Leyden, Benjamin T.

Working Paper

Platform Design and Innovation Incentives: Evidence from the Product Ratings System on Apple's App Store

CESifo Working Paper, No. 9113

Provided in Cooperation with:
Ifo Institute – Leibniz Institute for Economic Research at the University of Munich

Suggested Citation: Leyden, Benjamin T. (2021) : Platform Design and Innovation Incentives: Evidence from the Product Ratings System on Apple's App Store, CESifo Working Paper, No. 9113, Center for Economic Studies and Ifo Institute (CESifo), Munich

This Version is available at:
https://hdl.handle.net/10419/236655

CESifo WORKING PAPERS

9113
2021

May 2021

Platform Design and Innovation Incentives: Evidence from the Product Ratings System on Apple's App Store

Benjamin T. Leyden

CESifo

Impressum:

CESifo Working Papers
ISSN 2364-1428 (electronic version)
Publisher and distributor: Munich Society for the Promotion of Economic Research - CESifo GmbH
The international platform of Ludwigs-Maximilians University's Center for Economic Studies and the ifo Institute
Poschingerstr. 5, 81679 Munich, Germany
Telephone +49 (0)89 2180-2740, Telefax +49 (0)89 2180-17845, email office@cesifo.de
Editor: Clemens Fuest
https://www.cesifo.org/en/wp
An electronic version of the paper may be downloaded
· from the SSRN website: www.SSRN.com
· from the RePEc website: www.RePEc.org
· from the CESifo website: https://www.cesifo.org/en/wp

CESifo *Working Paper No. 9113*

Platform Design and Innovation Incentives: Evidence from the Product Ratings System on Apple's App Store

Abstract

A lack of platform-level competition among digital marketplaces can result in socially inefficient platform design and meaningful welfare losses, even independent of actively anticompetitive behavior. To illustrate the first-order effects platform design can have on competitive outcomes, I investigate how the longstanding design of the product ratings system on Apple's App Store affected innovative behavior by platform participants. I leverage an exogenous change in this system to show that for nearly a decade, the design of the App Store's product ratings system led to less frequent product updating by high-quality products. I provide suggestive evidence that this policy resulted in lost, as opposed to simply delayed, innovation.

Benjamin T. Leyden
Dyson School of Applied Economics and Management
Cornell University / Ithaca / NY / USA
leyden@cornell.edu

May 18, 2021
I am grateful to Gaurab Aryal, Ambre Nicolle, Michael Ward, Georgios Zervas, and seminar participants at the Centre for Competition Policy at the University of East Anglia, the Toulouse School of Economics Online Seminar on the Economics of Platforms, the CESifo Area Conference on the Economics of Digitization, the Toulouse School of Economics 14th Digital Economics Conference, the 12th Paris Conference on Digital Economics, and the 19th Annual International Industrial Organization Conference for helpful comments.

1 Introduction

A common issue in digital marketplaces is the presence of information asymmetries between one side of the platform and another. As Tadelis (2016) notes, "for a marketplace to flourish ... it is necessary that both sides of the market feel comfortable trusting each other, and for that, they need to have safeguards that alleviate the problems caused by asymmetric information." Frequently, platforms attempt to resolve these problems through the implementation of a product ratings system, where participants on one or both sides of the market can rate and comment on the other side. Despite the seeming ubiquity of these systems in modern digital marketplaces, how these systems affect firms' strategic behavior is understudied.

Firms' strategic responses to the design of these systems is an important question for two primary reasons. First, digital platforms are becoming an increasingly important part of the economy. In 2019, Amazon reported over a quarter of a trillion dollars in net sales, and Apple processed over fifty billion dollars in transactions (Amazon.com Inc., 2020; Borck, Caminade, and von Wartburg, 2020).[1] Second, it is not obvious whether a profit-maximizing platform would choose to implement a ratings system that results in the full provision of information (Belleflamme and Peitz (2018)), and to the extent that such a decision might affect consumer demand, it could therefore affect firms' strategic behavior.

Indeed, there has been a significant amount of research in recent years about the optimal aggregation of ratings, and the ratings systems that result from this body of work typically differ from what is currently implemented in digital marketplaces (Dai et al., 2018; Filippas, Horton, and Golden, 2019). Moreover, Lizzeri (1999) has shown in the context of certifying intermediaries that a monopoly platform might choose to provide only enough information to verify that a minimal level of quality is met. And, while Lizzeri shows that competition can have a disciplining effect, leading to full information provision, there's growing concern about a lack of competition among digital platforms. In recent years, there have been a number of reports documenting a lack of competition and various antitrust concerns in digital markets.[2]

In this paper, I show that a longstanding policy governing the product ratings system on

[1] Borck, Caminade, and von Wartburg (2020) report that Apple paid developers approximately 39 billion dollars in 2019, and Apple typically takes a 30% cut of all transactions.

[2] E.g., Cremer, de Montjoye, and Schweitzer (2019); Scott Morton et al. (2019); Furman et al. (2019).

Apple's App Store led developers on the platform to engage in less frequent product innovation. Additionally, I find that developers whose product demand is more dependent on the product ratings system, or that have a harder time building a reputation on the platform, showed signs of a greater discouragement effect. Finally, I provide suggestive evidence that this innovation may have been lost, as opposed to simply delayed.

My analysis centers on a policy on Apple's App Store regarding how product ratings were aggregated. For nearly a decade, the App Store would reset an app's salient average rating each time the app was updated, which, in theory, discouraged product updating from higher quality apps, and potentially encouraged more frequent updating from lower-quality apps. I leverage an exogenous—from the perspective of an app developer—change in this policy to show that the policy indeed led to a suboptimal level of product innovation among higher quality products. While measuring the welfare impact of this policy using the empirical methods employed herein is difficult, I use data on the content of the product updates to provide suggestive evidence that in many, although likely not all cases, this policy resulted in lost, and not just delayed, innovation. These findings provide evidence that the design of a ratings platform can have a first-order effect on intra-platform competitive behavior, and, ultimately, may be socially suboptimal.

To date, there is relatively limited research on how firms respond to the design of ratings systems. Davis and Chhabra (2016) provide evidence that multi-product developers reallocate effort across products based on the feedback they receive via user ratings, and Klein, Lambertz, and Stahl (2016) show that when eBay improved the quality of its review system, there was no change in the composition of sellers on the platform, and that seller effort likely increased following the design change. More recently, Hunter (2020) shows that firms engage in short-run gaming of ratings in a context where average ratings are rounded to the nearest half-star.[3]

Beyond rating and review systems, there is growing evidence about the role a platform's design can have on competitive outcomes. Claussen, Kretschmer, and Mayrhofer (2013) study a policy change on Facebook that made the number of notifications an app on the platform could send a function of how compelling (proxied by user engagement) the app's past notifications were. Meant to discourage spam notifications, Claussen, Kretschmer, and Mayrhofer provide evidence that this

[3]There is also evidence that the design of a rating system can affect demand-side behavior. Vana and Lambrecht (2020) show that the ordering of reviews, can affect the likelihood of a consumer purchasing a product.

policy change led to an increase in the average quality of an app on Facebook. In the context of Google's mobile app store, the Google Play Store, Ershov (2020) shows that the introduction of new product categories lowered search costs, and resulted in increased entry. He finds that the change to the platform's organizational structure led to welfare increases, although those effects are mitigated by both a decline in the quality of the marginal entrant, and an increase in congestion on the platform.

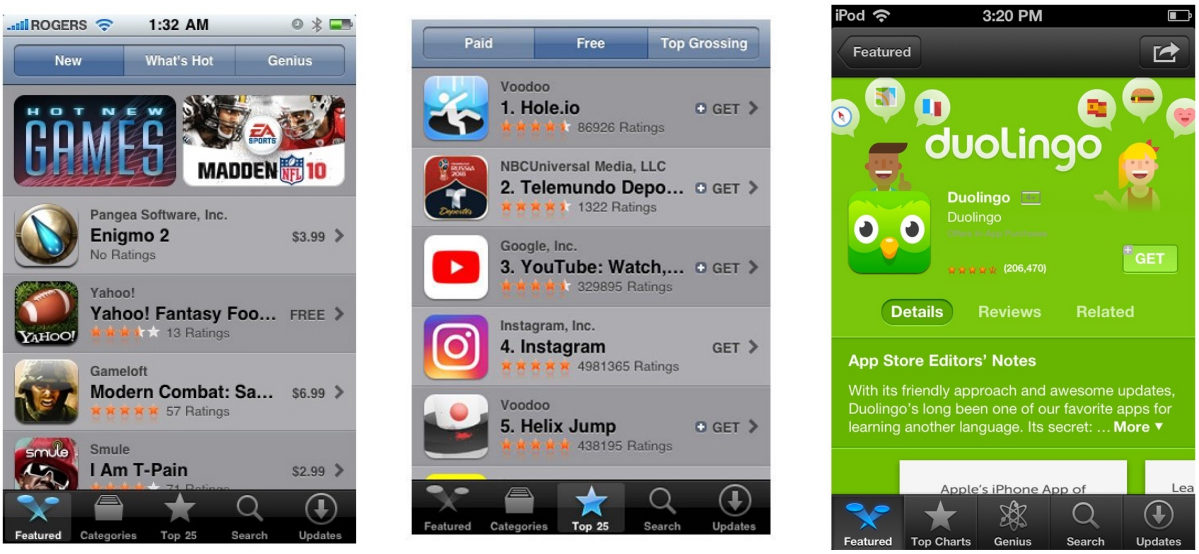
Finally, this paper contributes to these areas of research by showing how a specific platform design decision—how product ratings are calculated and displayed—can distort the *innovative* behavior of the firms competing on the platform. This builds on two recent papers that suggest platform design can have a meaningful impact on innovation by platform participants. Using data on Apple's App Store and the Google Play Store, Comino, Manenti, and Mariuzzo (2018) argue that the level of quality control by a platform can affect the returns to product updating, and therefore affect the incentive to engage in product innovation. Leyden (2019) shows that new technologies provided to developers by Apple's mobile platform, namely the ability to push automatic updates to consumers, as well as to monetize product use via ads, add-on purchases, and other means, leads to both increased and more substantial product innovation.

2 The App Store & Policy Change

Introduced in 2008, Apple's App Store is the exclusive marketplace for buying and selling software for Apple's mobile products, which encompass the iPhone, iPad, Apple Watch, and Apple TV product lines. In most cases, developers can join the platform for $99 per year, and membership on the platform allows a developer to sell as many different apps as they would like. Apple collects additional revenue from developers via a so-called "tax" on revenues earned on the platform.

Apple maintains a product review system in order to better inform potential customers about the quality of an app. Customers have the ability to rate and/or review apps they have previously purchased. Users rate apps on an integer, 1-5 star scale. All text reviews are also accompanied by a rating on this scale. An app's average rating is displayed, in half-star increments, to users in search results, app ranking lists, and on the app's individual store page. Fig. 1 provides examples of these displays. In cases where an app has not yet received at least five ratings the phrase "No

Figure 1: Display of App Ratings



(a) App Store Homepage

(b) "Free" Ranking List

(c) App Storefront

Source: Hackett (2018)

Ratings" is displayed. See the entry for the app Enigmo 2 in Fig. 1a for an example of this.

When the App Store debuted in July 2008, the salient user ratings for an app were those for the app's *current* version. Since these averages are constructed from ratings for the currently available version, any update to the app by the developer automatically resets the score to the "No Ratings" display. This ratings reset policy received significant criticism from app developers, who felt the policy penalized high quality work by imposing an additional, reputational cost of updating.

In July, 2017, Apple announced plans to change this policy, with the executive in charge of the App Store explaining,

> "... some developers don't like submitting their updates, because it resets the rating. So they would get upset saying, 'Oh, man, I have a choice, fix some bugs and blow away my ratings or keep my high rating. I have a 4.7, I don't want to submit it.' And I thought that was kind of stupid." — Philip Schiller, Apple SVP of Worldwide Marketing (Gruber, 2017)

Apple's new policy, officially enacted on September 19, 2017, with the release of Apple's iOS 11 operating system, gave developers the option of either reseting their average rating with a product update—consistent with the original policy—or keeping their existing average rating.

3 The Effect of the Ratings Policy on App Updating

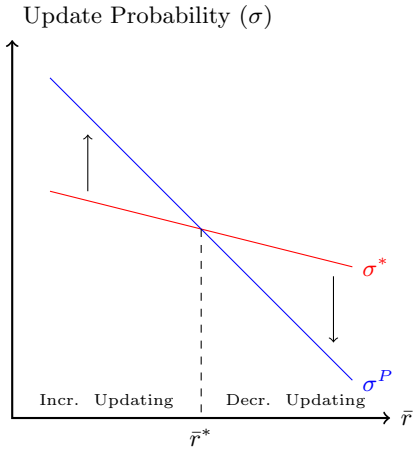
In this section, I outline the expected impact of Apple's review resetting policy (henceforth, reset policy) on app developers' product updating decisions. Consider a developer producing an app for a mobile software platform, with the developer and their app indexed by j. Developers maximize their discounted profit stream, $\sum_t \beta^t \pi_{j,t}(R_{j,t}, F_{j,t})$, by choosing each period whether to update their app. A developer's per-period profits $\pi_{j,t}(R_{j,t}, F_{j,t})$ depend on the revenue the app earns in period t, $R_{j,t}$, and the fixed cost of producing an app update, should the developer decide to update their product, $F_{j,t}$. I assume that updates increase demand, and therefore the developer's revenue, which is consistent with prior empirical work (see Ghose and Han (2014), Comino, Manenti, and Mariuzzo (2018), and Leyden (2019)). Each period, developers weigh the benefits associated with an update (increased revenue in the current and, possibly, future periods) against its cost. With this in mind, we can consider a developer's optimal updating policy, which I represent by the probability of updating in a given period, $\sigma^*_{j,t}$.

Next, consider a ratings system on this platform where users (i.e., past purchasers) have the option of rating the app. The existence of a ratings system slightly modifies the developer's original optimization problem. To understand how, first note that a number of studies have shown shown that ratings affect demand (see, e.g., Jin and Leslie (2003); Chevalier and Mayzlin (2006); Luca (2016); Luca and Zervas (2016); Jabr et al. (2020)).[4] Furthermore, if ratings reflect (at least on average) the quality of the product, an app update will have both a direct, positive effect on profits, and an indirect, positive effect on profits via its effect on the product's future average rating. Thus, the firm's updating decision may depend on the value of its average rating, $\bar{r}_{j,t}$ at the time of the update decision. In the interest of simplifying exposition, assume without loss of generality, that that this relationship is downward sloping ($\frac{\partial \sigma^*}{\partial \bar{r}} < 0$). That is, suppose that apps with a higher average rating are less likely to update.

To understand how Apple's original policy of resetting an app's rating after an update affects a developer's product updating decision, we can focus our analysis on how the policy might affect the slope of the firm's updating policy function σ with respect to $\bar{r}$ (i.e., $\frac{\partial \sigma}{\partial \bar{r}}$). If a policy is enacted that resets ratings with each product update, then when an app updates it will lose whatever amount

[4] Consistent with the mechanism developed in this section, Jabr et al. (2020) provide evidence that low-reputation (high-reputation) apps on the App Store saw their demand increase (decrease) following update-induced ratings resets.

Figure 2: Effect of Review Reset Policy on Updating Decisions



A policy that resets product ratings with each app update will distort developers' updating behavior from their optimal updating policy, σ^*, to σ^P, which reflects the encouraging effect of this policy to apps with an average rating $\bar{r} < \bar{r}^*$, and the discouraging effect for apps with an average rating $\bar{r} > \bar{r}^*$. The new policy, enacted in September 2017, corrects the distortion at the high end ($\bar{r} > \bar{r}^*$), while preserving it on the low end ($\bar{r} < \bar{r}^*$). I.e., following the policy change, a developer's update probability curve will be upper envelope of the two lines.

of demand was due to its prior average rating. For example, if a 5-star average rating causes a 10% increase in demand (relative to a baseline of no rating), then the first-order effect of resetting ratings will be to lose that 10% boost in demand. Of course, the update will still increase the quality of the product (at least in expectation), therefore increasing demand (and future average ratings), and so many developers will still update. Overall, though, if ratings have a positive effect on demand, then developers of highly rated products will become less likely to update, all else equal, in the presence of the reputational penalty imposed by this policy. Moreover, developers' responses to this policy will vary with the degree to which the demand for their product is affected by the penalty.

The effect of this policy on developers of lower-rated apps is less clear, and ultimately depends on the consumer response to a product with no ratings. There are two possible cases: One possibility is that consumers will treat a product with no ratings as if it has an average rating of 0. If this is the case, then we can expect the reset policy to result in a reduction in updating across all products. The second possibility is that consumers view especially low ratings, say a 1-star average, as a stronger signal of low quality than no ratings. In this case, the policy would actually encourage increased updating by low quality apps, as the ratings reset would boost demand for the product (at least until new ratings/reviews arrive). If we define $\bar{r}^*$ to be the average rating that consumers

view as equivalent to having no ratings, then we can expect that Apple's reset policy will *discourage* updating for apps rated $\bar{r}_{j,t} > \bar{r}^*$, and *encourage* updating for apps rated $\bar{r}_{j,t} < \bar{r}^*$, relative to the optimal updating frequency σ^*.[5]

As discussed in Section 2, the policy change that was enacted in 2017 with the release of iOS 11 gave developers the *option* of keeping or reseting their ratings with each update. This preserves any potential benefits of the old policy, which would accrue to apps rated $\bar{r}_{j,t} < \bar{r}^*$, but eliminates any discouraging effect for apps rated $\bar{r}_{j,t} > \bar{r}^*$ (assuming developers optimally select when to reset their ratings). Therefore, we should expect to see the update response function flatten with respect to $\bar{r}$ after the policy change for highly rated apps ($\bar{r} > \bar{r}^*$), as higher rated apps become relatively more likely to update, but remain steeper than σ^* for lower rated apps ($\bar{r} < \bar{r}^*$).

4 Data

To better understand Apple's ratings reset policy affected product innovation, I collected weekly app characteristic and review data from Apple's App Store marketplace beginning in January 14, 2015 and ending in December 25, 2019. The policy change took place on September 19, 2017. Each review in the dataset includes a numerical rating, on a 1-5 star scale, which I use to construct an average rating for each app in each week. This sample consists of all apps from the Education, Productivity, and Utilities categories. The Education category encompasses all apps that serve an educational purpose, such as apps that help someone learn a second language. Productivity apps include task-management, calendar, note taking, and similar apps. The Utilities category covers a more diverse set of products, such as calculators, and privacy-focused web browsers.

A primary challenge in defining a sample for this analysis is that many of the apps on the App Store are not actively engaged in competition. This can be the case for two reasons. First, the cost of keeping a product on the platform is low, and as a result a large portion of the apps on the store can be considered "abandoned."[6] Second, a combination of the low cost of entry and the relatively low cost of product development can attract a large number of hobbyists (Boudreau,

[5] Note that the previous case, where the "No Ratings" display is viewed as a rating of zero, is a special case of this approach, with $\bar{r}^* = 0$.

[6] The typical cost to develop on Apple's platforms is $99 per person per year. This annual cost gives an individual the ability to maintain as many products on the platform as they would like. So, if a developer abandons the development of one app in order to focus on another, there's no cost to keeping the original product on the store.

Table 1: Update Classification Precision and Recall

	Precision	Recall	N
Bug Fix	0.86	0.94	180
Feature	0.77	0.57	65
Weighted Average/Total	0.83	0.84	245

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.

2018), which can result in products that are available on the platform, and are possibly being updated regularly, but for which the innovative decisions of the developer are motivated by reasons other than profit-maximization.

In order to identify the set of apps that are competitively active on the platform, I use the fact that Apple posts three category-specific, daily product ranking lists. These lists rank 1) free apps based on unit sales, 2) paid (i.e., not free) apps based on unit sales, and 3) all apps based on gross revenue. I construct my sample using all apps that 1) have an average ranking of at least 250^{th} on one or more of these lists during the pre period, and 2) average at least one user review per month.

In addition to studying the product updating decision by developers, I also consider the content of those updates. I classify the content of updates in two ways. The first uses an app's version number as a proxy for the content of the update. The second approach analyzes the app developer's notes about the product update to infer the content of the update.

Version Number (VN) Classification When a developer updates their app, they increment the product's "version number." The manner in which these numbers change provide a proxy for the content of the update. Traditionally, changes to the number before the first decimal of a version number (e.g., from 12.1 to 13.0) indicates a "Major' revision of the product, while changes of any of the subsequent numbers (e.g., 12.1 to 12.2) indicates a "Minor" update. Notably, this measure will be a noisy proxy of update content because the App Store does not enforce a formal versioning nomenclature, and while the approach described above is historically common, developers are free to use alternative approaches.[7]

[7]E.g., a new approach that is gaining popularity on the iOS platform is setting version numbers using the year and a incrementing count. E.g., 2020.1, 2020.2, etc.

Table 2: Percentage of Updates, by Type

	Minor	Major	Total
Bug Fix	67.80	3.30	71.10
Feature	25.60	3.30	28.90
Total	93.40	6.60	100.00

Cross-tabulation showing the percentage of updates under the Version Number (columns) and Release Notes (rows) classification systems.

Release Notes (RN) Classification In addition to changing an app's version number, developers also provide a written set of "release notes" with each update, documenting how the update changes the product. To do so, I employ the approach developed in Leyden (2019) to classify the content of updates using these documents. This method first uses natural language processing techniques to analyze these documents, and then uses a support vector machine (SVM), a supervised machine learning technique, to classify the content of updates based on the content of their release notes. In particular, I classify app updates as either "Feature" updates, which add new features or functionality to the product, or as "Bug Fix" updates, which correct software bugs, or make other incremental changes. While Major and Feature (or Minor and Bug Fix) updates can be viewed as somewhat analogous, throughout the rest of this paper I maintain the distinct vocabularies to help differentiate the two approaches.

In order to conduct this classification, I hand coded 782 release notes as either feature or bug fix updates. This training set was used to train the SVM, which is then used to predict the content of each update. Table 1 provides details of the precision and recall of the SVM. Table 2 provides a cross-tabulation of update types using both systems. Under both approaches, the less substantial type of update (Minor/Bug Fix) are more common than the more substantial updates (Major/Feature). Importantly, though, the overlap is not perfect—over a quarter of the Minor updates, under the VN classification approach, appear to have added additional features or functionality to the product.

Table 3 provides summary statistics for the key variables used in this analysis. Productivity apps account for nearly 36% of the sample, followed by Utility and then Education apps. The average rating (3.676 for the full sample) is relatively similar across categories. Fig. 3 shows the frequency of updates across the entire sample both in aggregate, and broken down by update type. Fig. 3a shows a relatively stable level of updating over time, though there remains a lot of

Table 3: Summary Statistics

	Update		No Ratings		Average Rating ($\bar{r} > 0$)		Price		Download Size (MB)		
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	N
Category											
Education	0.130	0.336	0.347	0.476	3.804	0.993	1.999	13.471	110.074	149.151	77,741
Productivity	0.126	0.332	0.254	0.435	3.728	0.965	2.837	7.201	58.131	69.888	89,969
Utilities	0.088	0.283	0.255	0.436	3.520	1.059	1.775	4.892	42.096	46.586	85,129
Time Period											
Pre-iOS 11	0.111	0.314	0.287	0.452	3.833	0.982	2.264	9.195	55.130	92.374	138,036
Post-iOS 11	0.119	0.323	0.278	0.448	3.491	1.019	2.171	8.949	85.022	107.356	114,803
Total	0.115	0.318	0.283	0.450	3.676	1.013	2.222	9.084	68.703	100.564	252,839

week-to-week variation across the entire sample.

5 Empirical Analysis

5.1 Overall Response

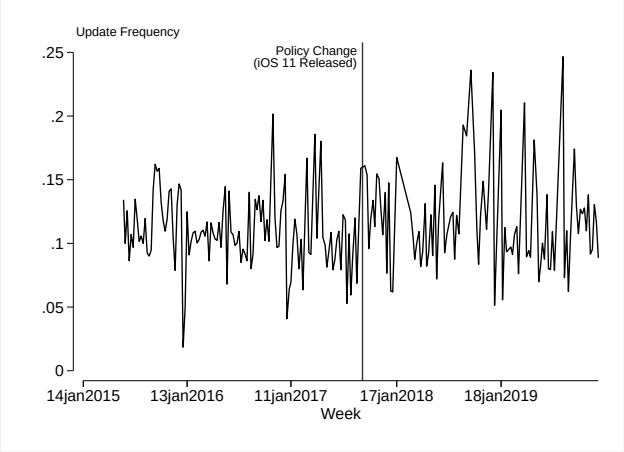
In order to understand how Apple's initial review resetting policy affected developers' innovative behavior, I estimate the following linear probability model using OLS,

$$\begin{aligned}\mathbb{1}(update)_{j,t} = {} & \beta_0 \bar{r}_{j,t} + \beta_1 \bar{r}_{j,t} \times \mathbb{1}(\text{Post-iOS 11})_{j,t} \\ & + \beta_2 \mathbb{1}(\text{No Rating})_{j,t} + \beta_3 \mathbb{1}(\text{No Rating})_{j,t} \times \mathbb{1}(\text{Post-iOS 11})_{j,t} \\ & + \beta_4 p_{j,t} + \beta_5 n_{j,t} + \beta_6 \text{version-age}_{j,t} + \beta_7 \text{size}_{j,t} + \mu_t + \mu_j + \epsilon_{j,t},\end{aligned} \tag{1}$$

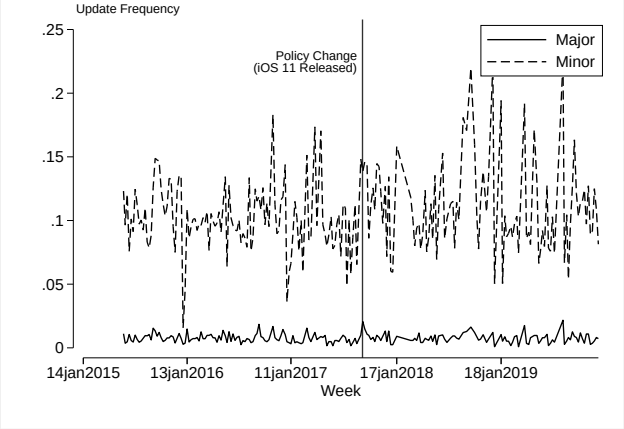
where, $\mathbb{1}(update)_{j,t}$ is a binary indicator for whether app j was updated in week t, and $\bar{r}_{j,t}$ is the average rating of the app at the start of period t (i.e., prior to an update, if one occurs). $\mathbb{1}(\text{Post-iOS 11})_{j,t}$ indicates whether period t is before or after the policy change, which occurred with the September 19, 2017 release of Apple's iOS 11 operating system. $\mathbb{1}(\text{No Rating})_{j,t}$ indicates whether an app is currently displaying the "No Ratings" label, which is applied to an app version with fewer than five ratings. β_1 is the coefficient of interest, and represents the extent to which the estimated relationship between an app's rating and the developer's updating decision changes following the removal of the reset policy.

In order to control for other relevant factors that might affect a developer's updating decision, I include the app's price ($p_{j,t}$), the number of reviews for the current version of the app ($n_{j,t}$), how

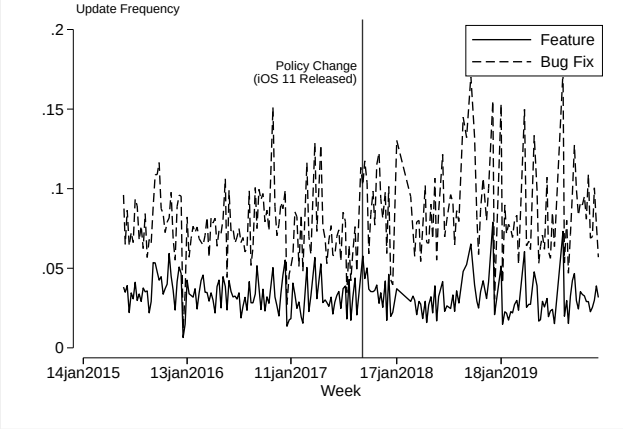
Figure 3: Sample Updating Frequency



(a) Overall Updating



(b) Updating by Type (VN)



(c) Updating by Type (RN)

Table 4: The Effect of Apple's Review Reset Policy on Updating Behavior

	(1) All	(2) Education	(3) Productivity	(4) Utilities
Avg Review Score	-0.0134***	-0.0078**	-0.0170***	-0.0149***
	(0.0019)	(0.0035)	(0.0036)	(0.0029)
Avg Review Score X Post iOS 11	0.0076***	0.0083**	0.0093**	0.0060**
	(0.0021)	(0.0041)	(0.0040)	(0.0029)
No Ratings	-0.0290***	-0.0215	-0.0351**	-0.0241**
	(0.0077)	(0.0145)	(0.0148)	(0.0109)
No Ratings X Post iOS 11	0.0430***	0.0642***	0.0413**	0.0264**
	(0.0092)	(0.0185)	(0.0169)	(0.0131)
N	252,839	77,741	89,969	85,129

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

long the current version of the app has been on the App Store (version-age$_{j,t}$), and the size of the app measured in megabytes (size$_{j,t}$). Additionally, I include week and app fixed effects, μ_t and μ_j.

Table 4 shows the results of estimating Eq. (1) for the full sample and individually for each category.[8] In all cases, I find evidence of a negative relationship between an app's rating and its likelihood of updating. However, I find a positive effect of the removal of Apple's resetting policy on this relationship (β_1). The relationship between a product's rating and its likelihood of updating increases from -0.0134 to -0.0058 after the policy change, which still indicates a statistically significant, negative relationship ($p < 0.01$).

I find that the effect of the policy change on the relationship between an app's average rating and the decision to update varies by category, with the largest effects in the Productivity and Utilities categories, and a smaller change in the Education category. These estimates for β_1 are consistent with the idea that Apple's original policy of resetting product ratings when an app was updated decreased the rate of updating among higher-quality apps, as discussed in Section 2. In light of these results, it seems likely that Apple's reset policy led developers of lower-rated apps to update more frequently than they otherwise would. However, because Apple's policy change in 2017 provided developers with the option of reseting their rating with each update, I an unable to

[8] In Section 6 I provide versions of this table under alternative sample definitions, and excluding the dates between the announcement and enactment of the policy change.

directly test this.

5.2 Response Heterogeneity

Having shown that on average, the ratings reset policy decreased the frequency of product updates, I now consider whether and how this response varied by developer. In particular, the extent to which an app's frequency of innovation was affected by the policy likely depends on the degree to which the app faced a meaningful reputational penalty from losing its past rating. The severity of this penalty will vary along the size and length of the penalty.

The size of the penalty reflects how dependent a given app is on its rating to attract new customers. For example, it's unlikely that sales of the Microsoft Word app, a dominant word processing application, was heavily dependent on its current App Store rating, while Ulysses, a more niche word processing app, may depend heavily on having a good rating to attract new customers. As a result, the negative demand shock associated with an update-induced ratings reset should be much larger for Ulysses than for Microsoft Word.

The length of the penalty reflects the fact that once an app's rating has been reset, the app is dependent on users rating the app in order to rebuild its reputation. For some apps, this may take only a few minutes or hours, while for others it could take days or weeks until a new rating is displayed. For apps in the former group, it's unlikely that the reset policy would significantly affect their innovative behavior, because the cost would be so short lived.

In order to investigate whether apps with particularly high demand, or those with the highest arrival rate of reviews are less distorted by the initial ratings reset policy, I split out the top quartile of the sample according to the app's average download ranking and according to the app's average number of new reviews per week. Within each subsample, I re-estimate Eq. (1). I present the results of these regressions in Table 5.

In columns (1) and (2) of Table 5 I present the results of the subsample regressions when the sample is split according to the apps' download ranking. I find no evidence of a response to the policy change among top-performing apps—the response appears to come entirely from the lower-demand apps. This finding is consistent with the expectation that apps that rely most on ratings, which should typically be those with lower demand, face a larger reputational penalty are most responsive to the policy change.

Table 5: Response Heterogeneity with the Size and Length of the Reputational Penalty

	Penalty Size		Penalty Length	
	(1)	(2)	(3)	(4)
	Top 25%	Bottom 75%	Top 25%	Bottom 75%
Avg Review Score	-0.0133***	-0.0134***	-0.0157***	-0.0121***
	(0.0041)	(0.0021)	(0.0049)	(0.0020)
Avg Review Score X Post iOS 11	0.0021	0.0095***	0.0028	0.0091***
	(0.0053)	(0.0021)	(0.0059)	(0.0020)
No Ratings	-0.0371**	-0.0269***	-0.0321	-0.0247***
	(0.0170)	(0.0086)	(0.0225)	(0.0080)
No Ratings X Post iOS 11	0.0331	0.0490***	0.0512*	0.0526***
	(0.0225)	(0.0095)	(0.0282)	(0.0090)
N	63,267	189,572	63,412	189,427

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

In columns (3) and (4), I split the sample based on the arrival rate of new reviews. Here again I find that the response appears to come entirely from the bottom subsample, which is consistent with the expectation that apps that are likely to face the reputational penalty over a longer period of time will be most responsive to the policy change.[9]

Taken together, the results in this subsample analysis further support the view that Apple's policy created a real reputational penalty that affected developers' innovative behavior on the platform, as those apps that faced a larger, or more long-lasting reputational penalty were most responsive to the 2017 policy change.

5.3 Content of Product Updates

I have shown that the original policy led to a decrease in the frequency of product updates. This result can be explained both by lost innovation—i.e., product updates that would have occurred but for the policy—or it could be the case that while developers updated their products less frequently when the reset policy was in place, the total amount of innovation remained the same, and was just distributed in less frequent, more aggregated releases. While this option would still result in

[9] In **??**, I consider an alternative subsample split and separate out the top decile. I find similar results to those presented here.

a policy-induced welfare loss—if an update is delayed for a week, the value that would have been enjoyed by consumers over the course of that week cannot be recovered—it would be a smaller loss than if the innovation were lost entirely.

If the ratings reset policy resulted in delayed innovation, then we should expect to see an increase in the relative likelihood of Minor/Bug Fix updates in the post period. This is because, in the presence of the reset policy, Minor/Bug Fix updates impose the same repetitional penalty on developers as Major/Feature updates, despite being of lesser importance. If, instead, the ratings reset policy resulted in lost innovation, the effect on the relative frequency of Minor/Bug Fix updates in the post period is theoretically ambiguous, as it would depend on the nature of the lost innovation.

Given this, I investigate whether updates were lost or simply delayed by estimating a variation of Eq. (1) on the set of observations where the developer chose to update, using an indicator for a Minor (as opposed to Major) or Bug Fix (as opposed to Feature) updates as the left-hand side variable. As before, the coefficient of interest is β_1, which indicates how the observed relationship between an app's average rating and the likelihood of creating a Minor/Bug Fix update conditional on updating changes after the ratings reset policy is changed. If innovation is simply delayed, then estimates of β_1 should be positive, indicating an increase in the relative frequency of Minor/Bug Fix updates in the post period. Otherwise, the data is consistent with the view that innovation was lost as a result of the ratings reset policy.

Table 6 shows the results of estimating this model.[10] Columns 1-4 of Table 6 display the results of estimating this model using the Version Number (VN) classification approach (Major/Minor updates), and columns 5-8 show the results of estimating the model using the Release Notes (RN) classification approach (Feature/Bug Fix updates).

Under the VN approach, I fail to find statistically significant evidence of a positive relationship between the removal of the reset policy and the content of updates when looking across all categories (column 1). This provides suggestive evidence that the initial policy resulted in lost innovation, and not just delayed innovation, as discussed above. This finding is supported when looking at each category separately, in columns 2-4.

[10] In Section 6 I provide versions of this table under alternative sample definitions, and excluding the dates between the announcement and enactment of the policy change.

Table 6: The Effect of Apple's Review Reset Policy on Update Content

	Version Number				Support Vector Machine			
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	All	Education	Productivity	Utilities	All	Education	Productivity	Utilities
Avg Review Score	-0.0086***	-0.0171***	-0.0093**	0.0008	-0.0240***	-0.0400***	-0.0118	-0.0184
	(0.0031)	(0.0061)	(0.0045)	(0.0063)	(0.0067)	(0.0115)	(0.0110)	(0.0123)
Post iOS 11 X Avg Review Score	0.0032	0.0137	0.0051	-0.0133	0.0182**	0.0254	0.0070	0.0090
	(0.0049)	(0.0089)	(0.0072)	(0.0090)	(0.0092)	(0.0158)	(0.0150)	(0.0168)
No Ratings	-0.0191	-0.0579**	-0.0116	0.0162	-0.0647**	-0.1186**	-0.0263	-0.0328
	(0.0129)	(0.0251)	(0.0187)	(0.0250)	(0.0264)	(0.0470)	(0.0426)	(0.0471)
Post iOS 11 X No Ratings	0.0153	0.0507	0.0194	-0.0396	0.0611*	0.0623	0.0141	0.0576
	(0.0193)	(0.0359)	(0.0299)	(0.0329)	(0.0360)	(0.0643)	(0.0599)	(0.0617)
N	28,909	10,098	11,353	7,458	28,909	10,098	11,353	7,458

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

The overall results using the RN approach (column 5) show that Apple's initial policy resulted in less frequent Bug Fix updates than developers would produce absent the policy. However, the magnitude of the estimates in columns 6-8 suggest that this effect may be driven by developers in the Education category, as the coefficients on the $\bar{r}_{j,t} \times \mathbb{1}$(Post-iOS 11) interaction for the Productivity and Utilities categories are both much smaller in magnitude and have much larger relative standard errors. The finding of an increase in the relative frequency of Bug Fix to Feature updates is consistent with *both* the lost innovation theory, and the theory that developers engaged in a practice of delaying updates in order to bundle small and large revisions in less frequent releases. Ultimately, these results across both categories provide suggestive evidence that Apple's initial review policy may have led to lost innovation, instead of simply delayed innovation. However, there is evidence that this effect may not have been uniform across categories.

6 Conclusion

In this paper, I investigate a policy on Apple's App Store that had the potential to discourage product updating for higher quality apps. I find that developers responded to the incentives created by this longstanding policy by reducing the frequency of product updates under the initial product ratings system on the App Store. I have provided additional evidence that developers who are more dependent on the ratings system to drive demand, or who have a harder time acquiring ratings and reviews, were more responsive to the discouragement effects of the policy. Finally, I have provided preliminary evidence that this policy led to lost innovation, although in some cases developers may

have simply delayed innovation in response to the policy by choosing to bundle together large and small product updates.

As the prominence of digital platforms increases, it is important to understand how the design of these centralized markets can affect intra-platform competition and innovation, and, ultimately, welfare. In particular, socially inefficient platform policies can persist given the relative dearth of platform-level competition. My finding of (likely) lost product innovation provides empirical evidence of how the design of a platform can have a first-order effect on competitive behavior, and, ultimately, lead to a reduction of consumer welfare. Notably, even delayed innovation would result in such a welfare loss, though this reduction would be smaller than had the product innovation been lost entirely.

That said, there are a number of ways in which these welfare losses could be mitigated. For example, the preceding analysis does not account for endogenous changes in product monetization, or effects on entry and exit, all of which might offset the welfare effects of lost product updates. Indeed, Vellodi (2021) shows that suppressing the ratings of top-rated firms could lead to increased entry and, consequently, improve consumer welfare. While mechanically distinct, Apple's initial reputation system is in many ways akin to the upper certification policy Vellodi considers. Ultimately, there remains substantial room for future research on the topic of how ratings systems, and platform design decisions more generally, affect consumer and overall welfare.

References

Amazon.com Inc. 2020. "Form 10-K for the Fiscal Year Ended December 31, 2019." URL https://ir.aboutamazon.com/sec-filings/. 2

Belleflamme, Paul and Martin Peitz. 2018. "Inside the engine room of digital platforms: Reviews, ratings, and recommendations." URL https://halshs.archives-ouvertes.fr/halshs-01714549/document. 2

Borck, Jonathan, Juliette Caminade, and Markus von Wartburg. 2020. "How Large is the Apple App Store Ecosystem? A Global Perspective for 2019." . 2

Boudreau, Kevin J. 2018. "Amateurs Crowds & Professional Entrepreneurs as Platform Complementors." *NBER Working paper* URL https://www.nber.org/papers/w24512. 8

Chevalier, Judith A and Dina Mayzlin. 2006. "The Effect of Word of Mouth on Sales: Online Book Reviews." *Journal of Marketing Research* 43:345–354. 6

Claussen, Jörg, Tobias Kretschmer, and Philip Mayrhofer. 2013. "The Effects of Rewarding User Engagement: The Case of Facebook Apps." *Information Systems Research* 24 (1):186–200. URL http://dx.doi.org/10.1287/isre.1120.0467. 3

Comino, Stefano, Fabio M. Manenti, and Franco Mariuzzo. 2018. "Updates Management in Mobile Applications : iTunes vs Google Play." *Journal of Economics and Management Strategy* 28 (3):392–419. 4, 6

Cremer, Jacques, Yves-Alexandre de Montjoye, and Heike Schweitzer. 2019. "Competition Policy for the Digital Era." . 2

Dai, Weijia, Ginger Jin, Jungmin Lee, and Michael Luca. 2018. "Aggregation of consumer ratings: an application to Yelp.com." *Quantitative Marketing and Economics* 16 (3):289–339. URL http://dx.doi.org/10.1007/s11129-017-9194-9. 2

Davis, Jason P. and Pai-ling Chhabra, Yulia Yin. 2016. "Entrepreneurial Diversification and the Scope of New Firms: Multiproduct Innovation in iPhone Apps." . 3

Ershov, Daniel. 2020. "Consumer Product Discovery Costs, Entry, Quality and Congestion in Online Markets." *Working Paper* . 4

Filippas, Apostolos, John Joseph Horton, and Joseph Golden. 2019. "Reputation inflation." *NBER Working Paper Series* WP 25857. 2

Furman, Jason, Diane Coyle, Amelia Fletcher, Derek McAules, and Philip Marsden. 2019. "Unlocking digital competition: Report of the digital competition expert panel." . 2

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

Gruber, John. 2017. "Live from WWDC 2017." *The Talk Show* :Podcast audioURL https://daringfireball.net/thetalkshow/2017/06/08/ep-193. 5

Hackett, Stephen. 2018. "10 Years of App Store: A Timeline of Changes." *MacStories* URL https://www.macstories.net/stories/10-years-of-app-store-a-timeline-of-changes/. 5

Hunter, Megan. 2020. "Chasing Stars: Firms' Strategic Responses to Online Consumer Ratings." *Available at SSRN 3554390* URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3554390. 3

Jabr, Wael, Dominik Gutt, Jurgen Neumann, and Dennis Kundisch. 2020. "The Fate of the App: Economic Implications of Updating under Reputation Resetting." *Working Paper* URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3565937. 6

Jin, Ginger Zhe and Phillip Leslie. 2003. "The effect of information on product quality: Evidence from restaurant hygiene grade cards." *The Quarterly Journal of Economics* 118 (2):409–451. URL https://academic.oup.com/qje/article-pdf/118/2/409/5364183/118-2-409.pdf?casa_token=TocvL4I0IkoAAAAA:bQEGMibNVBq0KVdc71FKkloi1WDKMsTY6HJFN_SsoShMU_Oh9Fd1iqfoz7elC2nSZ-p7u4eh-rb_. 6

Klein, Tobias J, Christian Lambertz, and Konrad O Stahl. 2016. "Market transparency, adverse selection, and moral hazard." *Journal of Political Economy* 124 (6):1677–1713. URL https://www.journals.uchicago.edu/doi/pdf/10.1086/688875. 3

Leyden, Benjamin T. 2019. "There's an App for That: Understanding Product Updating Under Digitization." *Working Paper* . 4, 6, 10

Lizzeri, Alessandro. 1999. "Information Revelation and Certification Intermediaries." *The RAND Journal of Economics* 30 (2):214–231. 2

Luca, Michael. 2016. "Reviews, reputation, and revenue: The case of Yelp. com." *Harvard Business School Working Paper* 12-016. URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1928601. 6

Luca, Michael and G. Zervas. 2016. "Fake It Till You Make It: Reputation, Competition, and Yelp Review Fraud." *Management Science* 62 (12):3393–3672. URL http://businessinnovation.berkeley.edu/WilliamsonSeminar/luca092613.pdf. 6

Scott Morton, Fiona, Pascal Bouvier, Ariel Ezrachi, Bruno Jullien, Roberta Katz, Gene Kimmelman, A Douglas Melamed, and Jamie Morgenstern. 2019. "Committee for the Study of Digital Platforms: Market Structure and Antitrust Subcommittee Report." . 2

Tadelis, Steven. 2016. "Reputation and Feedback Systems in Online Platform Markets." *Annual Review of Economics* 8:321–340. 2

Vana, Prasad and Anja Lambrecht. 2020. "The effect of individual online reviews on purchase likelihood." *Tuck School of Business Working Paper* (3108086). URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3108086. 3

Vellodi, Nikhil. 2021. "Ratings Design and Barriers to Entry." *Working Paper* . 18

Appendices

A Alternative Sample Definitions

In this appendix, I duplicate the results of the paper under alternative sample definitions. In Section A.1, I re-estimate the results of this paper while excluding all observations between the announcement and enactment of the policy change. In Section A.2, I change the average ranking cutoffs used to define the sample (and described in more detail in Section 4).

A.1 Accounting for the Delay Between the Announcement and Implementation of the Policy Change

As discussed in Section 2, while Apple announced its plans to change its rating reset policy in June, 2017, the actual policy change did not occur until September of that year. As a result, it could be the case that developers strategically withheld updates until after the policy change as a result of this early announcement. To address the concern that this behavior may have affected the findings of this paper, I re-estimate Eq. (1) using a sample that excludes all observations from the time of the announcement until the end of September, 2017. Table 7 shows the results of estimating the model with this sample, which are similar to the primary results presented in Table 4. The same is true when reproducing the analysis regarding heterogeneous responses to the policy based on the size and length of the reputational penalty with this reduced sample, as I show in Table 8.

In Table 9, I present the results of re-estimating the app content regressions with this reduced sample. Here too, the results are similar to those reported in Table 6. The one exception is that when estimating the model using the Version Number (VN) classification system and the Education category, I find evidence of a relative increase in Minor (compared to Major) updates after the policy change. While this finding differs from the primary results of this paper, it is important to note that this finding is consistent with both the lost innovation story, and the bundled updates story. Additionally, this result corresponds with the view discussed in the paper that under the Release Notes (RN) classification system, the Education category may have responded differently to the policy change. Overall, these results show that that to the extent that developers withheld updates during this period, this behavior did not bias the results discussed in the paper.

Table 7: The Effect of Apple's Review Reset Policy on Updating Behavior, Dropping Summer 2017

	(1) All	(2) Education	(3) Productivity	(4) Utilities
Avg Review Score	-0.0141***	-0.0088**	-0.0172***	-0.0155***
	(0.0021)	(0.0038)	(0.0037)	(0.0032)
Avg Review Score X Post iOS 11	0.0074***	0.0082*	0.0088**	0.0058*
	(0.0022)	(0.0045)	(0.0041)	(0.0032)
No Ratings	-0.0335***	-0.0271*	-0.0361**	-0.0291**
	(0.0083)	(0.0156)	(0.0154)	(0.0121)
No Ratings X Post iOS 11	0.0442***	0.0661***	0.0390**	0.0283**
	(0.0098)	(0.0198)	(0.0177)	(0.0143)
N	232,227	71,330	82,712	78,185

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 8: Response Heterogeneity with the Size and Length of the Reputational Penalty, Dropping Summer 2017

	Penalty Size		Penalty Length	
	(1) Top 25%	(2) Bottom 75%	(3) Top 25%	(4) Bottom 75%
Avg Review Score	-0.0155***	-0.0137***	-0.0176***	-0.0127***
	(0.0044)	(0.0023)	(0.0052)	(0.0022)
Avg Review Score X Post iOS 11	0.0032	0.0090***	0.0042	0.0088***
	(0.0057)	(0.0023)	(0.0062)	(0.0021)
No Ratings	-0.0498***	-0.0290***	-0.0459*	-0.0279***
	(0.0186)	(0.0092)	(0.0240)	(0.0086)
No Ratings X Post iOS 11	0.0423*	0.0482***	0.0658**	0.0523***
	(0.0245)	(0.0102)	(0.0298)	(0.0096)
N	58,087	174,140	57,944	174,283

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 9: The Effect of Apple's Review Reset Policy on Update Content, Dropping Summer 2017

	Version Number				Support Vector Machine			
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	All	Education	Productivity	Utilities	All	Education	Productivity	Utilities
Avg Review Score	-0.0085**	-0.0185***	-0.0084*	0.0022	-0.0259***	-0.0450***	-0.0130	-0.0160
	(0.0035)	(0.0066)	(0.0050)	(0.0068)	(0.0072)	(0.0122)	(0.0114)	(0.0129)
Post iOS 11 X Avg Review Score	0.0055	0.0162*	0.0076	-0.0112	0.0216**	0.0264	0.0129	0.0094
	(0.0055)	(0.0096)	(0.0088)	(0.0094)	(0.0095)	(0.0166)	(0.0148)	(0.0174)
No Ratings	-0.0146	-0.0542**	-0.0083	0.0221	-0.0730***	-0.1273**	-0.0424	-0.0235
	(0.0143)	(0.0270)	(0.0212)	(0.0268)	(0.0281)	(0.0504)	(0.0448)	(0.0500)
Post iOS 11 X No Ratings	0.0197	0.0527	0.0273	-0.0359	0.0794**	0.0629	0.0475	0.0675
	(0.0218)	(0.0381)	(0.0363)	(0.0340)	(0.0378)	(0.0691)	(0.0598)	(0.0647)
N	26,747	9,268	10,534	6,945	26,747	9,268	10,534	6,945

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

A.2 Results Under Alternative Average Ranking Cutoffs

In the primary analysis presented in the main body of this paper, the sample is restricted to apps in the Education, Productivity, and Utility categories that average a ranking of at least 250^{th} in at least one of the ranking lists provided by the App Store during the pre period. This restriction is made to eliminate abandoned, or otherwise non-competitive apps in the marketplace. In this section, I reproduce all of the tables from the paper under two alternative sample definitions. The first further restricts the sample to apps that rank at least 150^{th} in at least one of the ranking lists. The second loosens the restrictions in the paper, and includes all apps that appear on the ranking lists during the pre period.

The primary results of the paper, presented in Table 4, are recreated under these two sample definitions in Tables 10 and 11. The results regarding developers' responses to the size and length of the reputational penalty, presented in Table 5, are recreated under the alternative sample definitions in Tables 12 and 13. Finally, the results regarding the content of updates, presented in Table 6, are recreated using these alternative samples in Tables 15 and 16

The overall results in column 1 of Tables 10 and 11 are similar to the primary results presented in Table 4. However, the category-level results present some differences. In the more restrictive sample, see Table 10, I find a larger effect in the Education category than before, and no evidence of a response in the Productivity category. This failure to reject the null hypothesis of no response in the Productivity sample is likely related to the fact that this *more* selective sample consists of a much larger proportion of apps that are either not highly dependent on the product ratings

system, or that can quickly rebuild their reputation. As discussed in Section 5.2, such apps appear to be much less responsive to the original ratings policy. In the broader sample, see Table 11, the estimated effects are generally smaller, likely reflecting the fact that this sample definition fails to account for some number of abandoned or otherwise non-competitive apps.

Table 12 presents the results of re-estimating the heterogeneous response regressions with the more restrictive sample. In this case, the original findings of the paper are confirmed—apps that are presumably least dependent on the ratings system, and those with the shortest reputational penalty do not appear to respond to the original ratings system, while those that are more dependent, or take much longer to rebuild their reputation are more responsive. In Table 13, I consider this analysis in the larger, alternative sample. In this case, I do not find support for the theory that the penalty size and/or length determine which apps respond to the policy change. However, this finding is likely mechanical, because expanding the sample by such a large amount results in many apps that depend heavily on the ratings system (or that take a long time to rebuild their reputation) being included in the "top" subsample. Indeed, this intuition in confirmed in Table 14, where I present the results of re-running the analysis with the "All Ranked Apps" sample, but re-defining the top subsample as the top 10% (as opposed to 25%) of apps in each category.

In terms of the effect of the policy on the content of updates, the results under these alternative samples are again similar to the primary results presented in the paper. In the more restrictive sample, see Table 15, I still find no evidence of an effect of the policy on update content under the Version Number (VN) approach in the overall sample. In the Productivity and Utilities categories, I find evidence of a relative increase in Minor (as compared to Major) updates, a finding that is consistent with both the lost and delayed innovation theories. Unlike earlier results, in this sample I find no evidence of an effect of the policy on content under the Release Notes (RN) classification approach, which supports the lost innovation story. In the broader sample, see Table 16, I find no evidence of an effect under the VN approach in the aggregated sample (column 1), but I do find evidence of a response in the Education category. Under the RN approach, the estimated effect appears to be driven by the Education category, consistent with my findings in Section 5.3. Ultimately, this exercise supports the paper's conclusion of suggestive evidence in support of the lost innovation case, though this may vary by product category.

Table 10: The Effect of Apple's Review Reset Policy on Updating Behavior, Average Rank <= 150^{th}

	(1) All	(2) Education	(3) Productivity	(4) Utilities
Avg Review Score	-0.0147***	-0.0097**	-0.0151***	-0.0191***
	(0.0027)	(0.0046)	(0.0053)	(0.0040)
Avg Review Score X Post iOS 11	0.0078***	0.0147***	0.0044	0.0070*
	(0.0030)	(0.0057)	(0.0063)	(0.0040)
No Ratings	-0.0360***	-0.0398**	-0.0263	-0.0336**
	(0.0111)	(0.0193)	(0.0227)	(0.0157)
No Ratings X Post iOS 11	0.0453***	0.0945***	0.0118	0.0372**
	(0.0133)	(0.0253)	(0.0267)	(0.0178)
N	137,272	42,549	47,712	47,011

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 11: The Effect of Apple's Review Reset Policy on Updating Behavior, All Ranked Apps

	(1) All	(2) Education	(3) Productivity	(4) Utilities
Avg Review Score	-0.0109***	-0.0076***	-0.0156***	-0.0100***
	(0.0012)	(0.0023)	(0.0022)	(0.0019)
Avg Review Score X Post iOS 11	0.0064***	0.0043*	0.0117***	0.0048***
	(0.0013)	(0.0025)	(0.0027)	(0.0018)
No Ratings	-0.0178***	-0.0176*	-0.0263***	-0.0117*
	(0.0049)	(0.0095)	(0.0092)	(0.0071)
No Ratings X Post iOS 11	0.0356***	0.0432***	0.0452***	0.0266***
	(0.0056)	(0.0110)	(0.0111)	(0.0082)
N	555,208	161,822	193,600	199,786

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 12: Response Heterogeneity with the Size and Length of the Reputational Penalty, Average Rank <= 150^{th}

	Penalty Size		Penalty Length	
	(1)	(2)	(3)	(4)
	Top 25%	Bottom 75%	Top 25%	Bottom 75%
Avg Review Score	-0.0137**	-0.0154***	-0.0181**	-0.0128***
	(0.0066)	(0.0029)	(0.0078)	(0.0028)
Avg Review Score X Post iOS 11	0.0058	0.0088***	-0.0013	0.0106***
	(0.0092)	(0.0029)	(0.0095)	(0.0027)
No Ratings	-0.0488*	-0.0316***	-0.0450	-0.0276**
	(0.0272)	(0.0120)	(0.0353)	(0.0112)
No Ratings X Post iOS 11	0.0503	0.0445***	0.0297	0.0592***
	(0.0382)	(0.0131)	(0.0444)	(0.0122)
N	34,179	103,093	34,316	102,956

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 13: Response Heterogeneity with the Size and Length of the Reputational Penalty, All Ranked Apps

	Penalty Size		Penalty Length	
	(1)	(2)	(3)	(4)
	Top 25%	Bottom 75%	Top 25%	Bottom 75%
Avg Review Score	-0.0146***	-0.0097***	-0.0146***	-0.0100***
	(0.0027)	(0.0014)	(0.0030)	(0.0013)
Avg Review Score X Post iOS 11	0.0076**	0.0058***	0.0059*	0.0068***
	(0.0030)	(0.0014)	(0.0035)	(0.0012)
No Ratings	-0.0359***	-0.0129**	-0.0330**	-0.0138***
	(0.0110)	(0.0054)	(0.0129)	(0.0050)
No Ratings X Post iOS 11	0.0445***	0.0338***	0.0505***	0.0401***
	(0.0132)	(0.0061)	(0.0159)	(0.0055)
N	138,909	416,299	138,717	416,491

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 14: Response Heterogeneity with the Size and Length of the Reputational Penalty, All Ranked Apps (10-90 Split)

	Penalty Size		Penalty Length	
	(1)	(2)	(3)	(4)
	Top 10%	Bottom 90%	Top 10%	Bottom 90%
Avg Review Score	-0.0125***	-0.0107***	-0.0167***	-0.0102***
	(0.0043)	(0.0013)	(0.0055)	(0.0012)
Avg Review Score X Post iOS 11	0.0045	0.0066***	0.0038	0.0065***
	(0.0058)	(0.0013)	(0.0069)	(0.0012)
No Ratings	-0.0386**	-0.0162***	-0.0408*	-0.0151***
	(0.0181)	(0.0050)	(0.0241)	(0.0049)
No Ratings X Post iOS 11	0.0422*	0.0362***	0.0473	0.0395***
	(0.0245)	(0.0056)	(0.0334)	(0.0054)
N	55,570	499,638	55,496	499,712

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 15: The Effect of Apple's Review Reset Policy on Update Content, Average Rank <= 150^{th}

	Version Number				Support Vector Machine			
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	All	Education	Productivity	Utilities	All	Education	Productivity	Utilities
Avg Review Score	-0.0083**	-0.0123	-0.0121**	0.0039	-0.0117	-0.0310*	-0.0058	0.0082
	(0.0040)	(0.0081)	(0.0056)	(0.0080)	(0.0092)	(0.0159)	(0.0141)	(0.0167)
Post iOS 11 X Avg Review Score	0.0036	0.0031	0.0211**	-0.0251**	0.0135	0.0209	0.0077	-0.0060
	(0.0061)	(0.0117)	(0.0087)	(0.0106)	(0.0124)	(0.0208)	(0.0194)	(0.0229)
No Ratings	-0.0199	-0.0473	-0.0236	0.0232	-0.0262	-0.1049	-0.0125	0.0734
	(0.0169)	(0.0336)	(0.0242)	(0.0319)	(0.0368)	(0.0649)	(0.0580)	(0.0638)
Post iOS 11 X No Ratings	0.0147	0.0212	0.0669*	-0.0881**	0.0484	0.0536	0.0207	0.0077
	(0.0253)	(0.0474)	(0.0364)	(0.0404)	(0.0494)	(0.0849)	(0.0796)	(0.0898)
N	16,765	5,610	6,854	4,301	16,765	5,610	6,854	4,301

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 16: The Effect of Apple's Review Reset Policy on Update Content, All Ranked Apps

	Version Number				Support Vector Machine			
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	All	Education	Productivity	Utilities	All	Education	Productivity	Utilities
Avg Review Score	-0.0069***	-0.0164***	-0.0077*	0.0019	-0.0209***	-0.0360***	-0.0133*	-0.0139*
	(0.0025)	(0.0046)	(0.0044)	(0.0041)	(0.0047)	(0.0082)	(0.0075)	(0.0081)
Post iOS 11 X Avg Review Score	0.0017	0.0136**	0.0011	-0.0074	0.0174***	0.0281**	0.0093	0.0066
	(0.0036)	(0.0069)	(0.0061)	(0.0056)	(0.0063)	(0.0114)	(0.0108)	(0.0102)
No Ratings	-0.0080	-0.0447**	-0.0122	0.0249	-0.0456**	-0.0992***	-0.0219	-0.0114
	(0.0103)	(0.0188)	(0.0182)	(0.0165)	(0.0182)	(0.0333)	(0.0296)	(0.0307)
Post iOS 11 X No Ratings	0.0014	0.0362	0.0088	-0.0338	0.0454*	0.0514	0.0181	0.0304
	(0.0139)	(0.0266)	(0.0245)	(0.0206)	(0.0245)	(0.0461)	(0.0422)	(0.0380)
N	60,282	18,520	23,969	17,793	60,282	18,520	23,969	17,793

Standard errors are in parentheses, and are clustered at the app level. Other control variables included in all regressions, but whose coefficients are not reported, are price, the number of reviews for the current app version, the age of the current version, and the file size of the app, week fixed effects, and app fixed effects. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.