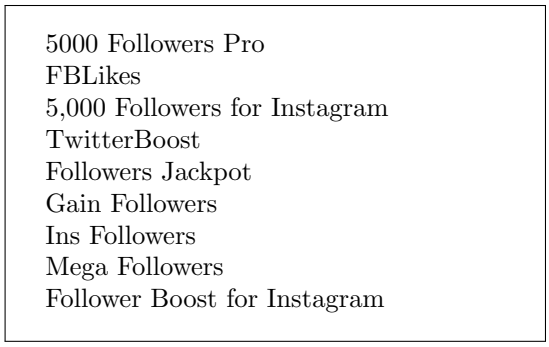
Figure 9: Example Markets



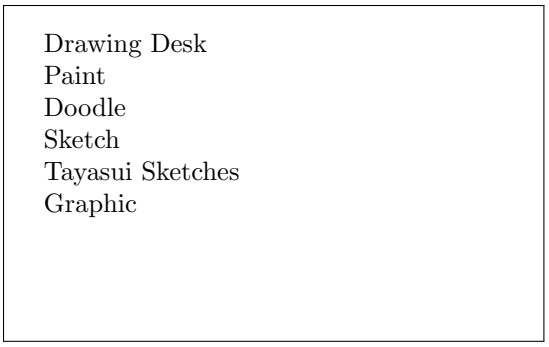
(a) Market 4 - Text Messaging



(b) Market 5 - Virtual Private Network



(c) Market 34 - Instagram Management



(d) Market 36 - Drawing

Table 6: 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. $N = 23,897$

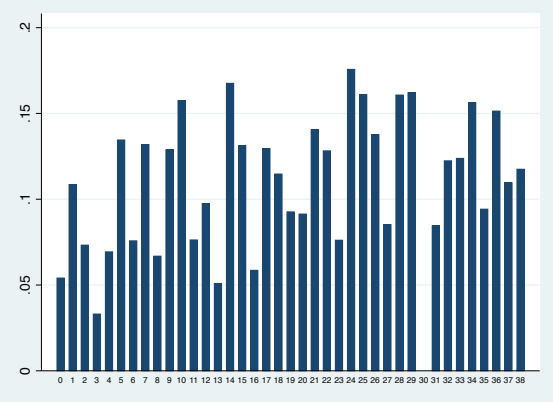
5.5.1 Updating

Figure 10a shows the average frequency of updates across markets, and Figure 10b shows the frequency of Major and Minor updates, conditional on updating. For example, we can see that when apps in market 13 update, they only produce Minor updates, while in markets 26 and 27 an app is just slightly more likely to submit a Major update than a Minor update when updating. 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. Figures 10c and 10d show the app-level distributions for the frequency of updating, and the conditional frequency of submitting a Major or Minor update. Figure 10c shows that most apps update very infrequently, though some update approximately every three weeks. Figure 10d shows that apps are, in general, more likely to submit Minor updates, but that there are some apps that focus almost entirely on submitting just one type of update.

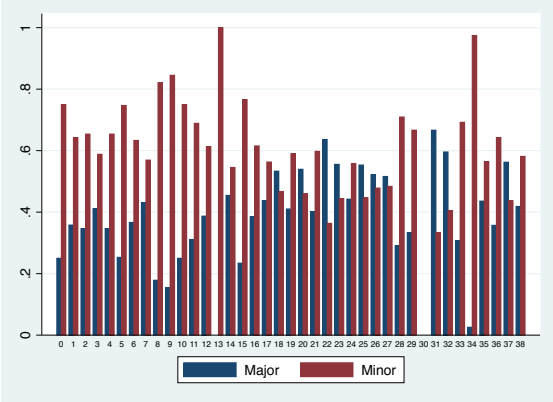
5.5.2 Monetization

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 Figure 11. Just under half (49%) of the apps in the sample are free to purchase. Figure 12 shows the percent of apps in each market that employ a particular pricing strategy. As is evident from Figures 12a and 12b, non-zero pricing and the use of IAPs are most common. Note that developers can combine monetization strategies, such as in market 29, where both of the apps in that market charge a non-zero price, sell IAPs, and offer a subscription service.

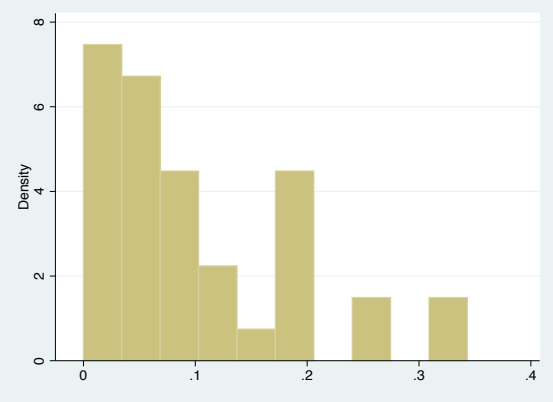
Figure 10: Updating Behavior



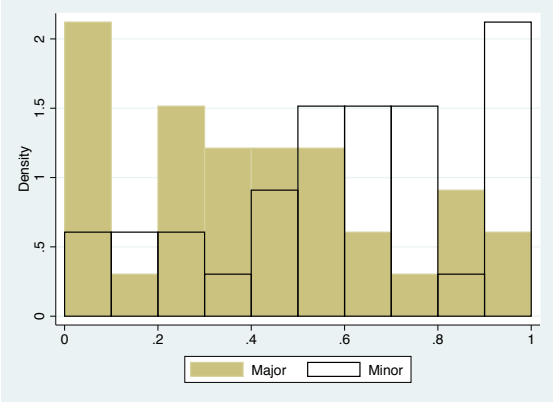
(a) Frequency of Updating, By Market



(b) Frequency of Update Types, Conditional on Updating, By Market



(c) Distribution of Frequency of Updating, by App



(d) Distribution of Frequency of Update Types, Conditional on Updating, By App

Figure 11: Distribution of Prices in Estimation Sample (On 12/31/14)

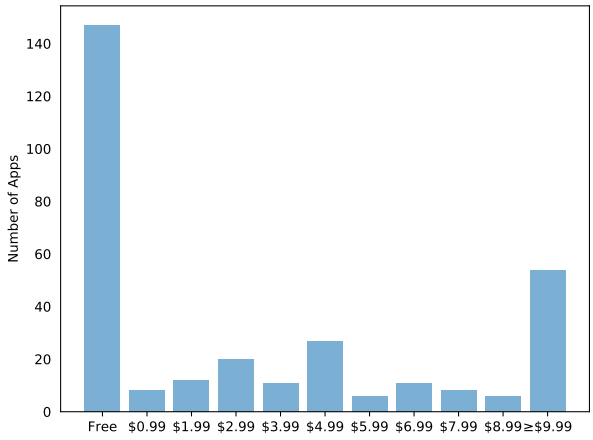
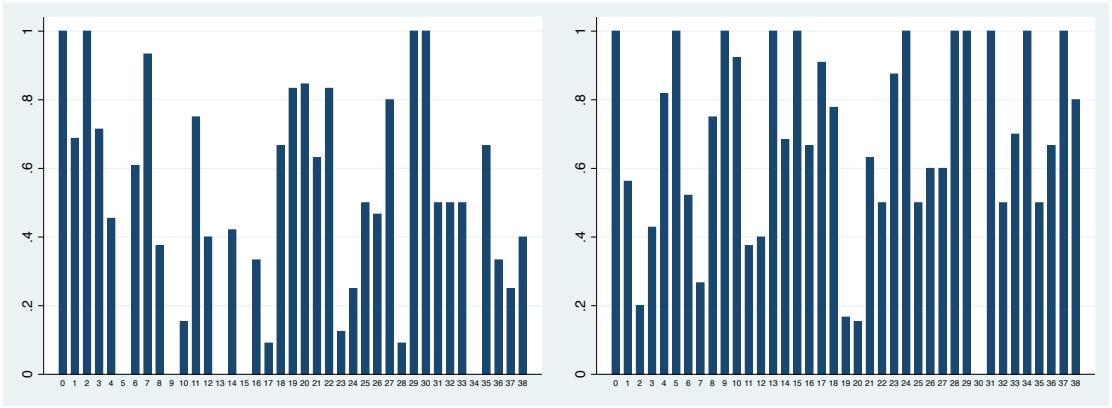
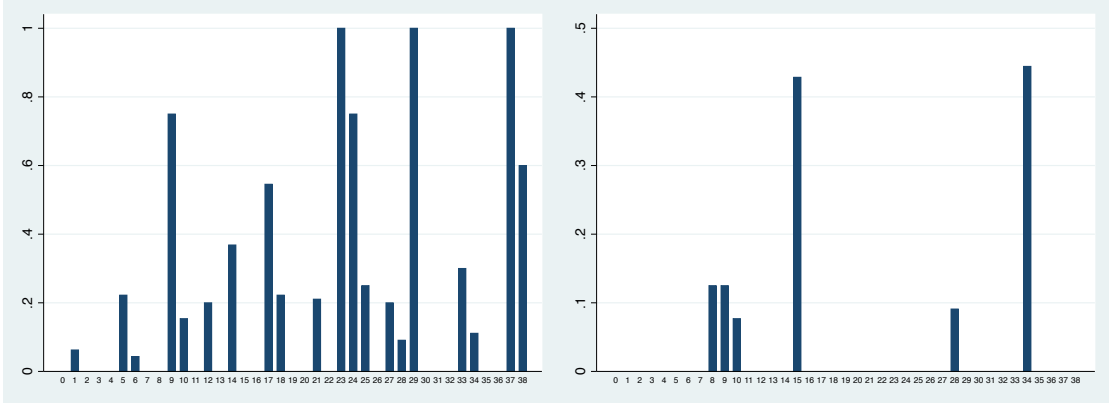


Figure 12: Presence of Monetization Strategies, by Market



(a) Percent of Apps with Non-zero Prices

(b) Percent of Apps with IAPs



(c) Percent of Apps with Subscriptions

(d) Percent of Apps with Advertisements

Table 7: 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

5.5.3 Preliminary Regression Analysis

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 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 Chapter 4 and the counterfactual analysis conducted in Chapter 8 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 7). 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 7 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.

Chapter 6

Estimation

I estimate the model in Section 4.2 using the sample and markets defined in Section 5.4.

6.1 Demand Estimation

6.1.1 Estimation Strategy

Recall that the demand model (Equation (4.2.1)) 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. This gives

$$\ln(s_{jt}) - \ln(s_{0t}) = X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \xi_{jt} \tag{6.1.1}$$

In a typical application of a logit demand model, Equation (6.1.1) can be estimated using ordinary least squares or two-stage least squares. However, because ξ follows the law of motion defined in Section 8.1, 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 Equation (6.1.1). Given this, and plugging in the ξ law of motion, the standard estimating equation, Equation (6.1.1), can be re-written as

$$y_{jt} = X_{jt}\beta + \alpha p_{jt} + \Lambda_{jt} + \rho\xi_{jt-1} + \mu_{Minor}1(a_{j,t} = \texttt{Minor}) - \mu_{Major}1(a_{j,t} = \texttt{Major}) + \eta_{jt} \tag{6.1.2}$$

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

$$\begin{aligned} y_{jt} - \rho y_{jt-1} = (X_{jt} - \rho X_{jt-1})\beta + \alpha(p_{jt} - \rho p_{jt-1}) + (\Lambda_{jt} - \Lambda_{jt-1}) \\ + \mu_{Minor}1(a_{j,t} = \texttt{Minor}) - \mu_{Major}1(a_{j,t} = \texttt{Major}) + \eta_{jt} \end{aligned} \tag{6.1.3}$$

I estimate Equation (6.1.3) using non-linear least squares (NLLS).

6.1.2 Calculating Λ_{jt}

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, as opposed to those parameters' intensive-margin counterparts, $\tilde{\beta}$ and $\tilde{\xi}$. 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.

I calculate Λ_{jt} using forward simulation. For each candidate parameter in the estimation of the demand model, I do the following. For each observation, simulate forward 50 periods, calculating Ew_{ijt} for each period. Then take the discounted sum of each Ew_{ijt}, 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}.

This process is computationally expensive. In order to make this estimation feasible, I first calculate and save each simulated path for each observation, using the observed conditional choice probabilities to determine when developers will update their app. Then, for each candidate parameter in the estimation procedure I just have to calculate Ew_{ijt} for each period of each path, take the discounted sum for each path, and then average across all of the simulations.

6.1.3 Capturing Non-linear Price Effects

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 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.

6.1.4 Endogeneity Concerns

As discussed in Chapter 3, 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.[1] This is similar to the approach taken in Doraszelski, Lewis, and Pakes (2016).

6.2 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.

[1] 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 8: 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	Section 8.1	Firm-level
ξ_{-j}	Section 8.1	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.

6.2.1 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 8 provides a list of state variables in the model, and a description of their evolution processes.

Following BBL, I rewrite Equation (4.3.5) to express a firm's value function as a

linear function of the parameters to be estimated.

$$\begin{aligned} V(S_{kjt}|\sigma_j,\sigma_{-j}) =& E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t R_j(S_{kjt},\theta^D) \\ &-\theta^{Minor} E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t 1(a_{jt}=\texttt{Minor}) \\ &-\theta^{Major} E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t 1(a_{jt}=\texttt{Major}) \\ &+\theta^{\epsilon} E_{j,\sigma_j,\sigma_{-j}} \sum_{t=0}^{\infty} \delta^t \epsilon_{jt}(a_{jt}) \\ =& \mathbf{R}_{j,\sigma_j,\sigma_{-j}} - \theta^{Minor}\mathbf{F}^{Minor}_{j,\sigma_j,\sigma_{-j}} - \theta^{Major}\mathbf{F}^{Major}_{j,\sigma_j,\sigma_{-j}} + \theta^{\epsilon}\boldsymbol{\epsilon}_{j,\sigma_j,\sigma_{-j}} \end{aligned} \tag{6.2.1}$$

As discussed in Chapter 4, 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.[2]

$$R_j(S_{k,j}) = \sum_{x\in\Phi} \phi_x(S_{k,j}) + \gamma_j(S_{k,j}) \tag{6.2.2}$$

Using the results of this 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. The value function used

[2]Revenue data for the firms is calculated using the Top Grossing ranking list. See Section 5.2 for more details.

in estimating this model is

$$V(S_{kjt}|\sigma_j, \sigma_{-j}) = \tilde{\mathbf{R}}_{j,\sigma_j,\sigma_{-j}} - \theta^{Minor}\mathbf{F}^{Minor}_{j,\sigma_j,\sigma_{-j}} - \theta^{Major}\mathbf{F}^{Major}_{j,\sigma_j,\sigma_{-j}} + \theta^{\epsilon}\boldsymbol{\epsilon}_{j,\sigma_j,\sigma_{-j}} \quad (6.2.3)$$

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 $\theta^S = (\theta^{BF}, \theta^F, \theta^{\epsilon})$ 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 a discount factor of $\delta = 0.95$.

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 based 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, this entails:

    - Re-calculating the download size for any updated apps according to the

estimated AR-1 process. If any apps were updated, the market average download size is also updated.

- Updating the last-period update status for all apps in the market.
- Updating ξ_j for all apps following Section 8.1.

4. Given the updated state, calculate the firms' revenues using the estimated revenue model 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 $\boldsymbol{R}_p$, $\boldsymbol{F}_p^{Minor}$, $\boldsymbol{F}_p^{Major}$, and $\boldsymbol{\epsilon}_p$ under alternative policies $p \in P$. In practice, I use eight alternative policies. I estimate a multinomial logit model in order to calculate developers' conditional choice probabilities, which express the likelihood of choosing each possible action conditional on the current state of the marketplace. The multinomial logit model is estimated using the state variables, and a large number of interactions of the state variables as covariates. When calculating each term of the value function, I average across 250 simulated paths for each initial state that is considered.

6.2.2 Second Stage

Recall from Section 6.2 that the equilibrium condition for a Markov Perfect Nash Equilibrium implies that

$$V(S_{kjt}|\sigma_j^*, \sigma_{-j}^*, \theta^{Supply}) \geq V(S_{kjt}|\sigma_j^p, \sigma_{-j}^*, \theta^{Supply}) \; \forall p \in P \tag{6.2.4}$$

Table 9: List of Alternative Policies

- All updates are Minor updates
- All updates are Major updates
- Developers never update
- Developers always update, using observed relative frequency of Major to Minor updates
- Minor updates are 5% more likely
- Major updates are 5% more likely
- Minor updates are 5% less likely
- Major updates are 5% less likely

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

$$\theta^S = \arg\min_{\theta} \sum_{p \in P} \left\{ V_{j,\sigma_j^p,\sigma_{-j}^*} - V_{j,\sigma_j^*,\sigma_{-j}^*})\theta, 0 \right\}^2 \tag{6.2.5}$$

Chapter 7

Estimation Results

7.1 Demand Results

Table 10 shows the results of estimating the demand model. 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, specification (3) is preferred. 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 5.3.

Table 10: NLLS Demand Estimates - SVM Classification

	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.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

7.1.1 Updates

SVM Classification

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. On average, approximately half of the effect an update remains after a month, 30% after 2 months, and 2% remains after 6 months. 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 10, consumers respond to both Major and Minor updates, but they do not appear to react differently to the different types of updates.[1] The fact that consumers do not respond differently between Major and Minor updates shows that 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.

Version Number Classification

In Table 11 I show the results of estimating the demand model using the Version Number classification approach. As discussed in Section 5.3.1, this approach classifies

[1] 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).

updates as Major or Minor based on whether or not the first number on a subsequent number in an app's version number changes. Using this classification approach, I find a larger effect on Major updates on demand than I did under the SVM classification. It is unclear how to interpret this finding. On one hand, it is possible that "major" changes in the version number reflect the most significant improvements to a product, reflected by this larger estimated coefficient. On the other hand, as discussed in Section 5.3.1, developers often use large changes in a version number as a promotional tool, so this larger estimated coefficient may simply reflect a response to this form of promotion.

7.1.2 Monetization

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), I find that 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 11: NLLS Demand Estimates - Version Number Classification

	Update Types		
	(1)	(2)	(3)
Major	0.1672*	0.1591*	0.1622*
	(0.0926)	(0.0924)	(0.0923)
Minor	0.0880***	0.0817***	0.0850***
	(0.0273)	(0.0273)	(0.0272)
Price	-0.0349***		-0.0310***
	(0.0048)		(0.0049)
Paid		-0.9149***	-0.6837***
		(0.1723)	(0.1731)
IAP	0.4171***	0.0331	-0.0173
	(0.1334)	(0.1737)	(0.1705)
Ad	0.2892	0.1410	0.1441
	(0.3660)	(0.3712)	(0.3647)
Subscription	-0.3074**	-0.3381***	-0.3685***
	(0.1218)	(0.1230)	(0.1217)
$\ln(size)$	0.1930***	0.1764***	0.1946***
	(0.0486)	(0.0487)	(0.0479)
ρ	0.8609***	0.8623***	0.8599***
	(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 12 shows the results when the model is estimated using non-linear, two-stage least squares (NLTSLS). When I instrument for prices, following Doraszelski, Lewis, and Pakes (2016), in specification (1) I find that the price coefficient is larger in magnitude compared to the NLLS results. The disparity between the NLLS and NLTSLS price coefficients in specification (3), however, is smaller than in specification (1). This 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 relationship with demand, 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 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

Table 12: NLTSLS Demand Estimates - SVM Classification

	Update		Update Types	
	(1)	(3)	(1)	(3)
Update	0.0927***	0.0899***		
	(0.0264)	(0.0264)		
Feature			0.0716*	0.0690*
			(0.0402)	(0.0402)
Bug Fix			0.1060***	0.1030***
			(0.0326)	(0.0326)
Price	-0.0552***	-0.0463***	-0.0552***	-0.0462***
	(0.0080)	(0.0084)	(0.0080)	(0.0084)
Paid		-0.6130***		-0.6128***
		(0.1774)		(0.1774)
IAP	0.2805**	-0.0719	0.2795**	-0.0728
	(0.1394)	(0.1706)	(0.1394)	(0.1706)
Ad	0.2576	0.1359	0.2480	0.1265
	(0.3646)	(0.3638)	(0.3650)	(0.3641)
Subscription	-0.3357***	-0.3827***	-0.3361***	-0.3832***
	(0.1217)	(0.1217)	(0.1217)	(0.1217)
$\ln(size)$	0.2055***	0.2038***	0.2072***	0.2055***
	(0.0485)	(0.0479)	(0.0486)	(0.0480)
ρ	0.8603***	0.8595***	0.8602***	0.8594***
	(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.
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

this selection effect would require modeling apps' entry decisions into the available forms of monetization.

7.2 Fixed Cost Results

Table 13 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 7.1, 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.

Finally, it is important to note that these estimated fixed costs reflect both differ-

Table 13: 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.

ences in the cost of producing Major and Minor updates, as well as any differences in the cost of generating ideas for those updates. The model I estimate assumes firms have the option of submitting either a Major or Minor update each period. An alternative possibility is that the ideas for Major and Minor updates arrive at different rates, and that in any given period developers might only have the option of one type (or no types) of updates. Since the arrival rates are not separately identified from the fixed costs of updates, I only estimate the fixed costs. Thus, these costs reflect any differences in the arrival rate of update ideas.

7.3 Understanding the Intensive Margin

Taken together, the demand and supply results appear to present a contradiction: In Section 7.2, I show that Major updates are more expensive to produce than Minor updates, yet Table 10 shows that demand is not differentially affected by Major updates. Despite these two facts, as evidenced by Figure 4 and Table 6, 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.

In particular, data on consumer use behavior — how often consumers use an app, how intensely they use it, and their IAP/subscription purchases — would be required in order to address both the apparent contradiction identified here, and the other questions regarding the overall welfare effects of product digitization.[2] This data would make it possible to estimate a model of intensive-margin demand, which would model both under which conditions consumers use their already purchased apps, and under which conditions consumers purchase IAPs and subscriptions.

[2]See Chapter 9 for further discussion.

Chapter 8

Counterfactual

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 use monetization and post-purchase updating. 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 where 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.

8.1 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 fixed at zero in the counterfactual regime. In terms of the incentives firms face, firms no longer consider the effect of updates on consumers' intensive margin, use 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 Chapter 4, 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

$$R_j^{counter}(S_{kjt}) = \overbrace{p_j^{Retail} s_{j,t}^{ext} M_{mt}}^{\text{Direct Sales Revenue}}, \tag{8.1.1}$$

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

$$\Lambda_{ijt}^{counter} = \sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \ln\left(1 + e^{X_{j\tau}\beta + \xi_{j\tau}}\right), \tag{8.1.2}$$

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 $\tau' > \tau$. That is,

$$\xi_{j\tau} = \rho\xi_{j\tau-1} + \eta_{j\tau} \quad \forall\tau$$

8.2 Solving the Model and Forward Simulation

In order to determine counterfactual updating behavior, I simulate updating decisions for each market using forward simulation. Markets are treated independently, and are simulated forward 50 periods. 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.

8.3 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 4.3, 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. 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 $\{0.99, 9.99, 19.99, 29.99, 39.99\}$.

8.4 Counterfactual Results

Table 14 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. Given the requirement of simulating the updating game for each pricing equilibrium, considering markets with more than four firms is computationally infeasible. Table 14 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.

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

Table 14: Counterfactual Results

	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

and Minor updates, though the rate of Major updates increases by slightly more than the rate of 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.

Chapter 9

Conclusion

9.1 Summary of Findings and Discussion

As consumer goods undergo a process of digitization, firms gain access to new strategic tools, including 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 of Major, feature-adding updates compared to 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. Understanding the role of market-level heterogeneity in driving these results 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, such as text or 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 in economic analysis.

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 selling digital products.

9.2 Future Work

The research presented here opens a number of directions for future research. First, while this research studies the first-order issue of how the new tools made available to firms through digitization affect product innovation, it will be important to understand to what extent the physical attributes of products undergoing digitization attenuates (or, perhaps, amplifies) the effects found in this study.

Additionally, there is a clear need for more research on intensive margin behavior. Understanding how consumers behave *after* purchasing digital or partially digital products will be essential for understanding the new, and potentially rapidly chang-

ing, incentives firms face regarding monetization and product innovation. Studying consumer behavior on this margin will present new challenges, as data regarding intensive-margin behavior will likely be even more limited than aggregate or transaction-level sales data has traditionally been. Given this, it is no surprise that academic researchers are increasingly partnering with firms to study these industries.

Ultimately, more detailed analysis of intensive-margin behavior will allow researchers to address the fundamental question of whether product digitization is welfare enhancing. At present, it is ambiguous whether these changes result in increased welfare, and it is especially uncertain whether these changes improve or diminish consumer surplus. While the work presented here shows clear evidence of an increase in the frequency of product innovation, it is possible that firms are able to extract the value of those innovations from consumers via the new forms of monetization that are available to firms. Alternatively, firms may enter at a lower quality than they would absent product digitization, which could result in overall reductions in consumer welfare, despite the more aggressive product updating.

Research on the welfare effects of product digitization should prove useful to both researchers studying how these technological changes affect society, as well as regulators and policymakers as they consider how to properly regulate industries in the digital age. As firms begin to monetize their products through use monetization, it may become increasingly difficult to measure market power, and to determine appropriate remedies in the case of anti-competitive behavior. Additionally, with the growth of digital platforms, a small number of firms may hold undue influence over a large swath of firms, which may raise additional regulatory concerns. Overall, it is clear that there is ample need for future research on competition and innovation in digital, and digitizing industries.

Bibliography

Aghion, Philippe, Nick Bloom, Richard Blundell, Rachel Griffith, and Peter Howitt. 2005. "Competition and Innovation: An Inverted-U Relationship." *The Quarterly Journal of Economics* 120 (2):701–728. 9

Anderson, Simon P., André de Palma, and Jacques-François Thisse. 1992. *Discrete Choice Theory of Product Differentiation*. The MIT Press. 29

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

Arrow, Kenneth. 1962. "Economic welfare and the allocation of resources for invention." *National Bureau of Economical research: The Rate and Direction of Inventive Activity: Economic and Social Factors* I:S. 609–626. URL http://www.nber.org/chapters/c2144.pdf. 8

Aryal, Gaurab, Federico Ciliberto, and Benjamin T. Leyden. 2017. "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. 68

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. 41, 42

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., James Levinsohn, and Ariel Pakes. 1995. "Automobile Prices in Market Equilibrium." *Econometrica* 63 (4):841–890. 13, 15

Boudreau, Kevin J. 2018. "Amateurs." 40

Brecko, Kristina. 2017. "New Features Free of Charge? Using Price to Sort Consumers Among Legacy Software Versions." *Working paper* :1–68. 12

Bresnahan, T, J Orsini, and Pl Yin. 2014. "Platform Choice By Mobile Apps Developers." *NBER working paper* . 14

Bresnahan, Timothy, Xing Li, and Pai-ling Yin. 2016. "Paying Incumbents and Customers to Enter an Industry: Buying Downloads." *Working Paper* 94305. 14, 43

Bresnahan, Timothy F. and M. Trajtenberg. 1995. "General purpose technologies 'Engines of growth'?" *Journal of Econometrics* 65 (1):83–108. 9

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. 42

———. 2010. "The Longer Tail: The Changing Shape of Amazon's Sales Distribution Curve." *Social Science Research Network* (September):1–13. 45

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. 42, 43

Coase, R. H. 1972. "Durability and Monopoly." *Journal of Law and Economics* 15 (1):143–149. 10

Comino, Stefano, Fabio Maria Manenti, and Franco Mariuzzo. 2016. "Updates Management in Mobile Applications : iTunes vs Google Play." (May):1–33. 15, 76

ComScore. 2016. "U.S. Smartphone Subscriber Market Share." URL https://www.comscore.com/Insights/Rankings/comScore-Reports-January-2016-US-Smartphone-Subscriber-Market-Share. 17, 18

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. 14

Dogan, Kutsal, Yonghua Ji, Vijay S. Mookerjee, and Suresh Radhakrishnan. 2011. "Managing the versions of a software product under variable and endogenous demand." *Information Systems Research* 22 (1):5–21. 12

Doraszelski, Ulrich, Gregory Lewis, and Ariel Pakes. 2016. "Just Starting Out: Learning and Equilibrium in a New Market." *NBER Working Paper* . 68, 79

Doraszelski, Ulrich and Mark Satterthwaite. 2010. "Computable Markov-perfect industry dynamics." *RAND Journal of Economics* 41 (2):215–243. 34

Draganska, Michaela, Michael Mazzeo, and Katja Seim. 2009. "Beyond plain vanilla: Modeling joint product assortment and pricing decisions." *Quantitative Marketing and Economics* 7 (2):105–146. 13

Eizenberg, Alon. 2014. "Upstream innovation and product variety in the U.S. home PC market." *Review of Economic Studies* 81 (3):1003–1045. 13

Ellison, Glenn and Drew Fudenberg. 2000. "The Neo-Luddite's Lament: Excessive Upgrades in the Software Industry." *The RAND Journal of Economics* 31 (2):253. URL http://doi.wiley.com/10.2307/2601040. 12

Ericson, Richard and Ariel Pakes. 1995. "Markov-perfect industry dynamics: A framework for empirical work." *The Review of Economic Studies* 62 (1):53–82. 32

Ershov, Daniel. 2017. "The Effect of Consumer Search Costs on Entry and Quality in the Mobile App Market." 14, 42, 44, 45

Evans, David S. 2011. "The antitrust economics of free." *Competition Policy International* 7 (1):70–89. 53

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, 76

Gandhi, Amit, Zhentong Lu, and Xiaoxia Shi. 2013. "Estimating Demand for Differentiated Products with Error in Market Shares." 40

Gans, Joshua S. 2012. "Mobile Application Pricing." :1–24. 14

Garg, R and Rahul Telang. 2012. "Estimating App Demand from Publicly Available Data." *Available at SSRN 1924044* 37 (4):1–22. 42, 44

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. 15, 42, 44, 55, 76

Goettler, Ronald L and Brett R Gordon. 2011. "Does AMD Spur Intel to Innovate More?" *Journal of Political Economy* 119 (6):1141–1200. 9, 13

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. 21

Gowrisankaran, Gautam and Marc Rysman. 2007. "Dynamics of Consumer Demand for New Durable Goods." *Journal of Political Economy* 120 (6):1173–1219. 10, 11

Hoctor, Kevin. 2013. "In-App Purchase - The Future is Here." URL http://blog.hoctor.com/in-app-purchase-the-future-is-here/. 27

Jovanovic, Boyan and Peter L. Rousseau. 2005. "Chapter 18 General Purpose Technologies." In *Handbook of Economic Growth*, vol. 1. 1181–1224. 9

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. 11

Liu, Yongdong. 2017. "Mobile App Platform Choice." :1–66. 14, 42

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." 14

Nelson, Phillip. 1970. "Information and Consumer Behavior." *Journal of Political Economy* 78 (2):311–329. 27

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=. 53, 54

Perry, Charles. 2015. "The Shape of the App Store." URL dazeend.org/2015/01/the-shape-of-the-app-store/28/. 42

Quan, Thomas W. and Kevin R. Williams. 2015. "Product variety, across market demand heterogeneity and the value of online retail." *Working Paper* November. 41

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. 57

Scherer, F. M. 1967. "Market Structure and the Employment of Scientists and Engineers." *The American Economic Review* 57 (3):524–531. 9

Schumpeter, Joseph a. 1947. "The Process of Creative Destruction." In *Capitalism, Socialism and Democracy*. 8

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/. 21

Strafach, Will. 2017. "Advisory: AccuWeather iOS app sends location information to data monetization firm." URL https://hackernoon.com/advisory-accuweather-ios-app-sends-location-information-to-data-monetization-firm-83327c6a4870. 31

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. 13, 14, 34

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. 14, 15

Zhou, Yiyi. 2014. "Failure to Launch in Two-Sided Markets: A Study of the U.S. Video Game Market." (February). 11