Information and Software Technology 87 (2017) 119–139

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Investigating the relationship between price, rating, and popularity in the Blackberry World App Store

Anthony Finkelstein, Mark Harman, Yue Jia, William Martin, Federica Sarro*, Yuanyuan Zhang

Department of Computer Science, University College London, London, UK

ARTICLE INFO

Article history:
Received 31 August 2016
Revised 2 February 2017
Accepted 10 March 2017
Available online 11 March 2017

Keywords:
App store analysis
App features
Mobile apps
Data mining
Natural language processing

ABSTRACT

Context: App stores provide a software development space and a market place that are both different from those to which we have become accustomed for traditional software development: The granularity is finer and there is a far greater source of information available for research and analysis. Information is available on price, customer rating and, through the data mining approach presented in this paper, the features claimed by app developers. These attributes make app stores ideal for empirical software engineering analysis.

Objective: This paper[1] exploits App Store Analysis to understand the rich interplay between app customers and their developers.

Method: We use data mining to extract app descriptions, price, rating, and popularity information from the Blackberry World App Store, and natural language processing to elicit each apps' claimed features from its description.

Results: The findings reveal that there are strong correlations between customer rating and popularity (rank of app downloads). We found evidence for a mild correlation between app price and the number of features claimed for the app and also found that higher priced features tended to be lower rated by their users. We also found that free apps have significantly (p-value < 0.001) higher ratings than non-free apps, with a moderately high effect size ($\hat{A}_{12} = 0.68$). All data from our experiments and analysis are made available on-line to support further investigations.

1. Introduction

App stores provide a rich source of information about apps concerning their customer-, business-, and technically- focused attributes. Customer information is available concerning the ratings accorded to apps by the users who downloaded them. This provides both qualitative and quantitative data about the customer perception of the apps. Business information is available, giving the number (or rank) of downloads and also price of apps. Technical information is available in the descriptions of apps, but it is in free text format, so data mining is necessary to extract the technical details.

We find ourselves at a unique situation in software engineering research: in no previous software engineering development and deployment environment have software engineering researchers been able to access publicly available data that links all of these important attributes:

- The customers' opinions of software, in the form of the reviews they leave;
- The popularity of software, in the form of its rank and/or number of downloads;
- The price charged for software;
- The technical claims made by developers concerning the list of features offered by their software.

Of course, this information may not be complete or fully reliable: customers may, for various reasons, leave reviews that do not reflect their true opinions. Either intentionally or unintentionally, developers may not be entirely truthful about the technical claims made. Price information may only concern the price of the

* Corresponding author.
E-mail addresses: a.finkelstein@ucl.ac.uk (A. Finkelstein), mark.harman@ucl.ac.uk (M. Harman), yue.jia@ucl.ac.uk (Y. Jia), w.martin@ucl.ac.uk (W. Martin), f.sarro@ucl.ac.uk (F. Sarro), yuanyuan.zhang@ucl.ac.uk (Y. Zhang).

[1] This paper is an extended version of our short paper at MSR 2012 [1]; a technical report is also available [2].

http://dx.doi.org/10.1016/j.infsof.2017.03.002

app, and may not include 'in app purchases' and other costs associated with using the app. Nevertheless, it is not unreasonable to hope that broad observations about whole classes of apps may still prove to be robust; the large number of apps on which such observations are based tends to support robustness.

In this paper we mine the Blackberry World App Store for data to support App Store Analysis. The technical information we mine is provided by the text description of each app. We mine this using techniques inspired by work on mining natural language descriptions for technical information. In this way, our work resembles work on mining other forms of natural language product information [3–5]. Though there has been previous work on app store analysis [6], Harman et al. [1] were the first to analyse the features extracted from app descriptions and their relationship to non-technical information. It is important to note that we are extracting *claimed features*, though hereinafter we shall often refer to them simply as 'features' for brevity. That is, the feature information we extract reflects features that are present in the descriptions of apps, but they are not necessarily *present* in the app itself. We believe that this is an interesting aspect of our app store analysis: it gives us an opportunity to explore the relationship between claimed features and other app store data. Claimed features denote an interesting technical category in its own right. Whether or not there is a relationship between claimed features and features present in the app remains an interesting topic to be investigate in future work.

Specifically, in this paper, we are concerned with the correlation between the price, popularity, and ratings accorded to apps by their users. We are also interested in the correlation between these three properties of the features of the apps. Correlation analysis allows us to address fundamental questions for any app store, such as:

1. Do apps that tend to get a higher rating also tend to be more popular?
2. Do apps that cost the customer more tend to get a lower rating?
3. Do the extracted features enjoy any of the above correlations?

The primary contributions of this paper are:

1. We investigate in more detail the concept of Mining App Stores for business, technical, and customer information introduced in our previous MSR 2012 short paper [1]. This is a considerably extended version of that work, which develops the research agenda set out in the MSR paper. It is important to note that there has been previous work analysing apps, for example app security [7], code reuse between apps [8], and dependence analysis [9]. The primary conceptual contribution by Harman et al. [1] has been to introduce the idea that App Stores can be mined for connected sets of data, allowing us to analyse the relationship between technical, customer, and business aspects of the market [1]. The present paper extends our preliminary analysis of non-free Blackberry apps [1], to consider both free and non-free apps and the correlations between their claimed features, rating, popularity, and price in more detail.
2. We study the distributions of prices and ratings over all apps. We found a very large number of zero-rated apps. We find that prices tend to be lower than $5.00 for most apps, but there are frequency peaks at 'round number' prices, such as $10 and $20.
3. We present a procedure to mine feature information from app descriptions. This approach uses natural language processing algorithms to extract likely feature descriptions as bitri-grams (i.e., 2-grams or 3-grams). Specifically , we describe the procedure outlined in our MSR short paper [1] in detail, thus allowing other researchers to replicate, extend, or build on our work.
4. We empirically investigate the correlations between price, rating, and popularity for free and non-free apps, and also their claimed features. For both we find evidence of a strong correlation between ratings and rank of downloads: highly rated apps are more frequently downloaded, as one might expect. We find little evidence of correlations between price and either rating or popularity for apps, but we did find evidence for a mild inverse correlation between feature price and feature rating when considering price points; customers tend to rate higher priced features less favourably than lower priced features. We also find that free apps have significantly (p-value < 0.001) higher rating than non-free apps, with a moderately high effect size ($\hat{A}_{12} = 0.68$), suggesting that users are not entirely insensitive to the pricing choices of developers.

The rest of the paper is organised as follows: Section 2 introduces our app analysis framework and describes the metrics that capture the attributes of a feature. Section 3 presents the design of our empirical study, the results of which are analysed in Section 4. Section 5 discusses the limitations of the present study, while Section 6 describes other work related to ours. Section 7 concludes and presents directions for future work.

2. App analysis framework

Our approach to app store analysis consists of four phases shown in Fig. 1. The first phase extracts raw data from the app store (in this case Blackberry World App Store[2], though our approach can be applied to other app stores with suitable changes to the extraction front end as detailed in the following). In the second phase we parse the raw data extracted in the first phase to retrieve all the available attributes of each app relating to price, ratings, and textual descriptions of the app itself. In the third phase we leverage app descriptions to identify technical information; in particular, we use information retrieval to extract the features of apps from their textual descriptions. The final phase computes metrics on the technical (i.e., claimed features), business (i.e., prices), and customer (i.e., ratings) information extracted.

The rest of this section explains each step of our approach in more detail.

Phase 1 (Data Extraction): We implemented a customised web crawler to collect raw webpage data from the Blackberry app store. Due to the existence of a large number of apps, the Blackberry app store does not provide a direct way to access all the apps iteratively. Thus, our crawler collects app data in two steps. First, it collects all category information from the app store and scans each category page to find the list of URLs of all the apps in each category. It then visits the webpage of each app within each category and saves it as raw app data.

Phase 2 (Parsing): We extract a set of attributes for each app by parsing the raw data according to a set of search rules. The search rules are based on HTML tags identified manually, each of which specifies a unique signature for each attribute of interest. For example, we can retrieve the title of an app by searching the value of the $\langle h1 \rangle$ HTML tag with the attributes 'id=title' and 'class=awwsProductDetailsContentItemTitle'.

The extraction process cannot be entirely automated. Some attribute fields populated by humans require a further refinement process that accounts for the various ways in which the humans who populate the App Store data might provide equivalent information. For example, the values of the price field for a free app could be '0', 'Free', 'Free for one week' or a word that means 'free' in a language other than English. We assign a value 0 for the price of all such apps.

[2] http://appworld.blackberry.com/webstore/.

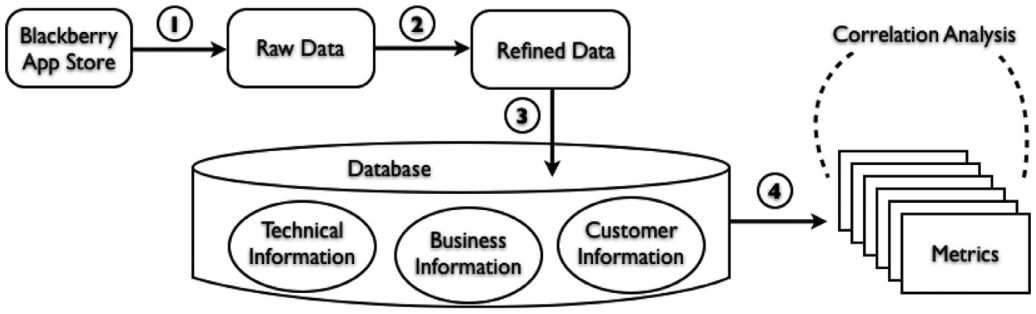


Fig. 1. Overall App Analysis Architecture: A four phase approach extracts, refines, and stores app information for subsequent analysis.

After having manually investigated Blackberry app web pages, we developed search rules to capture information about Name, Category, Icon, Description, Price, Release Time, Version, Size, Language, Customers' Rating, Number of Ratings, and Rank of Downloads. However, the analysis of this work is focused on the Category, Description, Price, Customers' Rating, and the Rank of Downloads attributes. Once this manual step is complete the entire process is fully automated (until such time that the app store changes structure).

To apply our approach to a different app store we need to modify the URL information in the data extractor and the search rules in the parsing phase in order to accommodate different app store structures and data representations, respectively.

Phase 3: (Data Mining Features): App features can be defined in many ways. For our purposes, feature information is data mined from app descriptions. For example, "7-days weather forecast" is a feature mined from apps in the weather category while "receive facebook message" is a feature mined from IM & Social Networking apps. The definition of an app feature (as mined by our process) is as follows:

"A feature is a claimed functionality offered by an app, captured by a set of collocated words in the app description and shared by a set of apps in the same category."

Since app descriptions are written in natural language, extracting features from the text requires data mining techniques usually associated with Natural Language Processing (NLP). We developed a simple four-step NLP algorithm to extract feature information and implemented it using the Natural Language Toolkit (NLTK), a comprehensive natural language processing package written in Python [10]. In this work we focus on app descriptions written in English, however the framework is language independent and works with different corpora [11].

Our feature extraction algorithm is presented in Algorithm 1.

Algorithm 1 Feature Extraction Algorithm.

```
Require: apps
  rawFeatures = [ ]
  featureLets = [ ]
  for all apps do
    if featurePattern exists in currentApp.description then
      rawFeatures.append (extractFeaturePattern (currentApp))
    end if
  end for
  for all rawFeatures do
    refinedFeatures=refineRawFeatures(currentRawFeature)
  end for
  featureLets = findTriaGramCollocation (refinedFeatures) {NLTK}
  features = getGreedyClusters (featureLets)
  return features
```

The first step extracts raw feature patterns, thereby identifying the 'coarse features' of apps. Feature patterns are informal patterns

```
Stay informed and prepared with live local weather,
severe weather alerts, in-depth forecasts, camera views,
and radar maps. The features of the WeatherBug
application for the BlackBerry Storm include:
* Live neighborhood weather from over 8,000 weather
 stations in the U.S.
* Current weather, forecast and NWS alerts.
* 7-day and weekend forecasts.
* radar animation and cloud coverage.
* View snapshots and time-lapse animations from more
 than 2,000 weather cameras
* WeatherBug Community photos. Share you own weather
 photos.
```

Fig. 2. WeatherBug: An example of description of a weather app.

which developers used to list and clarify the features released. Fig. 2 shows the description of a non-free Blackberry weather app, named "WeatherBug". We will use this example to illustrate our feature mining algorithm.

In Fig. 2, the list starting with '*' is an example of a raw feature pattern which summarises the main features of the app. Our algorithm searches for common HTML list elements, such as '*' or '-', in the description of apps to locate raw feature patterns If the sentence prior to an HTML list contains at least one keyword from the set of words "include, new, latest, key, free, improved, download, option, feature", the HTML list is saved as the raw feature pattern for this app. These keywords have been selected based on a manual assessment carried out on the apps of two randomly selected categories (i.e., `Weather` and `Finance`). We apply this process to all the apps in the same category to create a list of raw features, as shown in Fig. 2. A potential threat can arise if the features are not listed in an HTML list but in plain text. However, very often the features are listed in HTML list in the apps' description extracted from the Blackberry World App Store. Moreover, since we applied our analysis on all the apps of a given category (for all the categories), a feature that is missing from the description of one app might appear in the descriptions of other apps.

The second step of the algorithm refines the raw feature patterns by removing 'noise'. We first tokenise the raw feature patterns into a lower case token stream and then apply the following filtering: First, non-English and numerical characters are removed from the token stream. Secondly, incidental, unimportant 'noise' words are filtered out. The determination of these elements is delegated to the English language STOPWORDS set in the NLTK data package. If typos occur in the description, these are implicitly handled by the natural language process techniques, as typos should have very low occurrences compared to the correctly spelled words. Finally, each remaining word is transformed into its 'lemma form' using the WORDNETLEMMATIZER function from NLTK, thereby homogenising singular/plural, gerund endings, and other non-germane grammatical details. Fig. 3 shows an example of the refined feature pattern for the weather app example.

```
[live, neighborhood, weather, weather, station, us]
[current, weather, forecast, nws, alert]
[7-day, weekend, forecast]
[radar, animation, cloud, coverage]
[view, snapshot, time-lapse, animation, weather, camera]
weatherbug, community, photos, share, weather, photo]
```

Fig. 3. Examples of refined feature patterns from the WeatherBug app.

Table 1
Featurelets: This table shows some examples of the featurelets extracted by applying the proposed approach to the weather app description reported in Fig. 2.

Tri-gram collocated tokens	Tri-gram association score
[*animation, weather, camera*]	2891
[*neighborhood, weather, station*]	2826
[*share, weather, photo*]	2798
[*live, neighborhood, weather*]	2792
[*time–lapse, animation, weather*]	2780
[*7–day, weekend, forecast*]	2230

In the third step, the algorithm extracts a set of 'featurelets' from the refined feature patterns. A featurelet is a set of commonly occurring collocated words, describing a core function of apps. We perform a collocation analysis to find words that associate frequently from the refined feature pattern, built on top of NLTK's N-gramCollocationFinder package. The collocation analysis is designed to work with a set of apps, and in our experiments we applied it to apps in each category. We experimented with the settings for $N = [2, 3, 4]$ and found that the setting $N = 3$ generally achieved the best results. The determination of 'best results' was made by the experimenters' subjective human assessment of whether the resulting *n*-grams appeared to be meaningful. However, this human judgment was more systematically tested in the simple 'sanity check' human study detailed in our technical report [2].

Table 1 shows the featurelets extracted from the weather app example. Each of the featurelets on the left column has three tokens, because we used the tri-gram collocation model here. The right column shows the tri-gram association score, which indicates how frequently these tokens are associated together in the pool of the refined features of all weather apps. For each category of apps, we rank and select the best M featurelets based on the NLTK *N*-gram association measures. M is the number of featurelets. We experimented with the settings for $M = [100, 200, 500]$ and chose $M = 200$ in our experiments, once again based on the experimenters' assessment of the choice that produced the more apparently meaningful result.

Some extracted featurelets are similar to each other. For example, in Table 1, featurelets [*neighborhood, weather, station*] and [*live, neighborhood, weather*] share two common tokens ('neighborhood' and 'weather'). The higher the tri-gram association score between two featurelets, the more frequently they are associated together. Step 4 applies a greedy hierarchical clustering algorithm to aggregate similar featurelets together, as shown in Algorithm 2. This algorithm treats each featurelet as one cluster initially. It then repeatedly combines clusters if their similarity measure is greater than a predefined similarity threshold. The similarity measure is the number of common tokens shared by each cluster, and we chose 0.5 as the similarity threshold in our experiment, based on our assessment of result meaningfulness with different threshold values. The common words from each cluster are extracted as 'core features'. Table 2 shows the example of core features extracted from the featurelets shown in Table 1. We shall refer to a core feature as 'bitri-gram' since it can be represented by either a bi-gram or a tri-gram.

Algorithm 2 Greedy Feature Cluster Algorithm.

```
Require: featureLets
Require: greedyThreshold
  greedyClusters = [ ]
  greedySimilarities = [ ]
  for all featureLets do
    greedyClusters.add (featureLet)
  end for
  for i = 0 → len (greedyClusters) - 1 do
    currCluster = greedyClusters[i]
    for j = 0 → len (greedyClusters) - 1 do
      if i == j then
        currSimilairy = 0
      else
        currSimilairy = getSimilarity (currCluster, greedyClusters[j])
      end if
      greedySimilarities.add (currSimilairy)
    end for
    if max (greedySimilarites) > greedyThreshold then
      maxIndex = getMaxClusterIndex (greedySimilarites)
      mergeClusters (currCluster, greedyClusters [maxIndex])
    end if
  end for
  return greedyClusters
```

Table 2
Core Feature: This Table shows the core features (i.e., 'bitri-gram') extracted by applying the last step of the proposed approach to featurelets reported in Table 1.

Core feature	Optional tokens
[*animation, weather*]	[*time–lapse, camera*]
[*neighborhood, weather*]	[*station, live*]
[*share, weather, photo*]	N/A
[*7–day, weekend, forecast*]	N/A

Because of the importance of the feature mining process to any kind of analysis, we performed a sanity check of the features extracted by assessing whether these claimed features were meaningful to humans [2]. To this end, experts were asked to say whether they believed that a given claimed feature represented a feature or not. The questionnaire contained both claimed features (i.e., bitri-grams extracted by the mining technique used herein) and random features (i.e., bitri-grams created by randomly selecting words from app descriptions). The results showed that developers often classify the claimed features as a feature and the random features as a non-feature (i.e., Precision = 0.71 Recall = 0.77). This provided some initial evidence that the features we extract are meaningful to developers.[3]

Phase 4: (Analysis): The final phase of our approach involves the analysis of the mined information. This phase is application specific. The mined information can, indeed, support many other app related analyses. For example, feature metrics have been used by Sarro et al. [12] to investigate the migration of claimed features across product categories in two existing app stores, and subsequently by Al-Subaihin et al. [13] to cluster apps based on their claimed functionalities.

In the analyses presented in this paper, we collect metrics about apps and their features and use them in the correlation analysis based on features. Specifically, we introduce some simple metrics that capture the attributes of a feature, f in terms of the corre-

[3] Due to space limit imposed by the journal we cannot report herein greater details, which can be found in our technical report [2].

sponding attributes of all apps that posses the feature f. This allows us to compute useful information about the features of an app. In the following we formalise the definitions of these metrics to support replication and future work.

We shall define our metrics with respect to an app database, which contains the information extracted for the app store. Let $AR(a, d)$, $AD(a, d)$, and $AP(a, d)$ denote the rating, rank of downloads, and price, respectively, of the app a in the app database d. Let $\sharp(s)$ denote the size (cardinality) of set s. Let $S(f, d) = \{a_1, \ldots, a_m\}$ such that feature f is shared by all m apps $a_1, \ldots, a_m$ in an app database d.

We can extend $AR(a, d)$, $AD(a, d)$ and $AP(a, d)$ to the features extracted from app descriptions, by defining the rating, rank of downloads, and price of a feature, f to be the mean rating, rank of downloads, and price for all the apps that share f. More formally, we extend the metric X ($X \in \{AR, AD, AP\}$) defined from (app, database) pairs to numbers, to a metric F defined from (feature, database) pairs to numbers, as follows:

$$F(f, d) = \frac{\sum_{a_i \in S(f,d)} X(a_i, d)}{\sharp(S(f, d))}$$

The same approach can be used to extend any metric X of type app $\times$ database $\rightarrow \mathbb{R}$ to one of type feature $\times$ database $\rightarrow \mathbb{R}$

We also considered the median rank of downloads and rating, because app popularity is measured as an ordinal rank (called 'rank of downloads' by several app stores) and the rating is a star rating (recorded for each app as a value from 0 to 5 stars in half star increments). These two measurements are clearly ordinal scale measurements and so the median is the most suitable centrality measure [14]. For price, the use of median (instead of mean) for value aggregation is more questionable. We did observe ordinal pricing behaviour. For example, the app store requires developers to charge in whole dollar increments. Furthermore, prices chosen by developers tend to cluster around ten, twenty, and thirty dollar 'price points', suggesting some kind of implicit 'ordinal scale' properties. However, the scale could equally well be argued to be a ratio scale. In order to check that our choice of mean or median aggregation did not affect the results we report here, we computed all results using both mean and median to aggregate over app prices, ratings, and popularity. The findings remained as reported here, suggesting that the choice of aggregation technique is relatively unimportant for the features studied.

3. Empirical study design

This section explains the design of our empirical study, the research questions we set out to answer, and the methods and statistical tests we used to answer these questions.

3.1. Research Questions

We are studying relationships between price, rating, and popularity (rank of downloads) for apps and the features we extract from their descriptions. We therefore start by analysing the characteristics and distribution of these data.

Popularity is measured in terms of the rank of downloads, so this distribution is always a monotonically decreasing ranking. Also, since popularity is measured as a rank position in the league table of most downloaded apps (rank of downloads) this means that lower numbers (higher rank positions) indicate higher popularity on an ordinal scale.

We extracted 1,008 different features from the app descriptions in our dataset, and so a natural question to ask is how these features distribute over the apps from which they are extracted. In addition to app descriptions, we have mined rating and pricing information from the Blackberry World App Store. We present the distributions of these data, over both the apps and features extracted from apps. These data form the answer to RQ0:

RQ0: What are baseline data on Price, Rating, and Feature Distributions?

The next three research questions investigate the correlation between price, rating, and popularity (i.e., rank of downloads) for non-free apps and between rating and popularity (i.e., rank of downloads) for free apps (those for which the price charged at the time of download is zero). These questions were addressed in the conference version [1] of this paper only for non-free apps. In this journal extension, we also consider free apps and investigate the correlations we find in greater depth.

RQ1: Price/Rating Correlation. What is the correlation between the Price (P) and the Rating (R) for non-free apps, overall and in each category?

RQ2: Price/Popularity Correlation. What is the correlation between the Price (P) and the rank of Downloads (D) for non-free apps, overall and in each category?

RQ3: Rating/Popularity Correlation. What is the correlation between the Rating (R) and the rank of Downloads (D) for free and non-free apps, overall and in each category?

In the conference version [1] of this paper, we observed correlation between rating and popularity, both for the apps themselves, and also for the features we extracted from them. In this extended version of the paper, we study this question in greater detail. In the Blackberry World App Store, at the time we took our snapshot, it was possible for a reviewer to assign a zero rating score. It was also possible that the particular app may have no reviews at all, which would also yield a zero rating score. It is not possible to distinguish between these two types of zero rated score. Furthermore, we might speculate that an app which has relatively few ratings available lacks sufficient evidence for the overall mean rating recorded by users in general. Therefore, we consider both all apps and subsets of apps having different numbers of reviews (i.e., from 0 to 9) available, and thereby enjoy a larger evidence base, from which we may draw inferences about mean rating among the user community:

Finally, in the conference version of the paper, we observed that there was no correlation between price and either rating or popularity. This surprised us, since we might conjecture that an app developer would have to try harder, *per se*, to garner higher ratings and popularity should they choose to charge a higher price. Therefore, we investigate whether focusing on price ranges (rather than absolute price) might lead to different results. We also investigate whether there is a correlation between price and the number of features offered: perhaps apps that offer more features charge a higher price? This motivates our final research question, which investigates in more detail, the apparent absence of evidence for correlations involving price highlighted in the conference version of this paper [1]:

RQ4: Is there a stronger correlation involving Price when we 'zoom in' on specific ranges of price or between price and number of features or shared features in an app?

RQ4.1: Is there any difference in Rating and Popularity for free apps compared to non-free apps?

3.2. Data Employed in the Empirical Study

To answer the research questions, we constructed an app store database from the Blackberry World App Store, taken by extracting information from all free and non-free apps present on the 1st of September 2011, our census date for this study. We were able to mine all the data available in the store at that time, thus this study does not suffer from the App Sampling Problem [15].

Table 3 shows summary data (i.e., number of apps, features, mean, median, and minimum app price, rank of downloads, and rating) concerning the 19 categories in this app store database for non-free and free apps. The price is the price charged to download the app. The rating data is extracted from the reviews left by customers. The rank of downloads is the ranking position (relative to other apps) recorded by the app store for the downloads of the app at our census date.

We can observe that the number of apps contained in each category ranges from 45 to 11504 for non-free apps and 42 to 1,257 for free apps. The categories 'Shopping' and 'News' contain the lowest number of non-free apps (i.e., 45 and 68, respectively), while the 'Weather' category contains the lowest number of free apps (i.e., 42). The categories 'Reference & Books' and 'News' contain the highest number of non-free and free apps, respectively.

3.3. Evaluation Criteria

We answer RQ0 by means of graphical analysis. In particular, we use histograms to visualise how many apps/features share the same price, rating, and how many apps share a same feature, over non-free and free apps.

To answer RQs 1, 2, and 3 (i.e., investigate the correlation between price, rating, and popularity of apps and features) we use scatterplots to show the relationship between two sets of data (i.e., price/rating, price/popularity, rating/popularity) and two association statistics (i.e., the Spearman's Rank Correlation [16] and the Pearson Product–Moment Correlation [17]) to measure their statistical dependence. The Spearman's correlation assesses how well the relationship between two pairs of observations can be described using a monotonic function, while the Pearson's correlation is a measure of their linear relationship. Both statistics range from $+1$ to -1, where $+1$ indicates perfect correlation and -1 indicates a perfect inverse correlation. No correlation is indicated by 0.

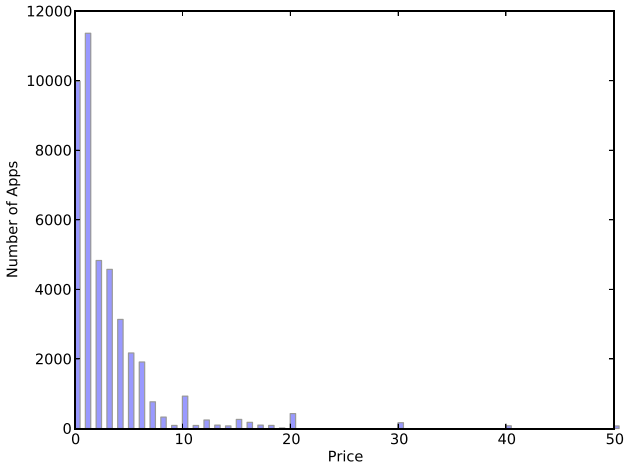
To provide an in-depth analysis of the statistical correlation between price, rating, and popularity (RQ3.2) we also analysed Spearman's (Pearson's) correlations at a finer grained level by grouping the apps depending on their minimum number of reviews (i.e., 0 to 9 reviews) and computing the correlation existing in each group. We visualise these results by means of graphs reporting, on the x axis, the number of reviews and, on the y axis, the rho and p-value provided by the Spearman's (Pearson's) test for each group. We refer to these graphs as correlation graphs.

To answer RQ4 we investigated the possibility that there may be correlations between price/rating and price/rank of downloads in sections of the data (perhaps for specific price ranges). We therefore further analysed these relationships by zooming into the scatterplots used to answer RQ3. We also considered the median rating and rank of downloads for each price point (for apps and features) to explore whether there is a correlation between these price points (chosen by developers) and the median rating (or popularity) given by customers for all apps (or features) charged at the associated price point. Moreover, we analysed whether there is any relationship between apps' price and their number of features or number of shared features, by means of scatterplots and Spearman's and Pearson's correlation tests.

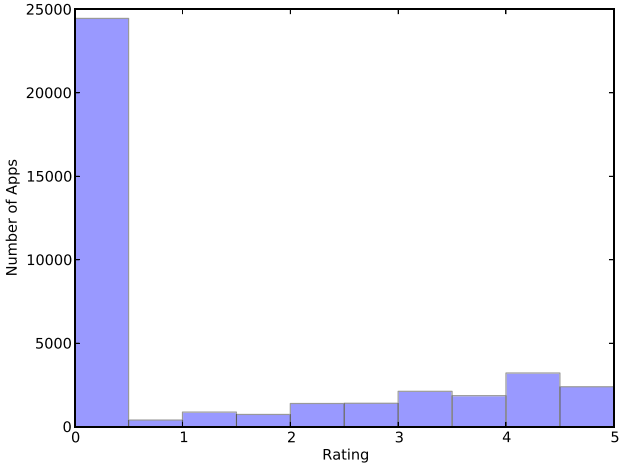
To answer RQ4.1 we investigated whether there is any statistically significant difference between the distributions of rating and rank of downloads for free and non-free apps. We also report on the effect size of any such significant differences, treating the apps for which we have data as a sample of all possible apps and using a non-parametric standardised effect-size measurement (Vargha and Delaney's $\hat{A}_{12}$ [18]) as a rough indicator of the degree of difference between the two types of app (free and non-free).

Table 3
Blackberry World App Store: Summary data computed for each category. The number of non-free apps and the mean price and rating are reported within the number of features mined for each category. Download information is provided by Blackberry World App Store as rank over all apps (free and non-free). To give a sense of the distributions of download rank positions, we present the mean, median, and minimum ranks.

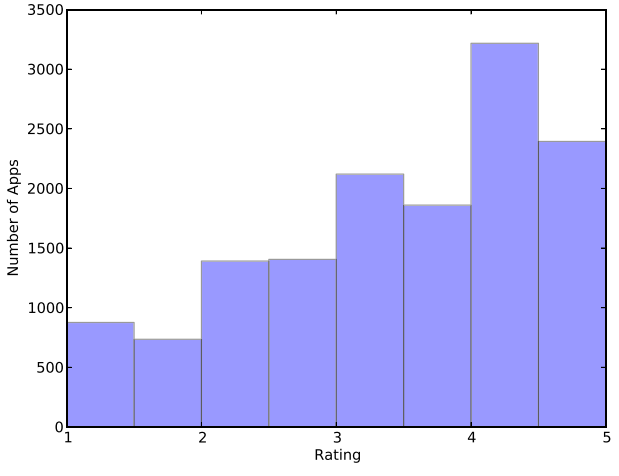
Category Name	Non-Free Apps							Free Apps						
	Number of	Number of	Price ($)	Rank of Downloads			Rating	Number of	Number of	Rank of Downloads			Rating
	Apps	Features	Mean	Mean	Median	Min	Mean	Apps	Features	Mean	Median	Min	Mean
Business	350	88	12.57	19063.48	18031	817	1.79	874	86	19491.55	16723	135	1.65
Education	576	81	5.68	22222.18	21739	1595	1.38	353	54	19024.98	15823	933	1.92
Entertainment	908	93	5.76	18413.13	16364	134	1.86	565	53	9721.01	6985	8	2.62
Finance	193	73	4.38	19593.59	16619	251	1.93	513	83	15394.49	12826	96	2.01
Games	2604	36	2.64	15919.49	13550.5	153	2.13	928	36	7109.65	5132.5	9	2.99
Health & Wellness	626	87	15.95	19852.88	18295.5	266	1.58	329	72	16168.94	13771	29	1.79
IM & Social Networking	150	70	4.42	14242.26	11513	22	2.55	305	77	8932.63	6171	0	2.29
Maps & Navigation	245	76	12.9	17140.75	13909	655	2.16	263	74	10489.11	7199	5	2.45
Music & Audio	499	83	2.05	24523.58	27248	204	0.99	1008	73	12699.36	10416.5	6	2.46
News	73	43	2.4	17485.36	15391	1393	1.73	1257	82	16037.91	13587	106	1.96
Photo & Video	393	89	2.51	21126.92	22879	15	1.4	192	68	7758.24	6064.5	33	2.47
Productivity	503	87	6.32	15124.95	11924	252	2.54	367	65	11050.12	8484	14	2.52
Reference & eBooks	11584	81	4.27	30388.93	31214.5	1155	0.12	366	38	18332.74	16800.5	16	1.67
Shopping	45	54	2.7	14785.51	11708	2543	2.33	229	64	13256.41	9933	56	2.08
Sports & Recreation	239	37	4.81	18808.38	16019	943	2.05	450	58	10509.9	8105.5	17	2.87
Themes	10936	32	3.12	21055.28	21254.5	18	1.68	552	30	5382.71	2836.5	27	3.35
Travel	764	81	4.81	25439.53	26112.5	553	0.67	441	85	13792.16	10828	12	2.38
Utilities	1362	73	4.61	16294.49	13994.5	63	2.32	950	67	11586.39	8932.5	7	2.67
Weather	58	65	7.51	12392.38	10288.5	309	2.44	42	58	7489.12	5193.5	21	2.61
All (avg)	1689.89	69.95	5.76	19151.21	17792.34	596.89	1.77	525.47	64.37	12327.76	9779.58	80.53	2.36



(a) Price distribution over apps

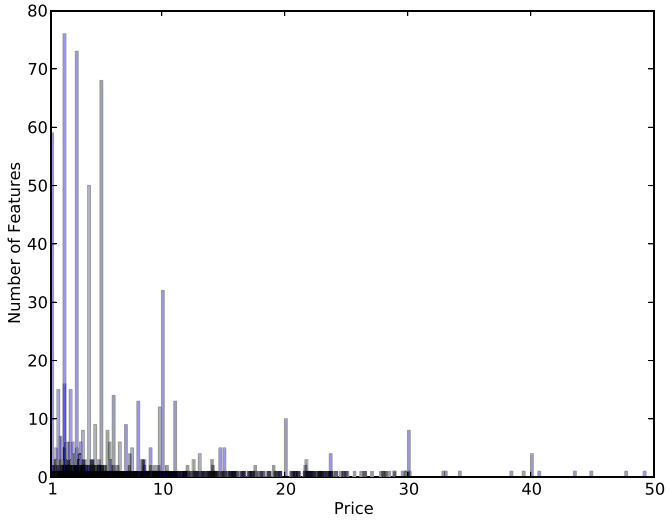


(b) Rating distribution over apps

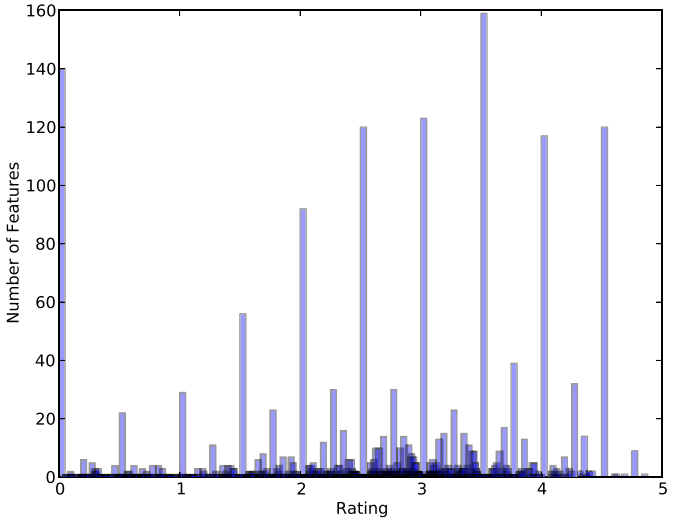


(c) Rating distribution over apps - zoom in

Fig. 4. RQ0: Distribution of prices and ratings at app level.



(a) Price distribution over features



(b) Rating distribution over features

Fig. 5. RQ0: Distribution of prices and ratings at feature level.

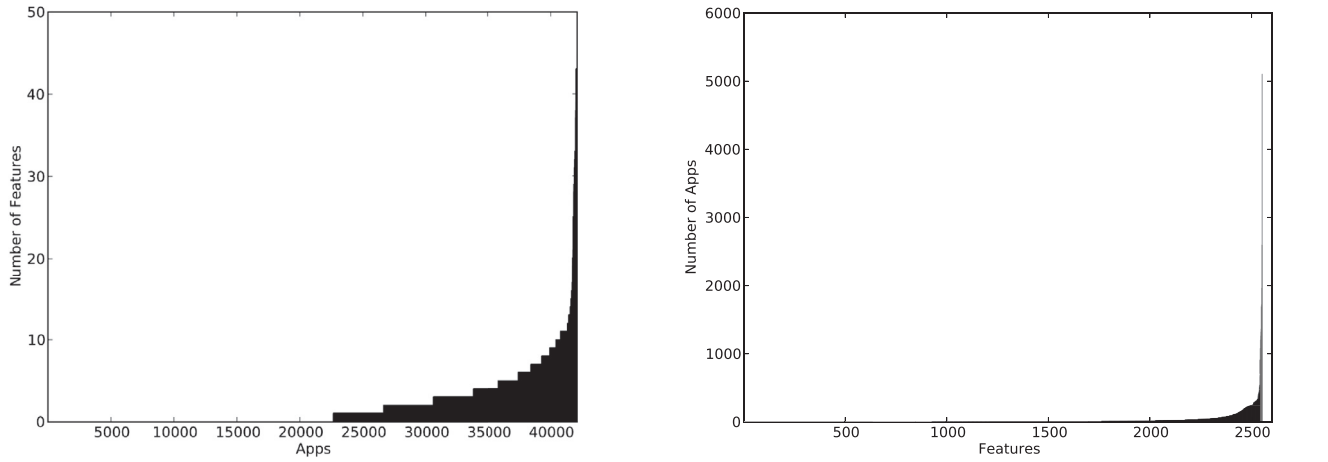
4. Result analysis

4.1. RQ0. What are baseline data on price, rating, and feature distributions?

Fig. 4 shows the distributions of prices and ratings over both apps and features extracted from app descriptions.

We can observe that there are fewer free-apps than non-free apps (see Fig. 4(a)). There is also a large number of zero-priced features (i.e., 1,223). These are, by definition, features only contained in zero-priced apps.[4] The largest app 'price point' (i.e., >10,000 apps) is at $0.99, dropping to approximately 5000 apps priced at $1.99 and hereinafter, the number of more expensive apps gradually decreases. Note that prices are set at discrete dollar intervals ($0.0, $0.99, $1.99, $2.99 ...) in the app store we considered.

[4] These features are not shown in the graph in Fig. 4 since this column would be an outlier, thereby making the differentiation of other columns harder to read.



(a) Number of features per each app (b) Number of apps sharing a same feature

Fig. 6. RQ0: Distribution of features over apps.

Despite the lower numbers of higher priced apps overall, we can observe peaks in the number of apps at the 'round number' price points ($10, $20, and $30), though these prices are, more precisely $9.99, $19.99, and $29.99 respectively. We observe a similar pattern for the prices of features extracted from the apps (see Fig. 5(a)). Since the attributes of a feature (such as prices) are aggregates over all apps that share that feature, the averaging effect produces a more fine-grained distribution of possible price points.

Turning our attention to the ratings distribution over apps (see Fig. 4(b)), we can observe a very large number of zero-rated apps. Looking at the zoomed-in subfigure for non-zero rated apps (Fig. 4(c)), we observe that the majority of these apps (i.e., more than 2,500) are rated 4 or 5 stars, about 2000 apps are rated between 3 and 4 stars, about 1500 apps are rated between 2 and 3 stars, and fewer than 1000 apps are rated between 1 and 2 stars.

The rating over features (see Fig. 5(b)) also reveals that a relatively high number (140 of the 1008 features extracted) have a zero rating. These features, by definition, are only contained in apps that have a zero rating. They could be removed as being of little consequence, but we did not apply this (or any other) filter to our algorithm's results, since we seek to validate our feature selection mechanism and we did not want to bias these (or other) results by experimenter interference. Since feature ratings are averaged over all apps that share the features, we see a finer-grained distribution of ratings for feature ratings than for app ratings, clustered around the original star scale. Not surprisingly, like the features' price distribution, this feature rating distribution is similar to the corresponding distribution for apps.

We also report the distributions of the features we extracted from app descriptions (see in Fig. 6). Specifically, Fig. 6(a) shows the number of features provided by each of the apps we considered. We can observe that this distribution follows a power law: A very few (69) apps (plotted on the right side of *Apps* axis) have more than 40 features, while a few (324 apps) have more than 20 features, and the majority (40,773 apps) have 10 or fewer features. In fact, more than half of the apps (85%) have fewer than 5 features. This is partly due to the fact that 65% of these apps belong to categories such as 'Themes' and 'Reference & eBooks', which provide users the sole functionality to download content (e.g., a theme or a book) and partially due to the fact that there are no feature patterns in their descriptions. In the rightmost half of the graph, that does not include these categories, we find that 6229 apps (14%) have more than 5 features.

A manual inspection of the attributes of the 69 apps that had more than 40 features revealed that these apps were created by the same developers, have a similar description, and share the same features that are related to photo editor and language dictionary functionality. This result is in line with the finding by Ruiz et al. [8] that there is, in the Android app store, heavy code reuse in photography apps.

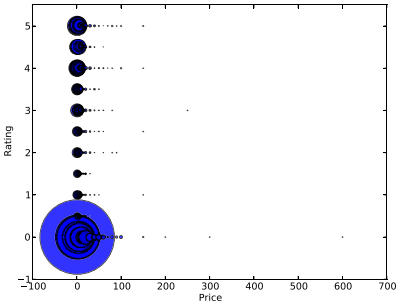
We also investigated the number of apps sharing a feature. This also follows a power law as can be seen from Fig. 6(b), which shows that a very few features are shared by more than one thousand apps. There are only 9 such highly prevalent features. A manual inspection revealed that these 9 features belong to apps from the 'Themes' category. They are features such as [*icon, set*], [*home, screen, icon*], [*background, screen*].

4.2. RQ1-3. Three correlations for non-free and free apps

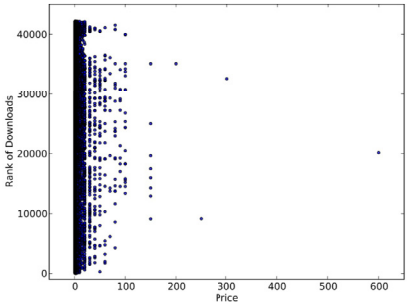
Figs. 7 and 8 show the scatterplots between Price (P), Rating (R), and Rank of Downloads (D) at app and feature levels. The size of each point denotes the number of apps to which that data point refers.

Graphs 7(a) and (b) suggest that the price of the apps is not strongly correlated with their popularity (i.e., apps of the same price can have different rank of downloads). However, we can observe from Fig. 7(a) that the cheapest apps often have zero ratings, while this observation does not applies at feature level (see Fig. 8(a) and (b)). There is an outlier in terms of price at $599; a price higher than many of the handsets on which it would reside when downloaded.

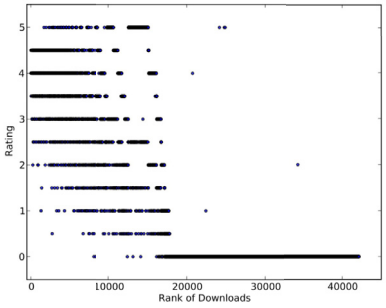
From graphs 7(c) and (d) we can observe that, regardless by their price, the non-rated apps tend to be less popular than the rated ones and that the higher the rating for an app, the more popular it tends to be. Perhaps more importantly, when we look at the overall trend of the median values of rank of downloads for a given rating (Fig. 7(e) and (f)), we can observe an apparently strong linear relationship for non-free apps. The relationship also appears to exist for free apps, though it may have a slightly more exponential character. To further investigate these observations based on the scatter plots of rating scores to median rank of downloads, we calculate both Spearman and Pearson correlation coefficients. For non-free apps the Pearson correlation coefficient (rho) is 0.78 (p=0.004), and the Spearman correlation coefficient (rho) is 0.71 (p=0.013). For the free apps the Pearson correlation coefficient (rho) is 0.83 (p=0.033), and the Spearman correlation coefficient (rho) is 0.65 (p=0.032). This indicates that there is a



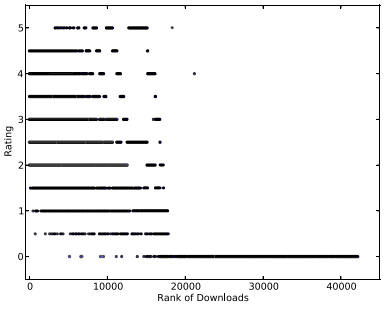
(a) PR non-free apps



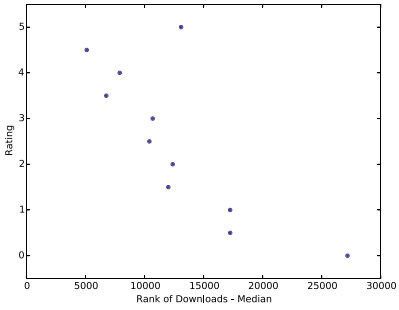
(b) PD non-free apps



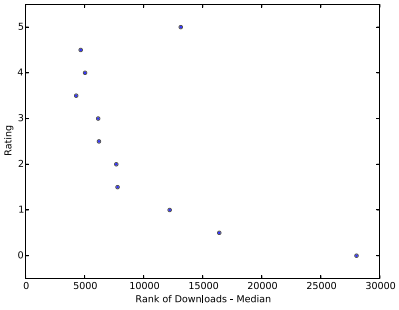
(c) RD non-free apps



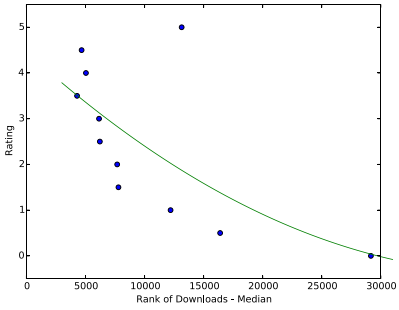
(d) RD free apps



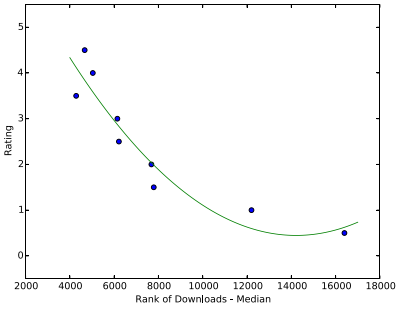
(e) MedianRD non-free apps



(f) MedianRD free apps



(g) MedianRD free apps - Polynomial model



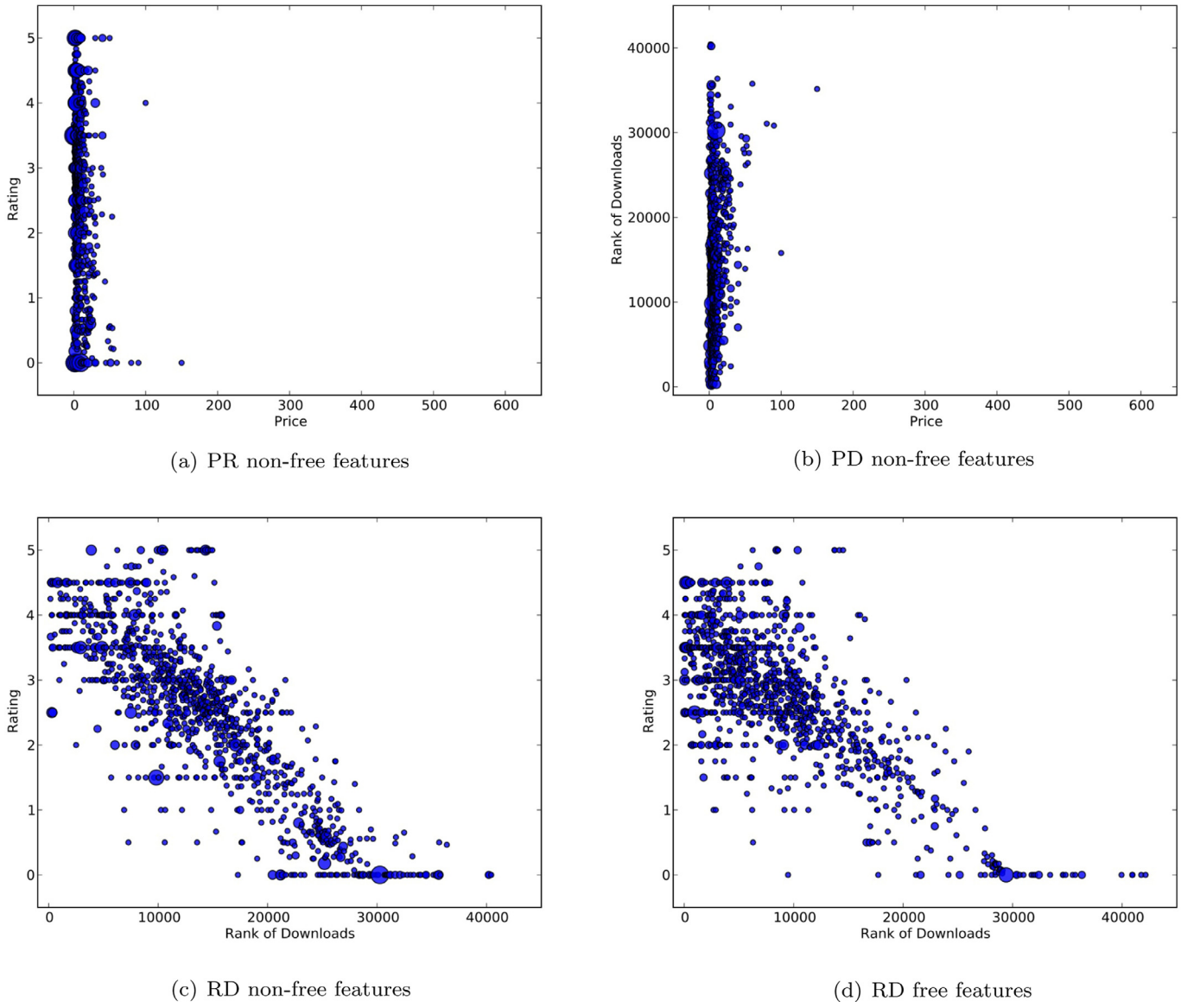
(h) MedianRD free apps ($0 < R < 5$) - Polynomial model

Fig. 7. RQ1-3: Scatterplot of Price (P), Rank of Downloads (D), and Rating (R) at app level.

strong correlation between rating and popularity for both free and non-free apps.[5]

In Fig. 7(e) and (f) we can also observe an interesting outlier for those apps with rating of 5.0 (the highest possible rating; five

[5] Similar findings hold when the overall rating (rating multiplied by number of ratings) is considered: Pearson rho is 0.69 (p=0.020) for free apps and 0.63 (p=0.035) for non-free apps; Spearman rho is 0.95 (p< 0.001) for free apps and 0.91 (p< 0.001) for non-free apps.



(a) PR non-free features

(b) PD non-free features

(c) RD non-free features

(d) RD free features

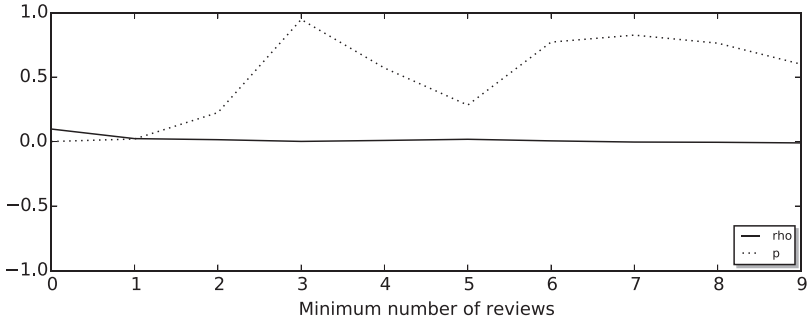
Fig. 8. RQ1-3: Scatterplot of Price (P), Rank of Downloads (D), and Rating (R) at feature level.

stars). The rank of downloads for these apps is notably higher than the overall trend would suggest; five star apps seem to be peculiarly unpopular, on average. If we remove the outlier then the Pearson rho for non-free apps becomes 0.91 (p=0.000) and for free apps it becomes 0.83 (p=0.003), while the Spearman pho becomes 0.96 (p=0.000) for both free and non-free apps. Therefore without this outlier the correlation for free apps seems to be more exponential than linear (so higher rated apps tend to be exponentially more popular), whereas for non-free apps the relationship appears to be linear whether or not we exclude the five star outlier. It is impossible to know exactly why the rating of five stars should be peculiar in this way. It would be tempting to speculate that there is something less reliable about five star ratings (particularly for apps that have only this top rating), even perhaps that a larger proportion of five star ratings might be suspicious than those at other rating levels. After all, if a developer were to rate their own app (or recruit others to do so) would that developer not wish for the highest possible rating? However, since correlation, on its own, cannot reveal causality, we leave this as an open question for further studies. Perhaps when we better understand how to assess the likely provenance of reviews, the question as to why five star ratings are peculiar can be answered more fully.

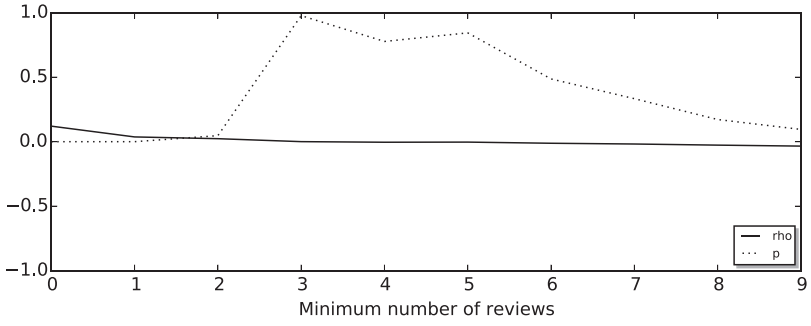
Similar observations about correlations between rating and popularity hold for the mined features. That is, there appears to be little correlation involving price (see Fig. 8(a) and(b)), while there is a strong (and apparently generally linear) correlation between rating and popularity: more highly rated features tend to be more popular (they have a lower rank of downloads). The correlation is far from perfect, overall, but the general linear trend is visually quite evident in Fig. 8(c) and (d)).

To provide a more quantitative assessment of these correlations for features and apps, both within each category and overall, we report in Tables 4 – 7 all the Pearson and Spearman correlation values. Figs. 9 and 10 show the Spearman's Rank and Pearson's correlation (solid line) and their significance (dashed line) obtained by grouping non-free and free apps, respectively, by their minimum number of reviews.[6] In particular, we set the minimum number of reviews to range from 0 to 9 and plot the x axis as minimum

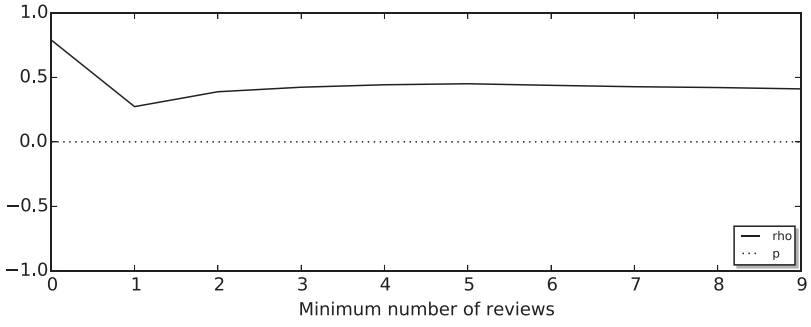
[6] Space does not permit us to include all graphs of this form per category. However, the 38 graphs (one per category) are available at the



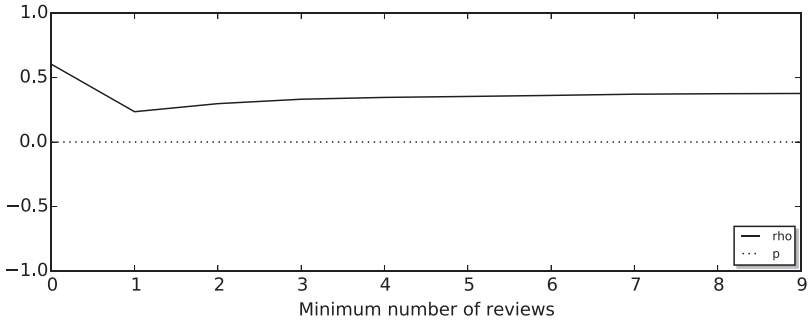
(a) Price vs. Rating (non-free apps)



(b) Price vs. Rank of Downloads (non-free apps)

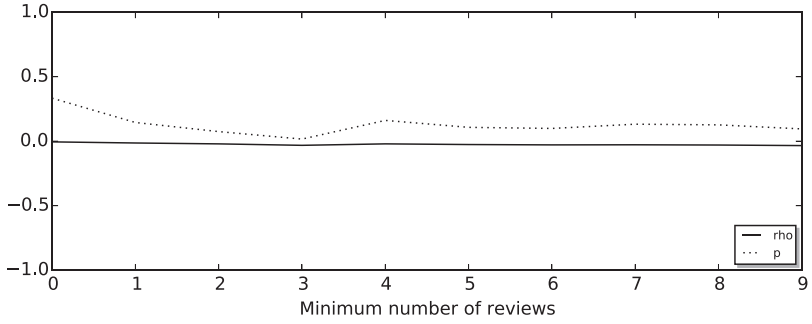


(c) Rating vs. Rank of Downloads (non-free apps)

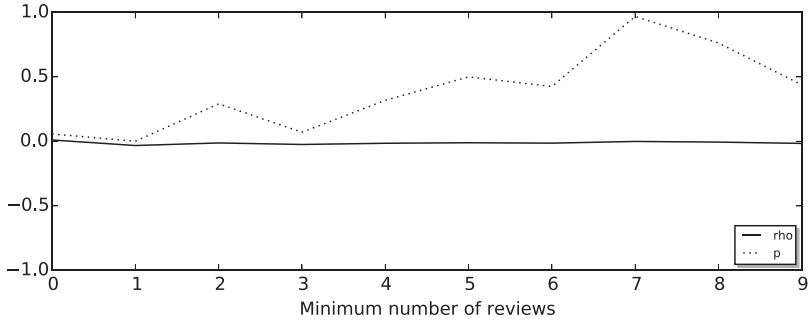


(d) Rating vs. Rank of Downloads (free apps)

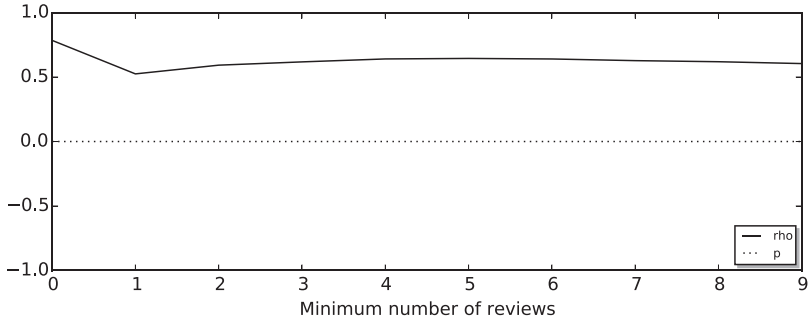
Fig. 9. RQ1-3. Correlations Graphs: The figures show the Spearman Rank correlation values (solid line) and their significance (dashed line) obtained by grouping the apps by their minimum number of reviews.



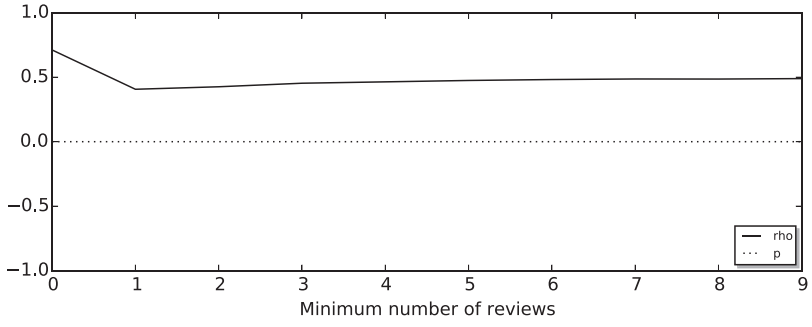
(a) Price vs. Rating (non-free apps)



(b) Price vs. Rank of Downloads (non-free apps)



(c) Rating vs. Rank of Downloads (non-free apps)



(d) Rating vs. Rank of Downloads (free apps)

Fig. 10. RQ1-3. Correlations Graphs: The figures show the Pearson correlation values (solid line) and their significance (dashed line) obtained by grouping the apps by their minimum number of reviews.

Table 4
Spearman Correlation Results for RQ1-3 at the App Level: The first 9 columns present the Spearman Rank correlation values computed for non-free apps, while the final 3 present the values we computed for free apps. We present the results obtained for each subset of apps having at least 0, 1, and 2 reviews. In all of these columns, the single letter labels stand for (P)rice, (R)ating, and (D)ownloads.

Name of Categories	Non-Free Apps									Free Apps		
	MinReviews=0			MinReviews=1			MinReviews=2			MinReviews=0	MinReviews=1	MinReviews=2
	PR	PD	RD	PR	PD	RD	PR	PD	RD	RD	RD	RD
Business	0.02	0.03	0.83	−0.04	0.03	0.40	−0.01	0.07	0.52	0.81	0.40	0.46
Education	−0.10	−0.05	0.83	−0.06	0.04	0.52	−0.08	0.11	0.64	0.80	0.43	0.53
Entertainment	−0.17	−0.21	0.81	0.12	0.00	0.37	0.02	0.05	0.57	0.46	0.27	0.31
Finance	0.33	0.43	0.81	0.09	0.27	0.35	0.28	0.38	0.46	0.71	0.14	0.25
Games	−0.10	−0.01	0.76	−0.20	−0.03	0.42	−0.17	−0.05	0.49	0.27	0.16	0.23
Health& Wellness	−0.28	−0.26	0.85	−0.15	−0.06	0.52	−0.15	0.03	0.54	0.75	0.30	0.38
IM & Social Networking	−0.21	0.02	0.63	−0.30	0.10	0.28	−0.30	0.08	0.41	0.50	0.32	0.35
Maps & Navigation	−0.06	0.01	0.78	−0.04	0.15	0.45	−0.12	0.19	0.50	0.56	0.34	0.43
Music & Audio	0.42	0.33	0.76	−0.08	0.11	0.44	−0.20	−0.05	0.52	0.52	0.05	0.05
News	0.07	0.16	0.79	0.06	0.32	0.33	0.20	0.31	0.40	0.73	0.31	0.39
Photo& Video	0.02	0.06	0.82	−0.11	0.09	0.21	−0.09	−0.01	0.50	0.48	0.43	0.45
Productivity	0.01	0.08	0.73	0.02	0.14	0.35	0.00	0.12	0.37	0.58	0.37	0.45
Reference & eBooks	0.09	0.13	0.32	0.01	0.03	0.60	0.00	0.02	0.58	0.83	0.57	0.53
Shopping	0.26	0.21	0.67	0.12	0.03	0.28	−0.08	0.19	0.16	0.59	0.31	0.30
Sports & Recreation	−0.10	−0.02	0.77	−0.14	0.04	0.31	0.13	0.23	0.56	0.31	0.04	0.17
Themes	0.16	0.15	0.81	0.04	0.05	0.07	0.01	−0.02	0.17	0.04	−0.09	−0.09
Travel	0.04	−0.02	0.75	0.21	0.22	0.85	0.34	0.30	0.87	0.54	0.05	0.14
Utilities	−0.10	−0.03	0.77	−0.11	0.05	0.27	−0.12	0.00	0.45	0.55	0.29	0.37
Weather	0.07	0.12	0.54	0.19	0.21	−0.04	0.25	0.16	0.10	0.66	0.58	0.55
All	0.10	0.12	0.79	0.02	0.04	0.27	0.01	0.02	0.39	0.60	0.23	0.30
Some correlation	0	0	18	0	0	4	0	0	10	13	2	3

number of reviews, and the *y* axis as the correlation (rho) value and *p*-value of the correlation test.

From Figs. 9 and 10 we can observe that there is an atypically higher correlation coefficient reported for the case where we include all apps (that is, we include all apps with zero or more ratings in the analysis) for both the Spearman and Pearson tests. This stronger correlation could be an artefact of the many apps with zero ratings; since these rating values are tied, by definition, this may tend to (artificially) inflate the correlation coefficient.

This was our motivation for additionally reporting on higher thresholds for the number of ratings required in order for the app to be included in the correlation analysis. As we move rightwards in these graphs, we reduce the number of apps considered, but increase the number of ratings required per app in order for the app to be included in the analysis. This reflects a trade off in the quality and quantity of evidence for customer rating.

For correlation coefficients close to zero (no rank correlation) the amount of evidence needed is generally higher, in order for a reliable assessment of the correlation coefficient (rho) value. This is reflected by the change in the *p* value, which indicates insufficient evidence after $x = 2$ in the case of Figs. 9(b) and 10(b). In order to be cautiously conservative about the correlations reported, we therefore based our claims that rest on qualitative analysis of correlation coefficients on analysis with 'rating filters' only up to a maximum of 2 (that is all apps with two or more ratings).

This quantitative analysis of Spearman and Pearson correlation coefficients can be found in Tables 4 and 6, respectively. The decision as to when a correlation coefficient is sufficiently high that it reflects a degree of association is debatable. An absolute value for a correlation coefficient above 0.5 (with an associated *p* value less than 0.05) is, however, surely unlikely to arise by chance. Therefore, we treat this as a conservatively safe threshold above which we deem *some* correlation to exist in each case.[7]

With this threshold in mind, we counted the number of correlation coefficients in each category, the absolute value of which was 0.5 or above. This count is reported in the final row of each table. As can be seen (from the columns labeled 'RD') in these tables, there are clearly many app and feature categories where there is a correlation between the rating and popularity (Rank of Downloads).

In particular, when the 'minimum number of reviews' threshold is set to zero (its most inclusive value), there is a correlation between rating and popularity for all but one category and in all but one case (18 out of 19) in three of the tables (and all cases for the fourth, concerning linear feature correlations). Of course, this value could be unduly influenced by tied ratings data (those apps with zero ratings). However, many strong correlations exist when we filter out all zero rated apps (in columns labeled 'MinReviews=1' and 'MinReviews=2').

Perhaps somewhat surprisingly, we found little evidence for a correlation between *either* the price of an app and its rating, *or* between the price and the rank of downloads of an app. This finding applies to both the app store as a whole and to almost all of the categories within it. This would suggest that, despite the plethora of apps and fierce competition, customers of non-free apps may not be as price sensitive as one might have thought; they tend to accord neither higher nor lower rating scores to more expensive non-free apps.

Finally we observe that the Pearson's correlations between Rating and Rank of Downloads are stronger than the Spearman's ones suggesting that the relationship between these two variables has a more linear character than a monotonic one.

The correlations between rating and rank of downloads observed for apps can be also observed for the features we extracted, while no correlation has been found between feature's price and rating. As can be seen from Table 5, we found strong correlations

paper's companion websitehttp://www0.cs.ucl.ac.uk/staff/F.Sarro/projects/UCLappA/resources/CorrelationGraphs.pdf.

[7] Let us recall that correlation coefficients whose magnitude are between 0.9 and 1.0 indicate variables which can be considered very highly correlated. Correlation coefficients whose magnitude are between 0.7 and 0.9 indicate variables which can be considered highly correlated. Correlation coefficients whose magnitude are between 0.5 and 0.7 indicate variables which can be considered moderately correlated. Correlation coefficients whose magnitude are between 0.3 and 0.5 indicate variables which have a low correlation. Correlation coefficients whose magnitude are less than 0.3 have little if any correlation.

Table 5
Spearman Correlation Results for RQ1-3 at the Feature Level: The first 9 columns present the Spearman Rank correlation values we computed for non-free features, while the final 3 present the values we computed for free features. We present the results obtained for each subset of apps having at least 0, 1, and 2 reviews. In all of these columns, the single letter labels stand for (P)rice, (R)ating, and (D)ownloads.

Name of Categories	Non-Free Features									Free Features		
	MinReviews=0			MinReviews=1			MinReviews=2			MinReviews=0	MinReviews=1	MinReviews=2
	PR	PD	RD	PR	PD	RD	PR	PD	RD	RD	RD	RD
Business	−0.36	−0.38	0.78	−0.32	−0.22	0.61	−0.10	−0.08	0.72	0.85	0.50	0.56
Education	−0.16	−0.27	0.87	0.28	−0.25	0.03	−0.05	−0.63	0.21	0.68	0.31	0.19
Entertainment	−0.30	0.05	0.57	−0.23	0.37	−0.07	0.00	0.31	0.09	0.18	0.03	0.04
Finance	0.12	0.28	0.46	−0.01	0.16	0.31	0.29	0.04	0.09	0.64	0.32	0.32
Games	−0.20	0.10	0.77	−0.15	0.16	0.25	−0.12	−0.13	0.59	0.36	0.21	0.09
Health& Wellness	−0.40	−0.50	0.93	−0.25	−0.37	0.68	−0.35	−0.10	0.69	0.87	0.63	0.57
IM & Social Networking	−0.36	−0.19	0.57	−0.24	−0.15	0.31	−0.21	−0.19	0.44	0.59	0.45	0.42
Maps & Navigation	0.48	0.42	0.90	0.35	0.28	0.79	0.28	0.29	0.88	0.58	0.23	0.28
Music & Audio	−0.05	0.00	0.74	−0.15	−0.12	0.49	−0.18	−0.01	0.65	0.13	−0.14	−0.24
News	0.12	0.05	0.75	0.17	−0.43	0.35	−0.05	−0.77	0.40	0.78	0.35	0.57
Photo& Video	−0.37	−0.30	0.80	−0.47	−0.26	0.47	−0.55	−0.47	0.60	0.28	0.28	0.28
Productivity	0.24	0.23	0.86	0.19	0.26	0.37	0.09	0.12	0.41	0.76	0.68	0.58
Reference & eBooks	−0.02	−0.39	0.74	0.49	−0.03	0.31	0.58	−0.28	0.30	0.33	0.11	0.24
Shopping	−0.17	−0.56	0.73	−0.20	−0.70	0.52	0.27	−0.59	0.01	0.78	0.75	0.76
Sports & Recreation	0.25	0.25	0.79	0.00	0.08	−0.02	0.02	0.37	0.26	0.35	−0.18	−0.09
Themes	0.32	0.00	0.80	0.15	−0.12	0.19	0.07	−0.06	0.32	0.35	0.05	−0.15
Travel	0.34	0.15	0.82	0.27	0.02	0.55	0.24	−0.06	0.51	0.64	0.12	0.23
Utilities	0.03	0.06	0.87	−0.26	0.01	0.56	−0.29	−0.18	0.68	0.73	0.55	0.61
Weather	0.11	−0.03	0.67	−0.01	−0.22	0.67	0.01	−0.28	0.72	0.60	0.60	0.60
All	−0.17	−0.19	0.81	−0.10	−0.21	0.37	−0.14	−0.23	0.44	0.64	0.33	0.33
Some correlation	0	2	18	0	1	7	2	3	9	12	6	7

Table 6
Pearson Correlation Results for RQ1-3 at the App Level: The first 9 columns present the Pearson correlation values computed for non-free apps, while the final 3 present the values we computed for free features. We present the results obtained for each subset of apps having at least 0, 1, and 2 reviews. In all of these columns, the single letter labels stand for (P)rice, (R)ating, and (D)ownloads.

Name of Categories	Non-Free Apps									Free Apps		
	MinReviews=0			MinReviews=1			MinReviews=2			MinReviews=0	MinReviews=1	MinReviews=2
	PR	PD	RD	PR	PD	RD	PR	PD	RD	RD	RD	RD
Business	−0.09	−0.07	0.82	−0.13	−0.15	0.62	−0.05	−0.09	0.71	0.75	0.52	0.51
Education	−0.07	−0.08	0.76	0.09	0.05	0.55	0.06	0.13	0.72	0.78	0.54	0.55
Entertainment	−0.07	−0.17	0.77	0.19	−0.06	0.53	0.09	−0.02	0.65	0.61	0.38	0.38
Finance	0.17	0.13	0.83	0.11	0.04	0.61	0.12	0.26	0.64	0.73	0.3	0.39
Games	−0.09	−0.01	0.77	−0.18	−0.05	0.55	−0.15	−0.07	0.59	0.47	0.22	0.3
Health& Wellness	−0.27	−0.28	0.8	−0.06	−0.14	0.61	−0.12	−0.14	0.65	0.73	0.44	0.5
IM & Social Networking	−0.16	−0.18	0.74	−0.17	−0.15	0.51	−0.25	−0.26	0.6	0.63	0.42	0.43
Maps & Navigation	0.02	0.01	0.8	0	0.04	0.63	0	0.17	0.67	0.69	0.46	0.55
Music & Audio	0.23	0.26	0.8	−0.09	0.07	0.7	−0.22	−0.03	0.72	0.65	0.31	0.27
News	0.01	0.03	0.73	−0.07	−0.07	0.44	0.14	0.1	0.46	0.74	0.45	0.5
Photo& Video	0.1	0.13	0.85	−0.14	−0.09	0.44	−0.24	−0.22	0.61	0.54	0.46	0.5
Productivity	−0.03	−0.02	0.82	0.04	0.03	0.56	0.02	0.06	0.56	0.69	0.52	0.59
Reference & eBooks	0.1	0.15	0.42	0.05	0.01	0.62	0.05	0.01	0.65	0.82	0.69	0.65
Shopping	0.09	0.06	0.77	0.05	0.03	0.48	−0.13	0.17	0.35	0.65	0.44	0.39
Sports & Recreation	0.05	0.1	0.75	−0.04	0.03	0.54	0.07	0.14	0.64	0.59	0.23	0.29
Themes	0.12	0.1	0.81	0.04	0.02	0.42	0.03	−0.01	0.45	0.54	0.13	0.05
Travel	0.16	0.05	0.69	0.3	0.18	0.7	0.37	0.24	0.78	0.66	0.23	0.2
Utilities	−0.05	−0.09	0.82	0.02	−0.04	0.52	−0.06	−0.03	0.61	0.71	0.44	0.49
Weather	0.1	0.16	0.69	0.17	0.17	0.17	0.22	0.13	0.31	0.77	0.72	0.73
All	−0.01	0.01	0.78	−0.01	−0.03	0.52	−0.02	−0.01	0.59	0.71	0.41	0.43
Some correlation	0	0	18	0	0	14	0	0	15	18	5	9

between the rating and the rank of downloads for the features (as well as the apps) in almost every category (and also within the app store as a whole) when all the apps are considered (i.e., *MinReviews* = 0). As we become more restrictive, the correlation values decrease in many categories for the same reason observed at the app level. Finally, the correlations observed for free features are, in general, lower than those observed for non-free features, perhaps suggesting that free features might be popular regardless of their rating.

In general, our results show that there is a correlation between customer rating and the rank of feature downloads and there is no correlation between feature price and rank of feature downloads, nor between price and rating, replicating RQs1-3 at the feature level.

Thus, in answer to RQ1-3: Our results show that there is a correlation between customer rating and the rank of app downloads for apps and the features extracted from them for both free and non-free apps and features. However, there is very little evidence for any correlation between price and either rating or popularity.

Table 7
Pearson Correlation Results for RQ1-3 at the Feature Level: The first 9 columns present the Pearson correlation values computed for non-free features, while the final 3 present the values we computed for free features. We present the results obtained for each subset of apps having at least 0, 1, and 2 reviews. In all of these columns, the single letter labels stand for (P)rice, (R)ating, and (D)ownloads.

Name of categories	Non-Free features									Free features		
	MinReviews=0			MinReviews=1			MinReviews=2			MinReviews=0	MinReviews=1	MinReviews=2
	PR	PD	RD	PR	PD	RD	PR	PD	RD	RD	RD	RD
Business	−0.41	−0.48	0.76	−0.51	−0.65	0.67	−0.36	−0.60	0.66	0.85	0.71	0.58
Education	−0.08	−0.20	0.84	0.28	−0.22	−0.02	−0.13	−0.45	0.23	0.72	0.48	−0.02
Entertainment	−0.42	−0.27	0.74	0.01	−0.06	0.37	0.11	−0.07	0.32	0.28	0.14	0.19
Finance	0.15	0.30	0.73	−0.05	0.13	0.62	0.14	0.09	0.20	0.67	0.38	0.41
Games	0.05	0.21	0.76	−0.09	0.18	0.13	−0.32	−0.31	0.66	0.26	0.23	0.00
Health& Wellness	−0.36	−0.51	0.89	−0.04	−0.39	0.71	−0.43	−0.30	0.72	0.91	0.68	0.60
IM & Social Networking	−0.52	−0.39	0.67	−0.18	−0.04	0.30	−0.14	−0.10	0.48	0.56	0.41	0.36
Maps & Navigation	0.38	0.21	0.88	0.30	0.12	0.85	0.27	0.27	0.92	0.61	0.51	0.53
Music & Audio	−0.01	0.05	0.75	−0.12	−0.11	0.43	−0.22	0.10	0.57	0.44	0.10	0.00
News	0.10	0.16	0.73	−0.07	−0.24	0.49	−0.25	−0.77	0.54	0.72	0.36	0.60
Photo& Video	−0.30	−0.17	0.85	−0.43	−0.15	0.49	−0.59	−0.35	0.51	0.33	0.32	0.32
Productivity	0.35	0.29	0.89	0.25	0.23	0.42	0.21	0.13	0.54	0.78	0.66	0.48
Reference & eBooks	−0.19	−0.46	0.88	0.53	−0.19	0.28	0.50	−0.29	0.45	0.56	0.01	0.19
Shopping	−0.13	−0.50	0.79	−0.22	−0.67	0.71	0.39	−0.41	0.17	0.77	0.76	0.79
Sports & Recreation	0.00	0.11	0.78	−0.35	−0.25	0.23	−0.30	0.15	0.33	0.68	−0.19	−0.02
Themes	−0.02	−0.24	0.83	0.02	−0.12	0.27	−0.03	0.21	0.19	0.21	−0.01	−0.21
Travel	0.35	0.16	0.70	0.31	0.13	0.55	0.29	0.09	0.55	0.64	0.15	0.31
Utilities	0.02	−0.05	0.88	−0.06	−0.01	0.51	−0.23	−0.15	0.67	0.75	0.70	0.72
Weather	0.17	−0.07	0.70	0.09	−0.18	0.65	0.11	−0.23	0.69	0.82	0.79	0.79
All	−0.21	−0.26	0.83	−0.07	−0.27	0.52	−0.08	−0.27	0.58	0.75	0.47	0.45
Some correlation	1	2	19	2	2	8	2	2	12	14	7	7

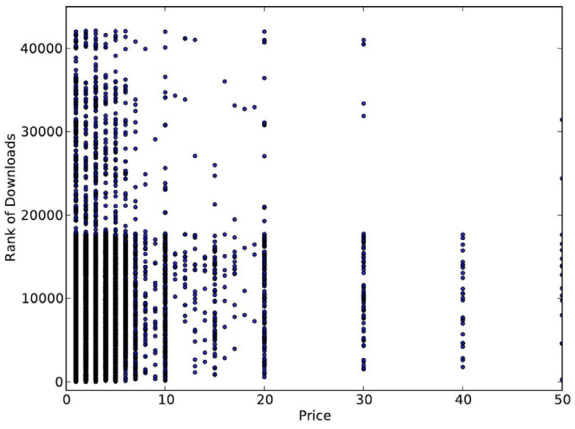
4.3. RQ4. Is there a stronger correlation involving Price when we 'zoom in' on specific ranges of price or between price and number of features or shared features in an app?

We found no evidence that price is correlated to either ratings or to popularity, neither for apps nor the features we extracted from them (see RQ1 and RQ2). However, we questioned the possibility that, though there is no overall correlation involving price, there may nevertheless, be correlations in sections of the data (perhaps for specific price ranges). We therefore further analysed the relationship between price/rating and price/download by zooming into the scatterplots shown in Fig. 7. Moreover, we analysed whether there is any relationship between app prices and their numbers of features or numbers of shared features.

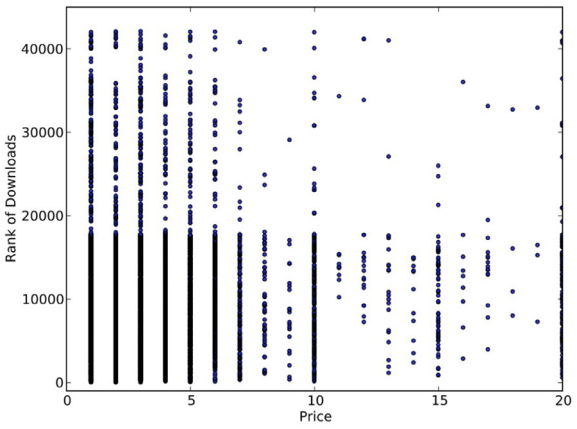
From Figs. 11 and 12 we can observe some interesting patterns:

1. Prices tend to be lower than $5.00 for most apps, but there are frequency peaks at 'round number' prices, such as $10 and $20 (see Fig. 11(b) and (a)). However, if we consider the median values of rank of downloads (see Fig. 11(c)), it is clear that there is no linear relationship between price and rank of downloads (i.e., Pearson rho = 0.165, p-value=0.385), while we can observe a mild rank correlation (i.e., Spearman rho=0.41, p-value=0.027).
2. The lower priced apps tend to have a higher rating (see Fig. 11(e) and (d)). From the scatter plots we do see some evidence that the ratings accorded to apps priced below $5.00 are slightly higher than those accorded to more expensive apps, but the correlation coefficient is extremely low: the Spearman rho=0.051, with a p-value = 0.000, while the Pearson rho=0.046, with a p-value = 0.000. Also, it should be noted that at this lower end of the price spectrum there are many tied values (e.g. all apps with price $0.99) and this can artificially inflate the correlation values reported. If we look at the median values (see Fig. 11(f)), we cannot find any significant correlations between price and rating (i.e., Pearson rho=−0.099, p-value=0.602 and Spearman rho=−0.159, p-value=0.401).
3. The more expensive apps tend to have more features (see Fig. 12(a) and (b)) and shared features (see Fig. 12(c) and (d)). We found moderate correlations between price and median number of features (Pearson rho=0.46, p-value=0.007, Spearman rho=0.46, p-value=0.006) and between price and median number of shared features (Pearson rho = 0.46, p-value=0.007, Spearman rho = 0.46, p-value=0.006) when considering all apps (i.e., including those having zero features).
The linear correlations between price and median number of features/shared features become stronger, while the Spearman's ones decreased dramatically (and, perhaps more importantly, lose their significance) when we consider only those apps having at least one feature (Pearson rho=0.54, p-value=0.001, Spearman rho=0.023, p-value=0.899) or at least one feature in common with other apps (Pearson rho = 0.54, p-value=0.001, Spearman rho=0.023, p-value=0.899).
This finding suggests that the apparent rank correlation for all apps (including those with no features at all) is a product of ties (the zero-featured apps have a tied number of features). However, the linear correlation is the one that is stronger so we conclude that there is overall evidence of a mild linear price to number-of-features correlation.
4. Though there is only the weakest evidence for any correlation between an app's price and either its rank of downloads or rating, there is stronger evidence for correlations between a feature's price and its rating (and also its rank of downloads). We investigated this further by computing correlation coefficients for the median rating and for the median rank of downloads per price point for all non-free features (see Fig. 12(e) and (f)). For ratings, we found evidence of an inverse correlation between price and rating for both Pearson (rho=−0.537, p-value=0.000) and Spearman (rho=−0.559, p-value=0.000) correlations. For rank of downloads, the evidence was less strong: Pearson (rho=−0.40, p-value=0.000) and Spearman (rho=−0.422, p-value=0.000).

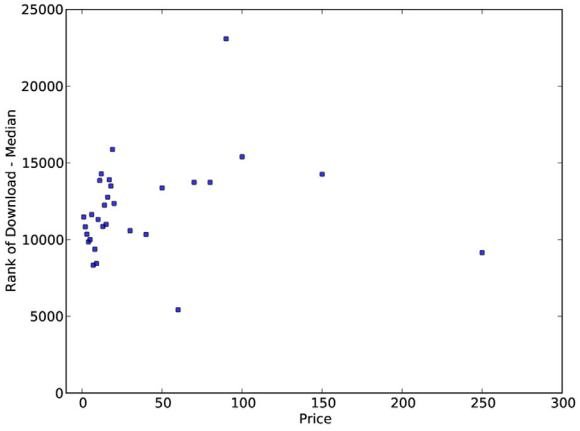
It is interesting to note that the correlation one might expect (higher prices are less likely to be favoured by users, surely?) is present with stronger evidence for the features than for the apps from which we extract these features. This could be because there are many more different price points and rating values for features



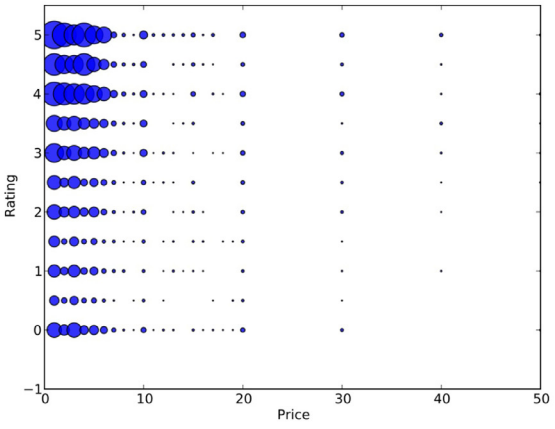
(a) Price vs. Rank of Downloads



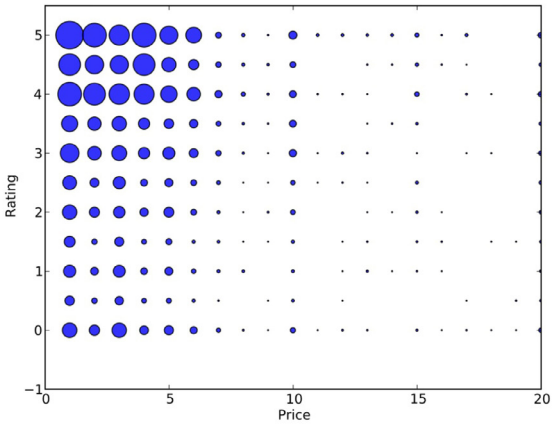
(b) Price vs. Rank of Downloads (zoom in)



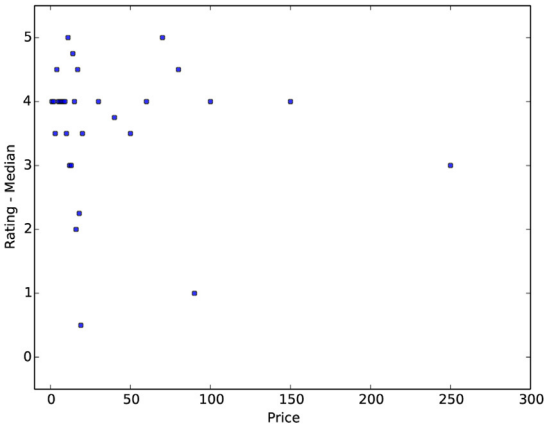
(c) Price vs. Rank of Downloads (median)



(d) Price vs. Rating

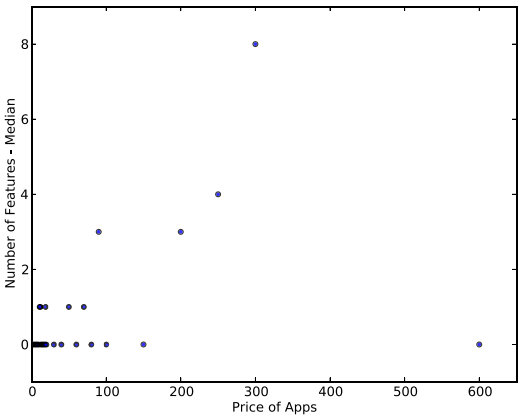


(e) Price vs. Rating (zoom in)

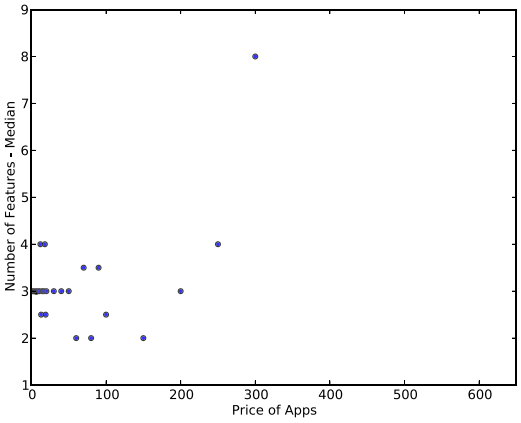


(f) Price vs. Rating (median)

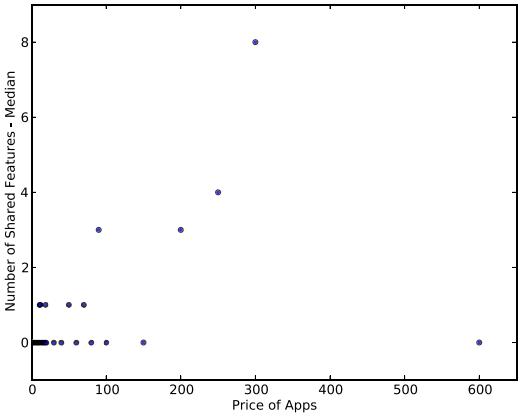
Fig. 11. RQ4: Scatterplots of Price vs. Rank of Downloads and Rating at different levels of granularity.



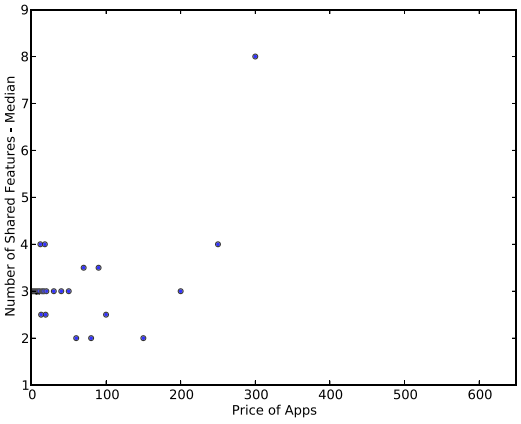
(a) Price vs. Feature (median), $f \geq 0$



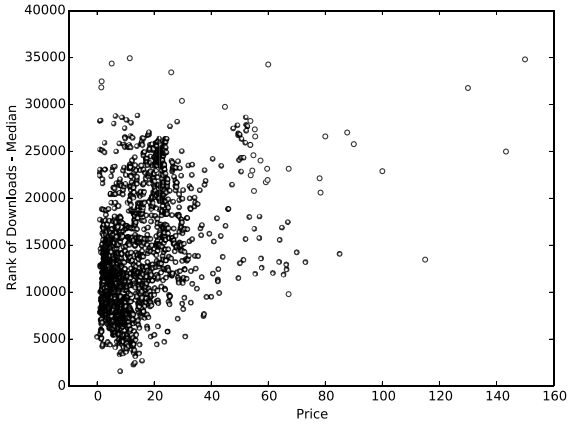
(b) Price vs. Feature (median), $f > 0$



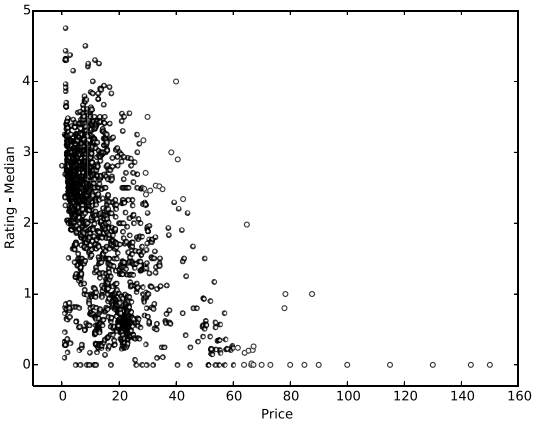
(c) Price vs. Shared Feature (median), $f \geq 0$



(d) Price vs. Shared Feature (median), $f > 0$



(e) Rank of Downloads - Median per Price point (features)



(f) Rating - Median per Price point (features)

Fig. 12. RQ4: Scatterplots of Price vs. Features and Shared Feature at different levels of granularity.

(since feature properties are computed as averages over the apps that share the features). However, the strong correlation found is further evidence that the features we extract carry some meaning and that this meaning could be useful to developers.

In answer to RQ4 for apps, we found that there is a moderate correlation between apps price and median number of (shared) features. The higher the price the more features are claimed to be provided. However, the answer for features provides stronger evidence of an inverse correlation between price and rating; more expensive features tend to be less highly ranked.

4.4. RQ4.1: Is there any difference in Rating and Popularity for free apps compared to non-free apps?

From Table 3 we can observe that, on average, free apps have a lower rank of downloads than non-free apps (suggesting that, in general, free apps are more popular). We found that this difference is statistically significant according to the non-parametric Mann-Whitney 'U' Test (p-value < 0.001), with a notable effect size (the Vargha-Delaney normalised non-parametric effect size $\hat{A}_{12}$ is 0.76). The same observation holds for free features (i.e., free features are more popular than non-free ones, p-value < 0.001 and $\hat{A}_{12} = 0.70$).

From Table 3 we also observe that free apps provide the users slightly fewer features (see Table 3) on average, than their non-free counterparts. However, we found that this difference is not statistically significant according to the non-parametric Mann-Whitney 'U' Test (p-value$= 0.847$, $\hat{A}_{12} = 0.50$) .

As for the rating, we can observe that the most highly rated non-free apps reside in the categories 'IM & Social Networking', 'Weather' and 'Productivity', while 'Themes' and 'Games' contain the most highly rated free apps. In general, we observe that free apps enjoy a higher rating, on average, compared to the non-free apps that reside in the same category (see Table 3). This difference is statistically significant (p-value < 0.001), according to the non-parametric Mann-Whitney 'U' Test and has a reasonably large effect size ($\hat{A}_{12}$ is $= 0.68$).

In answer to RQ4.1, we find that there is strong evidence that the free apps are, in general, more popular than non-free apps and that they also enjoy higher ratings.

5. Threats to Validity

In this section we discuss the validity of our study based on three types of threats, namely *construct, conclusion*, and *external* validity.

Construct validity concerns the methodology employed to construct the experiment. Since our data is extracted from the Blackberry App Store, we are relying on the maintainers of this store for the reliability of our raw data. Therefore inaccuracies and imprecision in these data may have affected some of our derived data. In order to protect against possibly incorrect conclusions that may be drawn from analysing such data, we have been careful to base all of our primary observations on analyses based on large sets of data. By focusing on such 'macro level' statistical observations (rather than fine-grained detailed observations), we hope that our findings will prove to be robust in the presence of any inaccuracies and imprecision in the raw data.

Conclusion validity threat concerns issues that may affect the ability to draw a correct conclusion. To mitigate this threat, we carefully applied the statistical tests, verifying all the assumptions each inferential test requires concerning the distributions to which it is applied.

Our approach to external threats is relatively standard for the empirical software engineering literature. That is, our data covers a set of categories that have a degree of diversity in application type and size, however we cannot claim that our results generalise beyond the subjects studied. Our results are based on a mobile app store (though there is no reason to assume that they may not apply to other app stores). However, the results presented here for Blackberry concern an app store that is worth several hundreds of millions of dollars, so the potential monetary impact of the findings remains considerable. Moreover, we described in detail the approach proposed and the empirical methodology we followed in order to allow other researcher to replicate and extend our work. A potential threat to generalisability lies in our extraction of feature information from descriptions. We mitigate this threat by extracting the features from a large and varied collection of app descriptions, and clarifying that it is clearly a constraint of our method (and of most NLP-based approaches [3]). Naturally, we do not claim that these extracted features include all the real features of the app. Indeed, we do not *even* claim that *any* of the features we extract can be found in the app (precision) neither that we extract all the features provided by an app (completeness). Rather, we claim that there is evidence that what we have extracted tends to be meaningful feature descriptions (as indicated by our human sanity check) and that they denote features *claimed* to be included in the apps (according to the developers' own descriptions). Great care is required in extending our findings from 'claimed features' to features that are truly available to users of the app. Such extrapolation of our findings is not valid unless future work demonstrates a strong correlation between claimed and actual features.

6. Related Work

There are several perspectives which have been studied in App Store Analysis. A comprehensive literature review is provided by Martin et al. [6]. In this section, we will focus on those studies that compare mobile apps with traditional software (Section 6.1), studies that investigate app descriptions and their features, (Section 6.2), and work investigating other apps' attributes (Section 4.2).

6.1. Comparison with Traditional Software

The goal of App Store Analysis is to combine technical data with non-technical data such as user and business data to understand their inter-relationships [1,19,20]. The number and granularity of the software products considered differs from previous work on mining non-app software: previous work typically uses a white box analysis of multiple applications [21] of software products of (sometimes) very large size [22]. By contrast, to mine app stores, we can use white box techniques where the source code of apps is available. However, we may also use a black box analysis of the apps, where source code is unavailable. As we have shown in this paper, technical information can be extracted from sources other than the code of the app itself. We are also likely to consider potentially many more software products, but of perhaps smaller size, at least for the apps available at the time of writing (they may grow in size and complexity in future, as all software generally tends to do [23]).

Several other authors have also commented on general properties of App Store Analysis and its relationship to traditional software repository mining. For example, Syer et al. [24] sought to understand the differences in characteristics between apps and more conventional applications, drawing parallels between apps and UNIX utilities, while Nagappan et al. [25] and Menzies [26] discussed challenges and opportunities in app analysis.

Minelli and Lanza [27] also compared apps with traditional software systems, finding that apps are smaller and simpler (consisting of approximately 5.6k Lines of code, on average). However, they claimed (and we agree) that this may be a transient effect, due to disappear as apps become larger and more complex.

Other authors have also investigated the relationship between the functionalities offered by mobile apps and their size/ development effort. Sethumadhavan [28] was the first to discuss the application of Function Point Analysis (FPA) to Android applications, pointing out that compared with traditional desktop applications, mobile apps contain limited functionality, and often functionality is merely a wrapper to system functionality. Subsequent studies showed how both FPA [29,30] and COSMIC [31–33] can be used to measure the functional size of mobile apps.

Ruiz et al. [8] analysed Android code reuse, finding it to be prevalent compared to non-Android open source software. They also found that developers reuse software through inheritance, libraries and frameworks (a result we partly replicated for the Blackberry world app store in Section 4.1).

In order to gain an understanding of the main challenges app developers face in practice, Joorabchi et al. [34] survey 188 developers from the mobile development community. The outcome highlighted that developing apps across multiple platforms, lack of robust monitoring, analysis, and testing tools, and emulators that are slow or miss many features of mobile devices, are some of the challenges currently faced by mobile app developers.

6.2. App Descriptions and their Features

Harman et al. [1] were the first to argue that App Store Analysis can be used to understand the relationships between technical, business and social aspects of app stores. They were also the first to propose the incorporation of technical information (such as feature information, mined from app descriptions) as part of this analysis process. The present paper extends their initial analysis of non-free Blackberry apps [1], to consider both free and non-free apps, and the correlations between their claimed features, rating, popularity, and price in more detail.

Subsequently features extracted from app descriptions have been used to cluster apps on the user device [35] or in existing app stores [13,35,36]. Lulu and Kuflik [35] cluster apps to help users retrieve the apps they have installed on their device.Their approach is also based on information extracted from the app description, but augmented by content from 'professional blogs'. Kim et al. [36] mine 100830 apps from Apple App Store and extract feature keywords from their descriptions using natural language processing in order to re-categorise these apps. More recently, Al-Subaihin et al. [13] investigate a clustering method based on the similarity between features extracted from mobile apps' descriptions by using the approach proposed herein. The approach was empirically validated using 17877 apps from Google Play and Blackberry app stores. The internal cluster quality they found is larger than the one the current app store categories exhibit. Additionally, they found a positive correlation between the similarity score of their technique and the similarity score assigned by human judgment.

App descriptions have been also used to support requirement analysis. Sarro et al. [12] have recently proposed a theoretical characterisation of feature lifecycles in app stores and used the approach proposed herein to extract features from the descriptions of non-free apps available in the Blackberry and Samsung app stores. The empirical analysis of the migratory and non-migratory behaviours of 4,053 non-free features reveal that, in both stores, intransitive features (those that neither migrate nor die out) exhibit significantly different behaviours with regard to important properties, such as their price. Further correlation analysis also highlights differences between trends relating price, rating, and popularity. These results indicate that feature lifecycle analysis can yield insights that may also help developers to understand feature behaviours and attribute relationships.

Previous work have also exploited app descriptions in order to detect malicious behaviours. Pandita et al. [37] introduce the tool WHYPER that compares the permissions requested by the app and the app description by using First Order Logic. This allows them to highlight apps with suspect descriptions. Suspicion arises when mismatches are found between an app's technical declaration of permissions sought and its public declaration of features it offers. Yang et al. [38] tackle the same problem by introducing an approach named APPIC to compare features extracted from descriptions (using topic modeling) with the permissions declared for an Android app. Gorla et al. [39] use app descriptions and API calls as a convenient way to understand the semantic behaviour of a large number of apps, the source code which they mine. They show how anomalous API calls can be used to detect aberrant or otherwise suspicious behaviour.

6.3. Investigating Apps' Attributes

Taba et al. [40] studied 1292 free Android apps from 8 app categories, reporting that users award significantly higher ratings to apps with simpler user interfaces. Syer et al. [41] reported a positive correlation between the number of defects found in Android apps and platform dependence assessed in terms of API calls. Angeren et al. [42] investigated dependence between various attributes of apps in the App Store itself to give a perspective on the App Store as a software ecosystem [43]. Ruiz et al. [44,45] studied the effect of ad-libraries on rating; and Avdiienko et al. [46] used extracted data flow information to detect potentially malicious apps through abnormal data flow. Linares–Vasquez et al. [47] analysed how the fault- and change-proneness of APIs used by 7097 free Android apps related to their success (i.e., the mean rating provided by the users to those apps). The study revealed that making heavy use of fault- and change-prone APIs can negatively impact the success of these apps. This analysis has been extended by Bavota et al. [48] by surveying 45 professional Android developers. Most of the developers interviewed confirmed that in their experience they have observed a direct relationship between problems due to the adopted APIs and the users' ratings.

McIlroy et al. [49] studied the frequency of updates of 10,713 mobile apps aiming at providing information regarding the update strategies employed by the top 400 mobile apps contained in the Google Store in 2014. They found that about the 1% of the apps studied are updated at a very frequent rate (i.e., more than one update per week) and 14% of the studied apps are updated on a bi-weekly basis (or more frequently). Moreover, users highly rank frequently-updated apps instead of being annoyed about the high update frequency. However, 45% of the frequently-updated apps do not provide the users with any information about the rationale for the new updates. Nayebi and Ruhe [50] extracted customer value using crowd-sourcing for app features and provide optimised trade-off service portfolio planning. Nayebi et al. [51] performed two surveys with both users and developers in order to understand common release strategies used for mobile apps, their rationale and the impact perceived on users. Their results suggest that the app's release strategy is a factor that affects the ongoing success of mobile apps. Martin et al. [52] investigated app releases by conducting a longitudinal study on 38,858 apps mined from Google Play. Specifically, they used causal inference to identify app releases with most impact on ratings and downloads. The results revealed that paid apps that had significant positive effects were more expensive. The authors also reached 56 developers of significant releases, finding that 78% agreed with the causal assessment and 33% would consider changing their release strategy based on the findings from their study.

7. Conclusions and future work

In this paper we introduced a method to extract, from app store descriptions, usable information about the features of apps that captures some of the technical aspects of the apps in the store. We evaluated our approach on both the free and the non-free apps in the Blackberry App Store.

We found that the number of features per app (and the number of shared features between apps) follow a power law. We also found that, though there are a large number of zero-rated apps, the non-zero ratings accorded to apps by their users are, generally speaking, positive; more ratings occupy the higher, more favourable end of the rating spectrum.

The degree of correlation between rating, price and popularity is different for different app categories, as one might expect and as we report in detail in the paper. Our analysis indicates that there is a strong overall correlation between the ratings given to apps by their users and their popularity (i.e., rank of downloads). This correlation was observed for both free and non-free apps. This correlation is also present in the features we extract and so this feature information may be useful in its own right. We found that free apps received significantly higher ratings than their non-free siblings and that there is a mild correlation between price and the number of features offered, but we found little evidence for any correlation between the price of a non-free app and either its rating or popularity. This finding may offer useful guidance to developers in determining which features to consider when designing apps. As an example, they can provide insights into the added value of features under consideration for new products or next releases.

There are many potential avenues for future work that result from our findings. For example, since the publications of our short paper, follow up work has investigated the migrations of features across categories over different snapshots of an app store [12] and feature level clustering to re-draw and re-consider the boundaries of the categories of apps in an app store [13]. In future, we also intend to investigate predictive models of customer evaluations, and the interplay between functional and non-functional properties of apps, and the data available in app stores. We will also seek to develop multi-objective predictive models using Search Based Software Engineering (SBSE) [53–55]. The use of multi objective SBSE will allow us to develop predictive models tailored to the conflicting and competing needs of different app store developers and, perhaps also, their customers.

We also believe our data may contain many other interesting relationships between features, prices, ratings and ranks-of-downloads, that have yet to be discovered and reported upon. To facilitate this future work, we make the full dataset available for other researchers to mine, analyse and experiment with. The data used in the work reported in this paper can be downloaded from the UCLappA page:

www0.cs.ucl.ac.uk/staff/F.Sarro/projects/UCLappA/home.html.

Acknowledgement

The research is funded by the EPSRC CREST platform grant (EP/G060525) and DAASE programme grant (EP/J017515).

References

[1] M. Harman, Y. Jia, Y. Zhang, App store mining and analysis: MSR for app stores, in: Proceedings of the 9th IEEE Working Conference on Mining Software Repositories (MSR '12), IEEE, Zurich, Swiss, 2012, pp. 108–111.
[2] A. Finkelstein, M. Harman, Y. Jia, W. Martin, F. Sarro, Y. Zhang, App Store Analysis: mining App Stores for Relationships between Customer, Business and Technical Characteristics, Technical Report, University College London, 2014.
[3] H. Dumitru, M. Gibiec, N. Hariri, J. Cleland-Huang, B. Mobasher, C. Castro-Herrera, M. Mirakhorli, On-demand feature recommendations derived from mining public product descriptions, in: Proceedings of the 33rd International Conference on Software Engineering (ICSE '11), ACM, Hawaii, USA, 2011, pp. 181–190.
[4] J.-M. Davril, E. Delfosse, N. Hariri, M. Acher, J. Cleland-Huang, P. Heymans, Feature model extraction from large collections of informal product descriptions, in: Proceedings of the 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE'13), 2013, pp. 290–300.
[5] N. Hariri, C. Castro-Herrera, M. Mirakhorli, J. Cleland-Huang, B. Mobasher, Supporting domain analysis through mining and recommending features from online product listings, IEEE Trans. Softw. Eng. 39 (12) (2013) 1736–1752.
[6] W. Martin, F. Sarro, Y. Jia, Y. Zhang, M. Harman, A survey of app store analysis for software engineering, IEEE Trans. Softw. Eng. (2016), doi:10.1109/TSE.2016.2630689.
[7] M.D. Syer, B. Adams, Y. Zou, A.E. Hassan, Exploring the development of micro-apps: a case study on the blackberry and android platforms, in: Source Code Analysis and Manipulation (SCAM), 2011 11th IEEE International Working Conference on, IEEE, 2011, pp. 55–64.
[8] I. Ruiz, M. Nagappan, B. Adams, A. Hassan, Understanding reuse in the android market, in: Program Comprehension (ICPC), 2012 IEEE 20th International Conference on, 2012, pp. 113–122.
[9] S. Dienst, T. Berger, Static Analysis of App Dependencies in Android Bytecode, Technical Report, 2012. Technical note
[10] E. Loper, S. Bird, NLTK: the natural language toolkit, in: Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics (TeachNLP '02), Association for Computational Linguistics, Philadelphia, USA, 2002, pp. 69–72.
[11] E.L. Steven Bird Ewan Klein, Natural language processing with Python analyzing text with the natural language toolkit, OŘeilly Media, 2009.
[12] F. Sarro, A.A. Al-Subaihin, M. Harman, Y. Jia, W. Martin, Y. Zhang, Feature lifecycles as they spread, migrate, remain, and die in app stores, in: 23rd IEEE International Requirements Engineering Conference, RE 2015, 2015, pp. 76–85, doi:10.1109/RE.2015.7320410.
[13] A.A. Al-Subaihin, F. Sarro, S. Black, L. Capra, M. Harman, Y. Jia, Y. Zhang, Clustering mobile apps based on mined textual features, in: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '16, 2016, pp. 38:1–38:10, doi:10.1145/2961111.2962600.
[14] M.J. Shepperd, Foundations of Software Measurement, Prentice Hall, 1995.
[15] W. Martin, M. Harman, Y. Jia, F. Sarro, Y. Zhang, The app sampling problem for app store mining, in: Working Conference on Mining Software Repositories, 2015, pp. 123–133.
[16] C.E. Spearman, The proof and measurement of association between two things, Am. J. Psychol. 15 (1) (1904) 72–101.
[17] K. Pearson, Notes on regression and inheritance in the case of two parents, Proc. R. Soc. London 58 (1895) 240–242.
[18] A. Arcuri, L. Briand, A practical guide for using statistical tests to assess randomized algorithms in software engineering, in: 33rd International Conference on Software Engineering (ICSE'11), ACM, New York, NY, USA, 2011, pp. 1–10.
[19] A.A. Al-Subaihin, A. Finkelstein, M. Harman, Y. Jia, W. Martin, F. Sarro, Y. Zhang, App store mining and analysis, in: Proceedings of the 3rd International Workshop on Software Development Lifecycle for Mobile, DeMobile 2015, Bergamo, Italy, August 31, - September 4, 2015, 2015, pp. 1–2, doi:10.1145/2804345.2804346.
[20] M. Harman, A. Al-Subaihin, Y. Jia, W. Martin, F. Sarro, Y. Zhang, Mobile app and app store analysis, testing and optimisation, in: Proceedings of the International Conference on Mobile Software Engineering and Systems, MOBILESoft '16, ACM, New York, NY, USA, 2016, pp. 243–244, doi:10.1145/2897073.2897076.
[21] N. Gruska, A. Wasylkowski, A. Zeller, Learning from 6,000 projects: lightweight cross-project anomaly detection, in: Proceedings of the International Symposium on Software Testing and Analysis (ISSTA '10), ACM, Trento, Italy, 2010, pp. 119–130.
[22] W. Shang, B. Adams, A.E. Hassan, Using pig as a data preparation language for large-scale mining software repositories studies: an experience report, J. Syst. Softw. 85 (10) (2012) 2195–2204.
[23] M.M. Lehman, On understanding laws, evolution and conservation in the large program life cycle, J. Syst. Softw. 1(3) (1980) 213–221.
[24] M.D. Syer, M. Nagappan, A.E. Hassan, B. Adams, Revisiting prior empirical findings for mobile apps: an empirical case study on the 15 most popular open–source android apps, in: Conference of the Center for Advanced Studies on Collaborative Research (CASCON '13), 2013, pp. 283–297.
[25] M. Nagappan, E. Shihab, A.E. Hassan, Challenges in mobile apps: a multi-disciplinary perspective, in: Conference of the Center for Advanced Studies on Collaborative Research (CASCON '13), 2013, pp. 378–381.
[26] T. Menzies, Beyond data mining; towards "Idea Engineering", in: Proceedings of the 2nd International NSF sponsored Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE '13), San Francisco, USA, 2013.
[27] R. Minelli, M. Lanza, Software analytics for mobile applications - insights & lessons learned, in: Proceedings of the 17th European Conference on Software Maintenance and Reengineering (CSMR '13), IEEE, Genova, Italy, 2013.
[28] G. Sethumadhavan, Sizing android mobile applications, in: 6th IFPUG International Software Measurement and Analysis Conference (ISMA), 2011.

[29] T. Preuss, Mobile applications, functional analysis, and the customer experience, in: IFPUG (Ed.), The IFPUG Guide to IT and Software Measurement, Auerbach Publications, 2012, pp. 408–433.
[30] T. Preuss, Mobile applications, Function points and cost estimating, in: International Cost Estimation & Analysis Association Conference, 2013.
[31] H. van Heeringen, E. Van Gorp, Measure the functional size of a mobile app: using the COSMIC functional size measurement method, in: Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2014 Joint Conference of the International Workshop on, IEEE, 2014, pp. 11–16.
[32] F. Ferrucci, C. Gravino, P. Salza, F. Sarro, Investigating functional and code size measures for mobile applications: a replicated study, in: Product-Focused Software Process Improvement - 16th International Conference, PROFES 2015, Bolzano, Italy, December 2–4, 2015, Proceedings, 2015, pp. 271–287, doi:10.1007/978-3-319-26844-6_20.
[33] F. Ferrucci, C. Gravino, P. Salza, F. Sarro, Investigating functional and code size measures for mobile applications, in: 41st Euromicro Conference on Software Engineering and Advanced Applications, EUROMICRO-SEAA 2015, Madeira, Portugal, August 26–28, 2015, 2015, pp. 365–368, doi:10.1109/SEAA.2015.23.
[34] M.E. Joorabchi, A. Mesbah, P. Kruchten, Real challenges in mobile app development, in: 2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement, 2013, pp. 15–24.
[35] D.L. Ben Lulu, T. Kuflik, Functionality-based clustering using short textual description: helping users to find apps installed on their mobile device, in: Proceedings of the 2013 International Conference on Intelligent User Interfaces (IUI '13), ACM, Santa Monica, CA, USA, 2013, pp. 297–306.
[36] J. Kim, Y. Park, C. Kim, H. Lee, Mobile application service networks: Apple's app store, Serv. Bus. 8 (1) (2014) 1–27.
[37] R. Pandita, X. Xiao, W. Yang, W. Enck, T. Xie, WHYPER: towards automating risk assessment of mobile npplications, in: Proceedings of the 22Nd USENIX Conference on Security, SEC'13, 2013, pp. 527–542.
[38] Y. Yang, J.S. Sun, M.W. Berry, APPIC: finding the hidden scene behind description files for android apps, 2014, Available on line from University of Tennessee, USA.
[39] A. Gorla, I. Tavecchia, F. Gross, A. Zeller, Checking app behavior against app descriptions, in: 36th International Conference on Software Engineering (ICSE 2014), 2014, pp. 1025–1035.
[40] S.E.S. Taba, I. Keivanloo, Y. Zou, J. Ng, T. Ng, An exploratory study on the relation between user interface complexity and the perceived quality of android applications, in: International Conference on Web Engineering (ICWE 2014)), 2014. Late Breaking Result.
[41] M.D. Syer, M. Nagappan, B. Adams, A.E. Hassan, Studying the relationship between source code quality and mobile platform dependence, Softw. Quality Control 23 (3) (2015) 485–508.
[42] J. van Angeren, V. Blijleven, S. Jansen, S. Brinkkemper, Complementor embeddedness in platform ecosystems: the case of google apps, in: 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST 2013), 2013, pp. 37–42.
[43] S.L. Lim, P.J. Bentley, Investigating app store ranking algorithms using a simulation of mobile app ecosystems, in: IEEE Congress on Evolutionary Computation, 2013, pp. 2672–2679.
[44] I.J.M. Ruiz, M. Nagappan, A. Bra, T. Berger, S. Dienst, A.E. Hassan, On the relationship between the number of Ad libraries in an android app and its rating, 2014, Available on line from Queen's University, Canada.
[45] I.J. Mojica, M. Nagappan, B. Adams, T. Berger, S. Dienst, A.E. Hassan, Impact of Ad libraries on ratings of android mobile apps, IEEE Softw. 31 (6) (2014) 86–92.
[46] V. Avdiienko, K. Kuznetsov, A. Gorla, A. Zeller, S. Arzt, S. Rasthofer, E. Bodden, Mining apps for abnormal usage of sensitive data, in: 2015 International Conference on Software Engineering (ICSE), IEEE, Florence, Italy, 2015, pp. 426–436.
[47] M. Linares-Vásquez, G. Bavota, C. Bernal-Cárdenas, M. Di Penta, R. Oliveto, D. Poshyvanyk, API change and fault proneness: a threat to the success of android apps, in: Proceedings of the 9th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2013, ACM, New York, NY, USA, 2013, pp. 477–487.
[48] G. Bavota, M. Linares-Vasquez, C. Bernal-Cardenas, M. Di Penta, R. Oliveto, D. Poshyvanyk, The impact of API change- and fault-proneness on the user ratings of android apps, Softw. Eng. IEEE Trans. 41 (4) (2015) 384–407.
[49] S. McIlroy, N. Ali, A.E. Hassan, Fresh apps: an empirical study of frequently-updated mobile apps in the Google play store, Empirical Softw. Eng. (2015) 1–25.
[50] M. Nayebi, G. Ruhe, Trade-off service portfolio planning–a case study on mining the android app market, PeerJ PrePrints (2015).
[51] M. Nayebi, B. Adams, G. Ruhe, Release practices for mobile apps – what do users and developers think? in: 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), 1, 2016, pp. 552–562.
[52] W. Martin, F. Sarro, M. Harman, Causal impact analysis for app releases in google play, in: Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2016, Seattle, WA, USA, November 13–18, 2016, 2016, pp. 435–446, doi:10.1145/2950290.2950320.
[53] M. Harman, The relationship between search based software engineering and predictive modeling, in: 6th International Conference on Predictive Models in Software Engineering (PROMISE 2010), Timisoara, Romania, 2010.
[54] R. Saborido, G. Beltrame, F. Khomh, E. Alba, G. Antoniol, Optimizing user experience in choosing android applications, in: IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2016, Suita, Osaka, Japan, March 14–18, 2016, 2016, pp. 438–448.
[55] F. Sarro, A. Petrozziello, M. Harman, Multi-objective software effort estimation, in: Proceedings of the 38th International Conference on Software Engineering, ICSE 2016, Austin, TX, USA, May 14–22, 2016, 2016, pp. 619–630.