Università di Pisa

Facolà di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Forecast emerging artists success on Last.fm music service: a data-driven study

Relatori:
Prof.essa Anna Monreale
Ph.D. Laura Pollacci

Candidata:
Alexandra Lavinia Bradan

Sessione straordinaria
Anno Accademico 2018/2019

Alla polpetta più prodigiosa che sia mai stata creata,
che dall'alto di nuvole puffose corre, dormicchia
e mi scruta con occhietti curiosi. Ti sono grata
per avermi scortata fino a qui, da prima
correndomi affianco e dopo continuando
a farlo tramite ali angeliche.
Ed è sempre a te che dedico
questa prima conquista.
A te che sei stato e rimani
il mio migliore amico,
il confidente più stretto,
l'affetto più sincero
ed il cagnolino più
beffardo del paradiso,

Al mio Ricko

Ringraziamenti

Alla fine di una scalata, spaziando con lo sguardo, non si può far altro che contemplare il panorama che si spande attorno, respirare per un momento il senso di libertà che riempie i polmoni e sentire nascere dentro di sé l'orgoglio personale per essere arrivati alla meta. È così che immagino il giorno in cui queste parole verranno incise su questa tesi, riassunto e compendio di un percorso lungo, tortuoso, mai scontato, ma incredibilmente straordinario e gratificante. Ad accompagnarmi in questa avventura, donandomi conforto, compagnia ed alle volte anche qualche bella strigliata per farmi ritornare in senno, ho avuto al mio fianco persone speciali ed incredibili, che tengo a ringraziare singolarmente.

Il grazie più importante lo riservo alla mia dolce Iustina, sorella premurosa e supporto inscalfibile in ogni tappa di questo percorso. Grazie per le sconfinate attenzioni e per l'affetto incondizionato che mi hai voluto regalare nelle mie rinomate sventure. Te ne sono immensamente grata.

Un grazie speciale lo riservo a Dino, eterno Peter Pan, che mi ha insegnato l'arte di vivere giorno per giorno, con semplicità e sorrisi, che sono gli unici ingredienti veramente essenziali per essere felici. Avrò cura di questa inestimabile lezione, a venire.

Ringrazio tutti i membri della mia speciale "famiglia del Fibo": la mia piccola Laura, dolce e forte al contempo, dispensatrice di biscotti ed incredibili aneddoti da raccontare; il piccolo Frank, maestro e mentore in questa "selva oscura", da cui spero di poter attingere altri insegnamenti a venire; il piccolo Fabio, consigliere tecnologico di fiducia e dispensatore di sorrisi e buon umore; la piccola Giulia, travolgente ciclone di vitalità e tenace modello da cui prendere spunto; il piccolo Carmine, speciale ed unico, come solo un vero molisano lo può essere; il piccolo Lo Sciulli, ideatore delle omonime sciullate (a cui fin troppo spesso mi rifaccio) e di una simpatia innata e coinvolgente; il piccolo Gabri, l'artista più impegnato e multigenere che conosca, da cui devo ancora apprendere l'arte di conciliare multiple cose alla volta; la piccola Elena G., lontana ma vicina allo stesso tempo, a cui vorrei rubare un po' del coraggio e della sicurezza che custodisce; il piccolo Simone e la piccola Elena I., che seppur conosca in minoranza, ammiro per semplicità e stima. A questa famiglia allargata unisco anche le mie piccole Eli ed Ari, che ringrazio per l'amicizia e l'affetto innati che da subito ho provato in vostra compagnia.

Ringrazio, infine, tutti coloro che hanno colorato la mia quotidianità al Fibonacci, nelle aule studio, durante le lezioni, nelle pause caffè ed in misura maggiore, in direzione delle

mie amate macchinette.

Senza tutti voi questo percorso sarebbe stato meno emozionante da vivere e da affrontare. Mi avete donato quella spensieratezza unica e magica per combattere le sfide più ardue (Chatty, fra tutte) ed arrivare sino a qui; ed ora che la linea del traguardo è vicina, vi voglio ringraziare dall'angolo più profondo del mio cuore, dove rimarrete custoditi, per sempre, con l'appellativo "i miei piccoli"...

Contents

List of Figures

List of Tables

Chapter 0

Introduction

As we watch a group or society over time, we'll see that new practices can be introduced and either become popular or remain obscure; meanwhile, established practices can persist or potentially fade over away.

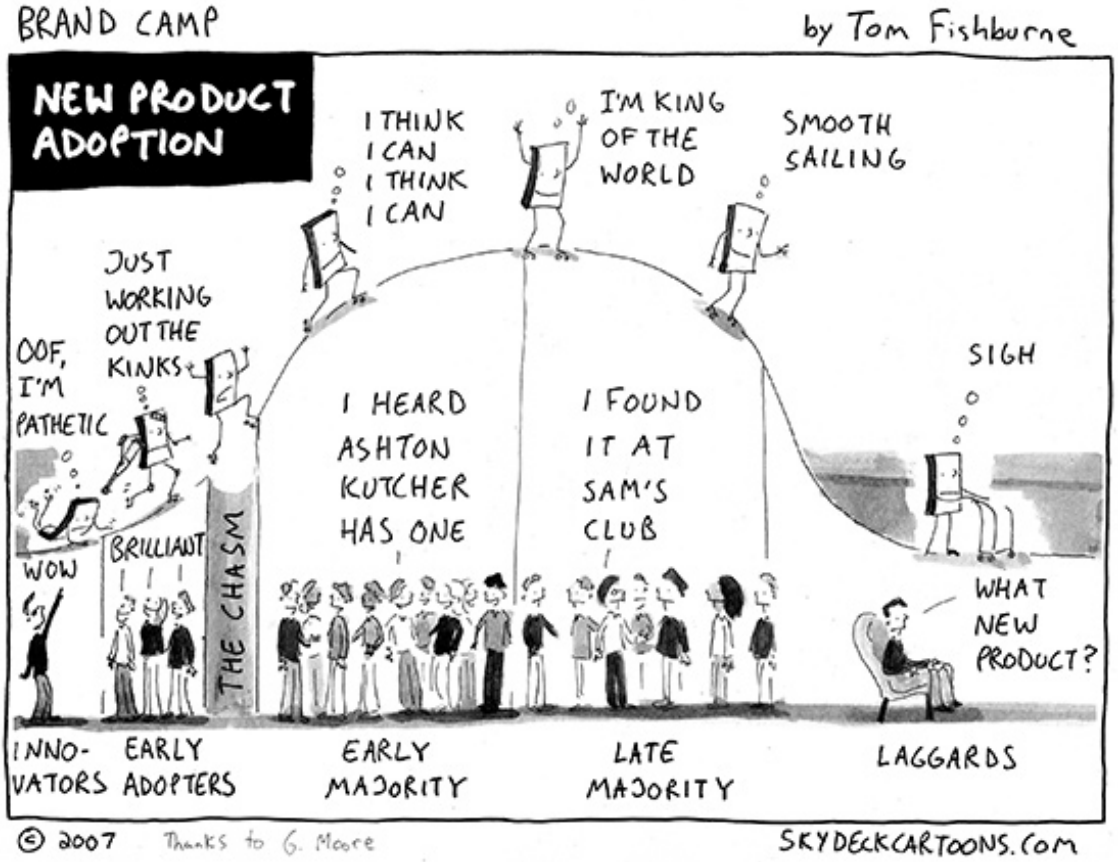

Figure 0: The adoption of a new product

The way new practices spread through a population depends largely on the fact that people influence each other's behavior, because of an underlying human tendency to conform (word-of-mouth effects [1]). When individuals have incentives to adopt their neighbors'

behavior in the network, we can get cascading effects, where a new behavior starts with a small set of initial adopters, and then spreads radially outward through the network. Cascading behavior in a network is sometimes referred to as *social contagion*, because it spreads from one person to another in the style of a biological epidemic. The surprising effectiveness with which people are able to accomplish such spreading suggests characteristic patterns of structure at the network level that help facilitate these types of activities. My goal in this report is to try to identify the early adopters (aka leaders) of such behaviors and to give them a topological characterization, as well as to train a predictive model to forecast whether their choices can be grounded to predict the success of new, previously unseen items. This will allow:

- to develop marketing strategies for targeting the "right" set of nodes to start a spreading process from them (*Influence maximisation problem* [22]);
- to understand if an innovation will be successful or not, in its early stage of adoption, grounding previous encountered diffusion patterns;
- to prevent traffic and server loads. The knowledge of item patterns can help service providers to forecast high workloads and adequately distribute content or provision resources in order to improve their Quality of service (QoS).

My first leader detection strategy is depicted from the paper *The Three Dimensions of Social Prominence* [2]. The paper uses a UK user sample graph extracted from the Last.fm service[1] to validate the intuition proposed: instead of just studying cascade effects and the maximization of influence by a given starting seed, the paper analyzes three different dimensions of a leader's prominence based on:

1. how many neighbors;
2. how distant nodes;
3. how engaged nodes are.

In the paper these dimensions are referred as:

1. Width;
2. Depth;
3. Strength.

My second and most peculiar innovator detection strategy is depicted from the paper *Forecasting success via early adoptions analysis: A data-driven study* [3], which still focuses on items' diffusive processes to detect whether a node is more influential than another. More over, the paper investigates each user adoption behavior to classify the adopters into three categories:

[1]Last.fm is a music service launched back in 2002 as an online music database, recommendation service, and music-focused social network. One of the most attractive things about Last.fm is the ability to scrobble the music that a user listens from various locations so it can better build music suggestions that suit his/her tastes, acting like a powerful music discovery engine.

1. the one that adopts mainly successful items, by chance or preference;

2. the one that adopts mainly unsuccessful items (still useful to detect if an unseen item will reach success);

3. the one that adopts equally successful and unsuccessful items which can be considered neutral.

Based on this classification the paper exploits an interesting predictive model to forecast whether a novel item will be a success (Hit) or a failure (Flop), grounding its forecasting power in the choices made by the special category of *Hit-Savvies* (adopters who adopts mainly successful items before others). I trained my data set with the predictive model too, to get some more insight into it. I finally tried to find some similarities among the two leader detection strategies proposed, and which dissimilarities they instead shared. I revised the papers' detection algorithms, broadening the social context to include a worldwide social graph and detecting prominent users among a directed graph[2]. The study was conducted using the Python programming language and the Python libraries:

- BeautifulSoup [4]: users and tags scrapping;
- NetworkX [5]: social graph analysis;
- IGraph [6]: social graph analysis and graph statistics visualisation;
- CDlib [7]: communities discovery;
- SciPy [8]: clustering algorithms;
- NumPy [9]: scientific computing;
- Pandas [10]: data analysis and manipulation;
- SqlLite [11]: SQL database querying;
- Matplotlib [12]: graph plotting;
- Seaborn [13]: statistical data visualisation;

and the visual tools:

1. Gephi [14]: social graph visualization and exploration;
2. Cytoscape [15]: social graph visualization and exploration.

The original code of the leader detection strategy and Hit-Savvy model can be found at:

$$https://github.com/GiulioRossetti/leader\_detect$$
$$https://github.com/GiulioRossetti/hit-savvy$$

[2]Both papers used as sample an undirected graph.

I had to adjust the leader detection algorithm to fit my data representation, as well as providing the code for the mining analysis carried. My version of the code is reachable at:

$$https://github.com/kdd-lab/2019\_Bradan/LeaderDetection$$
$$https://github.com/kdd-lab/2019\_Bradan/Hit-Savvy$$

Chapter 1 describes how I've managed to collect the data used in my study (Section 1), as well as, how I've used a direct mining approach to model the spread of social influence to detect Leaders (Section 2), characterize them (Section 3) and understand their musical flow over the network (Section 4). Chapter 2 introduces how exploiting special innovators as indicators (Section 5), may lead to a predictive model used to forecast the success of unseen artists (Section 6). My predictive scores and the insight behind them are discusses in Section 7, while Section 8 tries to investigate the existance of a correlation between the ability to spot successful items and the power to be influential. Chapter 3 offers a more in depth of user's listening patterns and tastes characterisation. Lastly, Chapter 4 compares users' musical features with topological node clustering, called communities.

Chapter 1

Leaders detection

1 Data Collection

Most of the data needed for the investigation was retrieved using Last.fm's developer API [16], like the friendship bounds between the users and their listening data[1]. Since Last.fm didn't provide a function to retrieve the users of the service, in the first place I had to scrap them:

1. I iterated over a list of music charts (mainly music charts involving new artists and new music genres, since I needed to focus on artists that were previously non-existent);
2. for every chart I retrieved the relative artists figuring in it;
3. for every retrieved artist I consulted its top listeners pages and from them I got the seed users which figured in my analysis.

Through the above scrapping process I was able to collect a total of 341,136 users, among which I had to exclude 147,727 users due to lack of friendship bounds. Figure 1.1 shows users' worldwide provenience. For each seed user, I retrieved:

1. his connections, up until the first degree of separation, chopping off the users with more than 500 friends due to space limitations;
2. for each week in the time window from July-16-2017 to July-16-2019, the artists he listened;
3. some personal information (nationality, number of play-counts and number of artists listened on the platform).

The resulting data set is composed by 193,409 users and 2,142,977 following connections among them. Since Last.fm has recently introduced an asymmetric friending model (where a user A may follow another user B, but B might choose not to follow A), the corresponding social graph is directed and unweighted (filters and weights will be add later in the analysis).

[1] All the user related information were anonymized after the collection, and before the analysis, to comply with Last.fm policy.

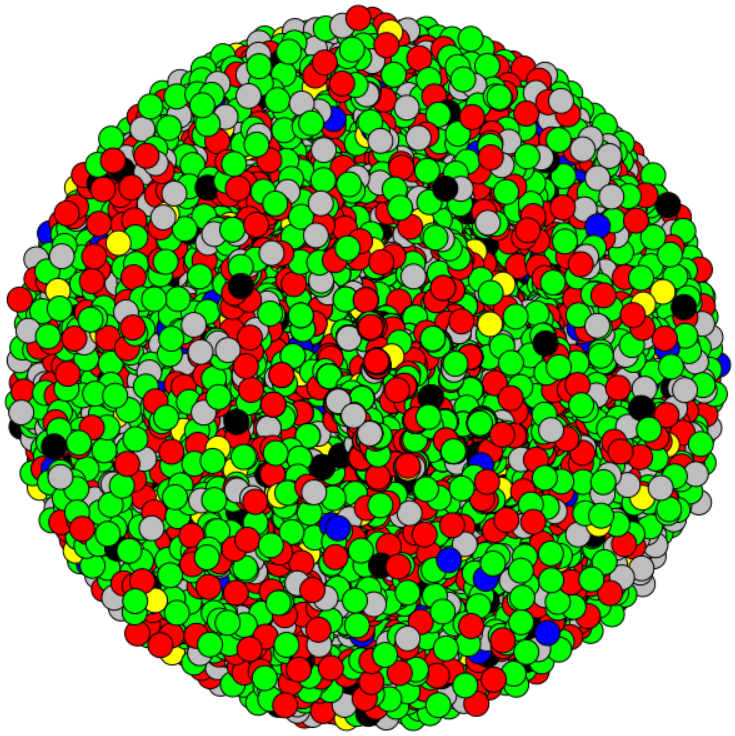

Figure 1.1: Social graph based on users' provenience (green: Europe (88,473), red: America (54,144), black: Africa (3,319), yellow: Asia (6,000), blue: Oceania (4,111), Gray: Missing information (37,361)).

Definition 1 (Social Graph):
A social graph G *is composed of a set of actors (nodes)* V *connected by their social relationships (edges)* E*. Each edge* e ∈ E *is defined as a couple* (u, v) *with* u, v ∈ V *and, where not otherwise specified, has to be considered directed. The direction of the edge from* u *to* v *represents the* ***following relationship*** *which exists among* u *and* v*: if the node* u *follows the node* v *then exists a starting edge from* u *to* v*.* $\Gamma(u)$ *identifies the neighbor set of the node* u ∈ V *(users he follows) and* F_G *identifies the* G*'s frontier set (graph's leaf nodes).*

Definition 2 (Action):
An action $a_{u,\psi} = (w, t)$ *defines the adoption by an actor* $u \in V$*, at a certain time* t*, of a specific object* ψ *with a weight* $w \in R$*. The set of all the actions of nodes belonging to a social graph* G *will be identified by* A*, while the object set will be called* Ψ*.* $G_\psi = (V_\psi, E_\psi)$*, where* $V_\psi \subset V$ *and* $E_\psi \subset E$*, identifies the induced sub-graph on* G *representing respectively the set of all the actors that have performed an action on* ψ*, and the edges connecting them. To detect the early adopters among the seed users:*

- *the actors are the users in the data set;*
- *the objects are the artists collected among users' listening;*
- *the actions correspond to artists' listening;*

- *action's weights are the play-counts made during the study period.*

To detect which are the most prominent users in my network, in this section I will follow, revisiting, the steps described in [2]. These users are called **leaders** and they are so crucial because they are capable to anticipate trends (by adopting before everybody the innovations) and as an effect they have the power to influence their neighborhood. In the context of my study, as an innovation, I identify the discovery and spreading of a new artist through the community: I'm interested in detecting who are the first listeners of such new artists and how the spread of such innovation took place. As a consequence, I gathered a set of *new* artists, namely artists whose absolute first listening was dated six months after my observation period length. If an artist was in activity before the observation time window, there is no way to know if a user has listened to it before, therefore nullifying the leader detection strategy. Since the original object set Ψ contains a total of 6,851,358 artists I had to filter it: 3,776,459 artists reckon their absolute first listening six moths after my observation time window, among which I furthermore excluded 3,754,458 artists:

- who had less than 100 listeners on the Last.fm platform;
- who hadn't music tags associated;
- who had music tags associated, but with less than 100 taggings,
- who had music tags associated with at least 100 taggings, but didn't refer to a musical meaning and as a result couldn't help with the classification of the artists in a main music genre (how I managed the tags mapping is described in the section 4);
- cutting off the artists listened by less than 5 seed users, to lighten even more the item set[2] .

The resulting restriction counts 22,001 artists, which from latter on will correspond to the object set Ψ of concern. The relative action set A is made up of 11,620,917 actions and the number of adopters involves 193,409 users. Table 1.1 sums up this information.

Dataset	**Items**	**Adopters**	**Adoptions**	**Timespan**	**Time unit**
Last.fm	22 001	193 409	11 620 917	2 years[3]	1 week

Table 1.1: Datasets' details

[2]The leader detection analysis is based on a Breadth-first search (BFS) on the graph, repeated at every iteration for each artist. As a result, I need to have a light item set for time and space complexity.

[3]Half of the first year is used only as a time-span to detect unseen artists. In fact, the total number of adoptions refers only to the remaining a year and a half [Jan-14-2018 - July-16-2019].

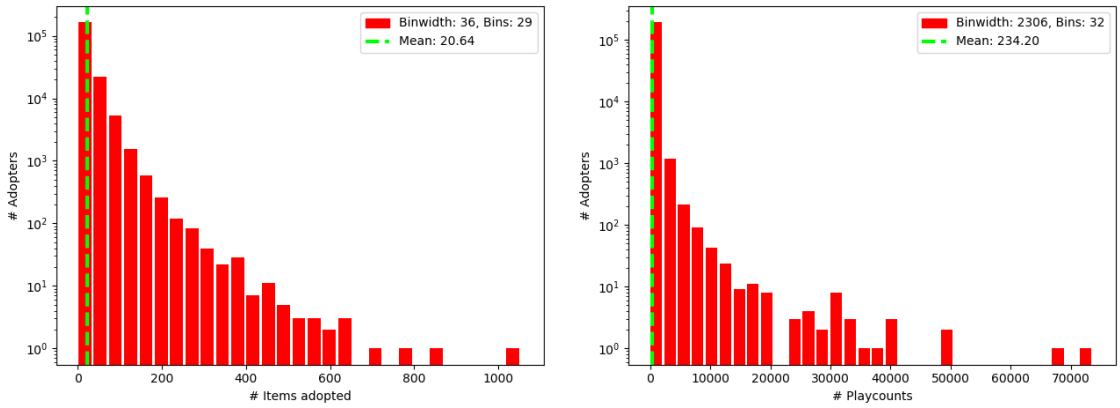

Figure 1.2: Distribution of users and artists adopted (left) and users and number of play counts made (right).

Figures 1.2 and 1.3 depict some useful distributions: seed users per artists adopted and seed users per numbers of play-counts made, artists per number of adopters and artists per number of play-counts received, respectively. They are all binned with the Doane Estimator[27] and the y axis is logarithmic to smooth overlapping bins[4]. The distribution in Figure 1.2's (left) shows how most of the users in my data set haven't an exorbitant adoptions rate. In the first three bins we can found respectively 163,420, 21,890 and 5,378 users, who adopted between one artist to 144, with an adoption mean of 20. A similar scenario can be found in Figure 1.2 (right), where 191,787 out of 193,409 seed users had made between 1 to 2,308 play counts, with a mean of 234. In regard to the total number of adopters and play counts that each of the 22,001 artists receive, Figure 1.3 (left/right) confirms the tendency of having low values, too. The majority of the artists rely on a mean of 181 unique adopters and of 2,058 play counts.

2 Leaders detection

On these premises, the process for the leader detection strategy that I've led is described as follows:

1. I constructed the directed social graph G, where:
   - the nodes are the actors (users);
   - the edges are their social ties (following relationships, having the previously described direction);
2. for every action $\psi \in \Psi$ (new artist) I retrieved the induced subgraph[24] G_ψ from G. The induced subgraph G_ψ contains all the actors that had performed an action on ψ in G (listen at least one time to the artist) and the edges connecting them;

[4]Among Freedman Diaconis, Scott, Rice, Sturges and Sqrt Estimators, I choose the Doane Estimator due to better data plotting output. All the methods to estimate the optimal number of bins were provided by default by the Scipy's *numpy.histogram* function [17].

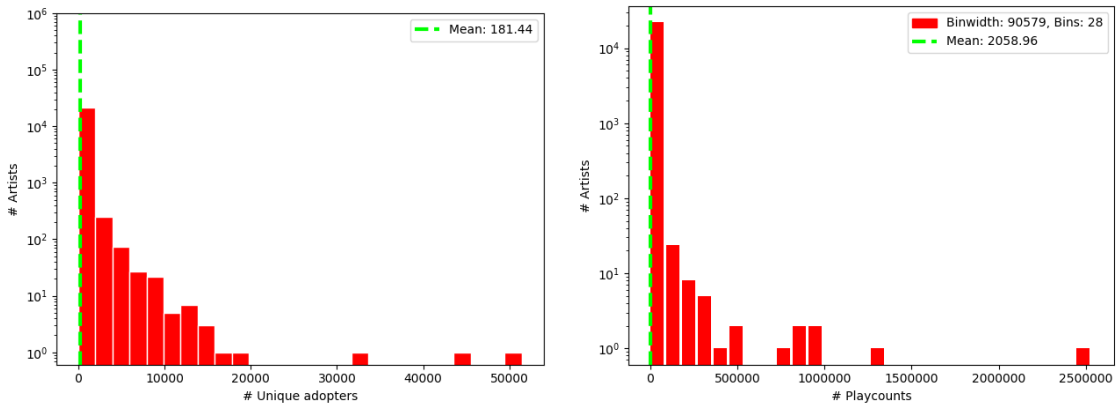

Figure 1.3: Distribution of artists and number of adopters (left) and artists and total number of play counts (right).

3. performing a Breadth First Search on G_ψ, I reconstructed the object's diffusion path, obtaining as a result the graph G'_ψ. To detect which node had influenced whom, I had to take account that the edges in G_ψ represent the following relationships among the users which adopted ψ. Unlike what the intuition suggested, I had to look to the edges the other way around:

   (a) if the target node of an edge in G_ψ had performed its action before the source node and its action was performed within a temporal resolution parameter δ[5], then the target node had influenced the source node on performing the action on ψ. In G'_ψ then I insert an edge starting from the target node in G_ψ and ending in the source node (switching the original edge's direction in G_ψ), labeling the edge with target node's temporal action $t_{target,\psi}$;

   (b) if the target node of an edge in G_ψ had performed its action after the source node or in any case not within the temporal resolution parameter δ, then it hadn't influenced the source node and in G'_ψ the edge between them is absent.

   Proceeding as I described resulted in the transformation of all the existing symmetrical friending relationships into asymmetrical ones: if user A follows user B and the way around, only one of them influences the other. If the temporal resolution parameter isn't exceeded, otherwise the symmetrical tie is cut off and the two nodes are independent of one another. Regarding the asymmetrical ties, they can be reversed or cut off;

4. consulting G'_ψ I detected the leaders for the given action ψ. To be defined a leader an actor should not have any incoming edges in G'_ψ. This is because a prominent user cannot act after another user (they are, in their surroundings, innovators). Given this definition, for each directed connected component $C_\psi \subset G'_\psi$ multiple nodes can belong to the ψ's set of leaders L_ψ;

[5]Like in the paper, I used a temporal resolution parameter δ equal to 3 weeks to limit the cascade effect made by a leader and detect when a play-count made by a user is independent of the one made by the leader.

5. for every leader $l \in L_\psi$ found, I computed its Minimum spanning tree[25] $T_{l,\psi}$, having its root in l and built minimizing the temporal label assigned to the edges, to study its path of diffusion, its topological characteristics and to validate the intuition behind the three dimensions of social prominence: Width, Depth and Strength.

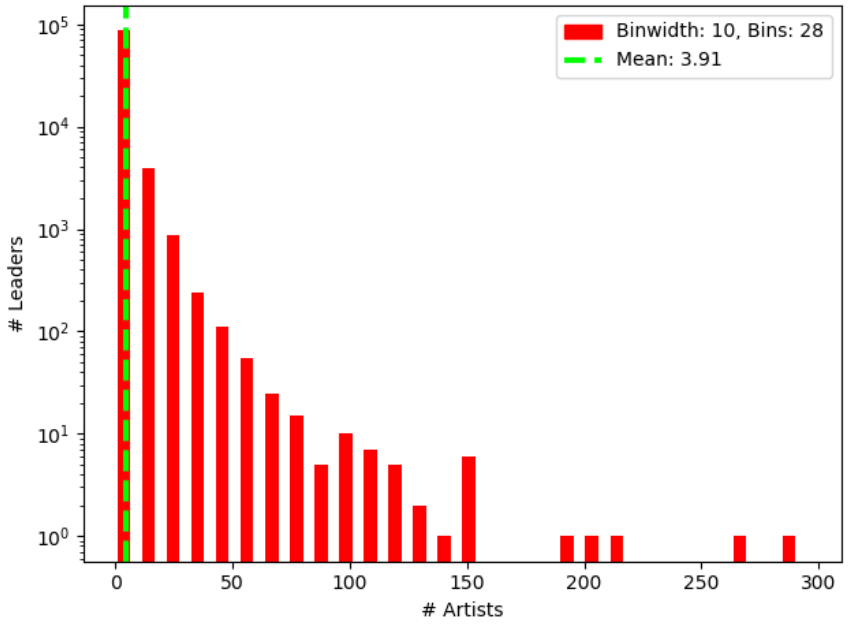

Figure 1.4: Distribution of artists per leaders.

The results of the described leader detection strategy can be summarized as follows:

1. I found **358,577 leaders** for **9,898 unseen artists**;

2. among all the leaders found, the number of **uni-vocal leaders** is up to **91,652**.

Figure 1.4 shows the distribution of artists per leader (how many user are leaders for a given artist) and depicts an important scenario: a leader is not bounded to be the leader just for one object ψ, but he is free to be prominent for multiple ψ. Here the Figure 1.4 is binned with the Doane Estimator[27] and the y axis is logarithmic to smooth overlapping bins. The majority of the users founded are leaders for 1 to 11 artists, as shown in the first bin, where in the **1-11 artists range** we can find a total of **86,325 leaders**. The value is the result of the summing up of all the leaders binned in the first bin and constitute the first bin's y axis value. The second and the third bin contains the users who are leaders for the 12-22 and 23-33 artists range. A total of 3,968 leaders for the second bin and 876 leaders for the third bin. Leaders frequency degrades quite linearly with the increasing number of artists, arriving to count 1 leader for 273 artists and **1 leader** for **295 artists** in the twenty-sixth bin and twenty-eighth and last bin, respectively.

3 Three dimensions of social prominence

To capture the three dimensions of social prominence proposed in [2], on every leader's Minimum spanning tree $T_{l,\psi}$, I performed the Width, Depth and Strength measures, according to the following definitions:

Definition 3 (Width):
Let G be a directed social graph, V the set of its actors (nodes), E the diffusion direction among them (edges), $\psi \in \Psi$ an object, $a_{u,\psi}$ an action performed by the actor $u \in V$ on the object ψ, $L_\psi \subset V$ ψ's leader set, $l \in L_\psi$ a leader and $\Gamma(l)$ the neighbor set of l*: the function $width : L_\psi \rightarrow [0, 1]$ is defined as:*

$$width(l, \psi) = \frac{|\{u | u \in \Gamma(l) \wedge \exists a_{u,\psi} \in A\}|}{|\Gamma(l)|}$$

and computes the ratio between the neighbors influenced by leader l (users who follow the leader) out of all his possible neighbors (total number of followers).

Definition 4 (Depth):
Let $T_{l,\psi}$ be a minimum diffusion tree for a leader $l \in L_\psi$ and a given object $\psi \in \Psi$: the function $depth : T_{l,\psi} \rightarrow N$ is defined as the the maximal path from l to every leaf node in the tree:

$$depth(l, \psi) = \max_{u \in T_{l,\psi}\ ,\ k_u^{out} = 0} shortest\_path(T_{l,\psi}, l, u)$$

where k_u^{out} is u's outgoing degree.
The function $depth\_avg : T_{l,\psi} \rightarrow R$ computes the average length of all the paths from l to any leaf node:

$$depth\_avg(l, \psi) = \frac{shortest\_path(T_{l,\psi}, l, u)}{|F_l|} \qquad \forall u \in T_{l,\psi}\ ,\ k_u^{out} = 0$$

where $F_l = \{u | u \in T_{l,\psi}\ ,\ k_u^{out} = 0\}$ is $T_{l,\psi}$'s frontier.

Definition 5 (Strength):
Let $T_{l,\psi}$ be a minimum diffusion tree for a leader $l \in L_\psi$ and an object $\psi \in \Psi$; $0 < \beta < 1$ a damping factor[6]*: the function $strength : T_{l,\psi} \times (0, 1) \rightarrow R$ is defined as:*

$$strength(T_{l,\psi}, \beta) = \sum_{i \in [0\ ,\ depth(l)]} \beta^i L(T_{l,\psi}, i)$$

where $L : T_{l,\psi} \times N \rightarrow R$ is defined as:

$$L(T_{l,\psi}, i) = \sum_{\{u | u \in T_{l,\psi} \wedge distance(l,u) = i} \} \frac{w_{u,\psi}}{w_u}$$

and represents the sum, over all the nodes u *at distance* i *from* l*, of the ratio between the weight of action ψ and the total weight of all the actions taken.*

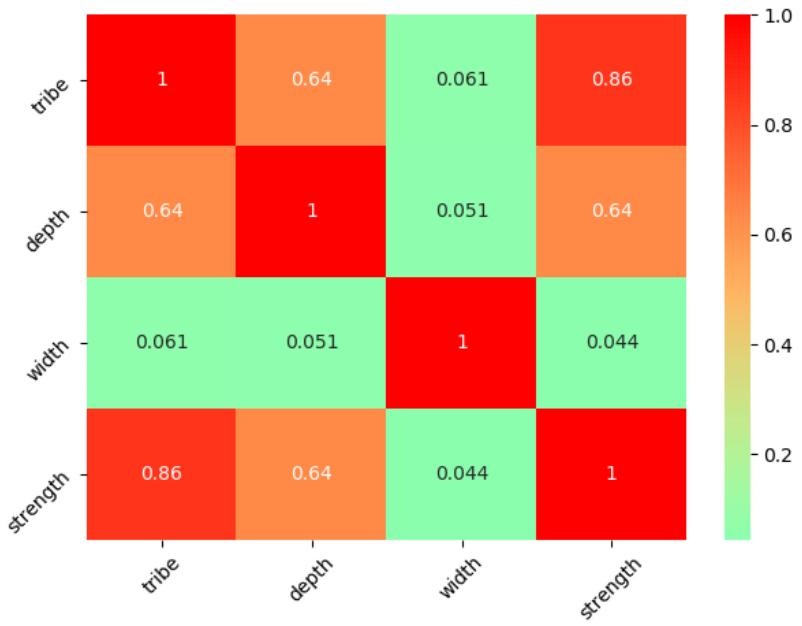

Figure 1.5: Correllogram of the three dimensions of social prominence.

As we can see from the Correllogram in Figure 1.5, Width, Depth and Strength are positive correlated, but we can't depict a clear dependence among them[7]. Instead it is the combination of the three measures that really characterizes the leaders, as we latter we will see. To validate this these measures as prominence features, I compared my results on traditional networks measures too, to detect associations among more traditional node centrality characteristics. Besides Width, Depth and Strength then, I calculated for each leader also:

- the In-Degree and the Out-Degree (number of edges pointing to[28] and out of[29] the node);
- the Clustering coefficient (ratio of triangles over the possible triads centered on the node[32]);
- the Betweenness centrality (share of the shortest paths that pass through the node[33]);
- the Closeness Centrality (inverse average distance between the node and all the other nodes of the network[34]);
- the PageRank (computes a ranking of the nodes in the graph G based on the structure of the incoming links[35]).

I recall that for a node its in-degree resembles the number of neighbor nodes that are following him, while the out-degree represents the number of neighbor nodes that the node is

[6] $\beta = 0.5$ in my case, in accordance to the paper.

[7] We notice that an increasing in Strength is associated with a wider range of nodes (Tribe) and greater Depth, as the common intuition suggests.

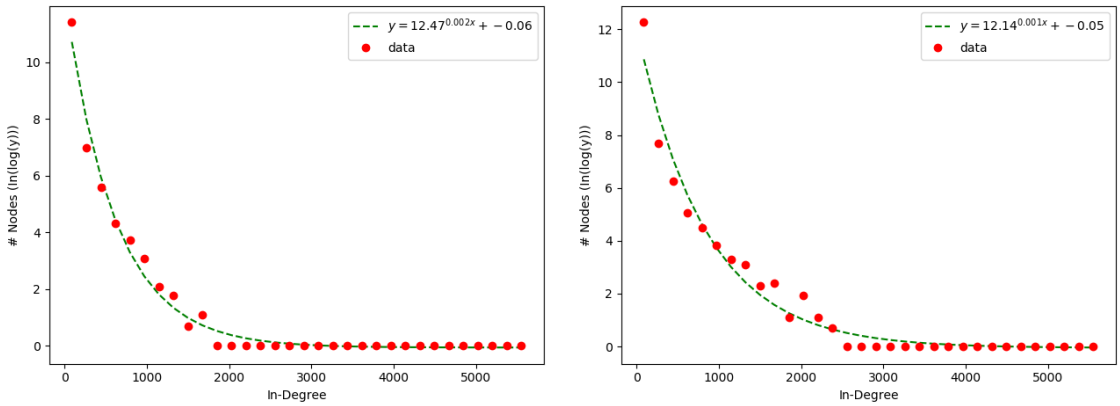

Figure 1.6: Graph's in-degree distribution (left) and Leaders' in-degree distribution (right).

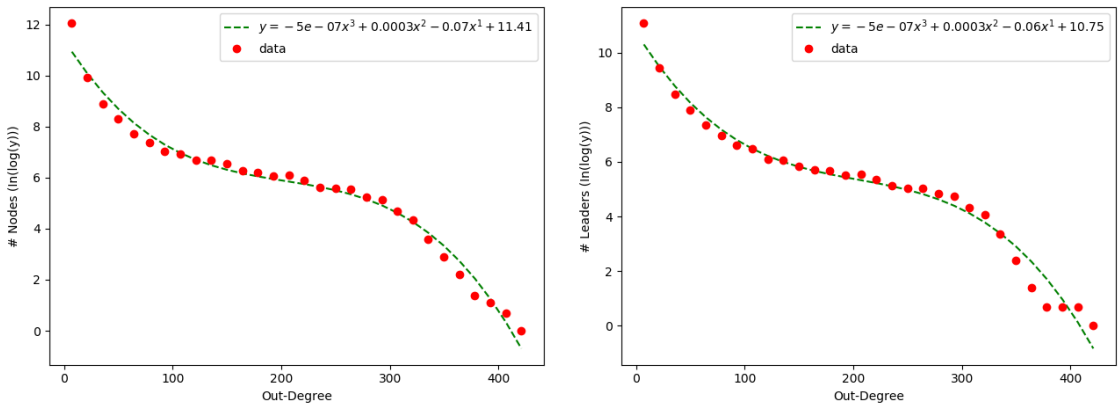

Figure 1.7: Graph's out-degree distribution (left) and Leaders' out-degree distribution (right).

following[8]. In Figure 1.6 and in Figure 1.7, I report the in-degree and out-degree distributions for all the nodes in the graph and for only the leaders, respectively. All the distributions are binned with the Doane Estimator[27], the y axis is double logarithmic scaled (first with $\log_{10} y$ than with $\ln(\log_{10} y)$) for plotting clearness and their functional fitting is done with a power law and a third polynomial function, respectively.

The in-degree plots are both binned with 31 bins and binwidth equal to 177. We can highlight a peak of in-degreeness in the first two bins: for 210,737 and 2,175 users in Figure 1.6 (left) and for 90,135 and 1,085 leaders in Figure 1.6 (right). Since the uni-vocal leaders that I found amount to 91,652, clearly a higher in-degree isn't synonym of more influential power. However, watching the two distributions' long tails, we can recall that the majority of hubs[9] are leaders and for them the influential power is explained thanks to their high connectivity to the rest of the network. In addition with an in-degree correlation coefficient $\mu = 0.0945$, we see a weak assortativity mixing between the nodes of the graph (nodes tend to be followed by nodes with a similar rate of followers, in most cases because they are mutual friends, as later we will see, and so are involve in symmetrical bonds).

The out-degree plots are both binned with 30 bins and binwidth equal to 15. We can highlight a peak in out-degreeness in the first eight bins with ranges of 169,954 (first bin) to 1,017 (eight bin) users and 64,050 (first bin) to 646 (eight bin) leaders. We can see again that the leaders out-degree distribution doesn't differ from the global one, confirming again the representative network's sample depicted by the leaders. The distribution degrades in fact like the graph's out-degree plotting, and this underlines how even if a user is a leader and so prominent for his neighborhood, he still can have in its turn a broad following circle. With an out-degree correlation coefficient $\mu = 0.0199$, we notice again a weak assortativity mixing between the nodes of the graph. Nodes tend to follow nodes with a similar rate of followings, again because in most case they are part of a mutual friendship circle.

[8]We consider out-degree as a proxy for activity, while in-degree measures node's popularity.

[9]Nodes with much higher in-degree or out-degree than the rest.

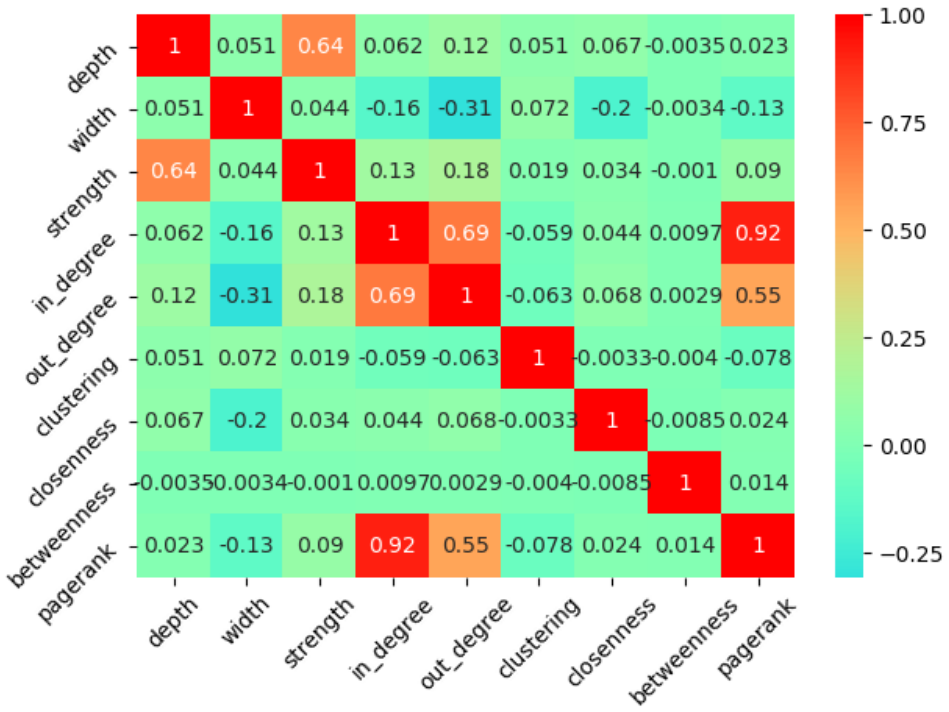

Figure 1.8: Pearson correlation coefficient between Width, Depth, Strength and other network statistics for leaders.

In Figure 1.8, I reported the Pearson correlation coefficient ρ[36] between the network measure that I've previously mentioned. For the significance of ρ-values, the traditional choice is to set the threshold at $\rho \leq 0.01$. However, given my number of observations, I decided to be more restrictive and set my threshold at $\rho \leq 0.0005$. I also consider a ρ value significant if $|\rho| \geq 0.1$. As the common intuition suggests, Depth is positively correlated with in-degree (users who follow a node and that the node may follow back) and to a greater extent with out-degree. More followings a node has and more chances exist that these users follow him back, being influenced and then spreading the received influence among the network. The Depth measure is also associated with high Strength, meaning that longer the chain propagated by a leader in the network are and stronger is the intensity with which the influenced users adopt a new item[10]. We have seen previously how leader nodes aren't characterized by an excessively high in-degree and out-degree[11], so they have to exploit mutual following-followers relationships to achieve influence power chains[12].

This is confirmed also by the positive correlation between Depth and Closeness Centrality, where we notice how leaders usually are central nodes, capable to achieve chains of diffusion due to their shortest distances to all the other nodes (Closeness Centrality), which in turn

[10]If the new item spread with long and strong chains among the network we are looking with high probability to a future successful hit.

[11]From the heatmap we discover that in-degree and out-degree are actually positive correlated, implying that the number of followers and followings are similar among them.

[12]If a node follows increasing users, the chances are that these users follow him back and exploiting this pattern recursively, node's actions can reach indirectly a broader action range (friends of friends and so on).

expands the influence received to the rest of the network. The influential status of a central node is confirmed also by the positive correlation between Depth and Clustering, and Depth and PageRank. Respectively, this means that it is easier for a central node to be prominent if member of a tight community (users which mutually are followers and followings) and this node is labeled as central due to its number of incoming links/follower relationships. Since Strength is strongly correlated with Depth, what I've said since now for the Depth measure is worth for it too. Regarding the Width measure, I have already highlighted the positive correlation with Depth (and as a result with Strength), denoting how nodes which have a broader neighborhood have the chance to be member of hubs (positive correlation with Clustering) and as seen previously to become central and in-fluent. However, having a broader following (nodes followed by the node) circle is not associated with such great power as we will be lead to think. Indeed the anti-correlation with the Out-degree suggests us that a central nodes is capable to influence only a small fraction of his followings and only because a small amount of them are following him back. We can conclude that mutual friendships are crucial in the influential power of a node.

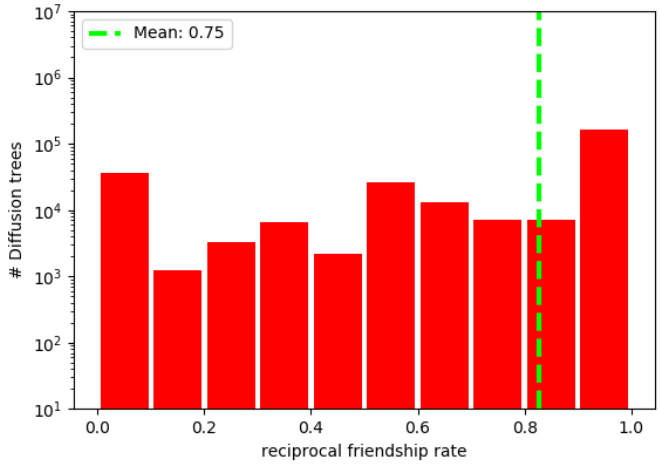

Figure 1.9: Mutual friending relationships in Diffusion Trees.

To validate this hunch, in Figure 1.9 I've compared for every leader's $T_{l,\psi}$ the number of followings (users who the leader follows) over leader's followers in $T_{l,\psi}$ (nodes who follow the leader and are influenced by him in taking action on ψ). A value equal to zero implies the absence of mutual friendship between the leader and his followers in $T_{l,\psi}$[13], while a value equal to one means a complete reciprocal following-follower relationship. We can clearly see how the most part of $T_{l,\psi}$ are involved in mutual follower-following between leaders and their first degree of influence, namely part of their followers.

Lastly, Figure 1.8 depicts also the anti-correlation between Width and In-degree, Width and Closenness Centrality and Width and PageRank, meaning in all cases that the more a node is followed and the less he will have the chance to follow back all his followers, having as a consequence a little chance to influence them all and so be central.

[13] $T_{l,\psi}$'s second level.

To sum up, the associations that I've found are the following:

1. central nodes are the one prominent in the social network (high Width and Depth);
2. be prominent among neighbors is easier if the node is in a tightly connected community;
3. longer cascades (higher Depths) are associated with a greater degree of engagement (higher Strengths);
4. only a small percentage of directed neighbors are influenced by a node;
5. the influence diffusion is maximized when exists mutual friendship relationships among the in-fluent node and the susceptible ones.

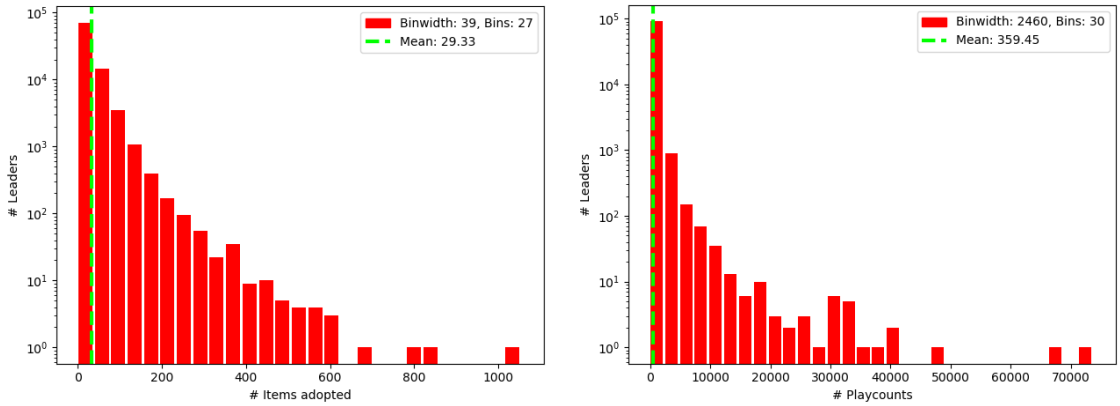

Figure 1.10: Distribution leaders per items adopted and leaders per total number of play counts.

To highlight how leader nodes don't manifest exceptional behaviours in the network[14] as well as in their choices, in Figure 1.10, I depict also their adoption and play count rate, which as we can notice resemble the global ones we had previously met.

4 Music Tags

In this section, my aim is to use the leaders found and their Width, Depth and Strength measures to characterize the spread of musical genres among the users of the Last.fm service. I recall that, as described in the Section 2, I found 91,652 leaders for 9,898 new artists, each one having a music tag corresponding to a finite number of music genres. This restricts the main music genre and it contains the following genres: 1. alternative; 2. blues; 3. classical; 4. country; 5. dance; 6. electronic; 7. hip-hop/rap; 8. jazz; 9. latin; 10. pop; 11. r&b/soul; 12. reggae; 13. rock.

The restricted main music genre set is the result of a restriction made to simplify the grouping together of the artists belonging to the same music panorama. Last.fm indeed allows assigning to each artist personal tags to describe him and his music, so the result is a

[14] A part for being central and active in their social surroundings.

messed up list of tags for every singer. In this scenario, I had first of all to outgo a mapping process and map every music tag encountered in the restrict music genre set with which I choose to work:

1. on every object $\psi \in \Psi$, I used the Last.fm API *artist.getInfo* which among artist's info returned a list of tags associated with the artist;

2. I weighted each tag in the list with the number of users that assigned the tag to the artist (information found calling the *artist.getTopTags* API);

3. I split the tags, associating the counter to each word;

4. I filtered the words referring to a musical genre, thanks to a manually constructed dictionary, built:

   - consulting the work done in [18] and [19];
   - adding the scrapped tags present at:
     - $https://www.musicgenreslist.com/$;
     - $https://en.wikipedia.org/wiki/List\_of\_music\_styles\#Avant-gard$;

   The self-built dictionary counts **1,182 tags** mapped in the 13 main music genres, adequate to classify the 9,898 artists for which I found leaders, as well as the remaining 12,103 new artists.

5. I assigned to an artist the musical genre of the survived tag with the greater relative weighted average:

$$main\_genre = max_{j=1,\dots,n}\left(\frac{tag\_counter_j}{\sum_{i=1}^{n} tag\_counter_i}\right)$$

   where n is the cardinality of each tags list.

6. to resolve the artists who had multiple greatest weighted averages, I collected all the tags present in the *artist.getTopTags*[15]. Where the indecision persisted I collected artist's similar singer tags, to widen the tags list and so to better characterize artist's music genre membership.

[15]The tags retrieved with the *artist.getInfo* were only the first five tags present in the list returned by *artist.getTopTags*.

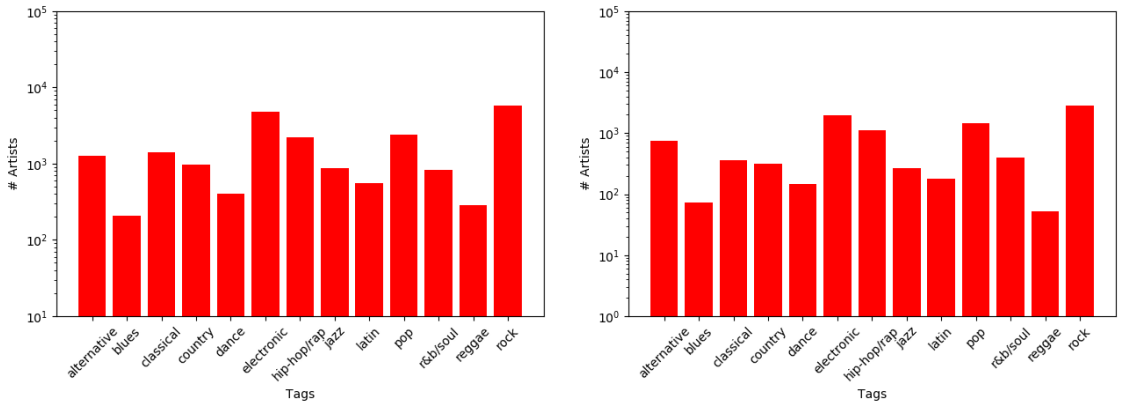

Figure 1.11: Distribution of main tags per artists (left) and distribution of main tags per artists with leaders (right).

Figure 1.11 (left) depicts the total number of artists per main tag distribution, while the result mapping considering only the artists for which I found a leader is showed in Figure 1.11 (right). In both plots the least prominent tags are reggae[16] and blues. This is not surprising, considering that these music genres involve respectively rural and niche audience. Counterpart the most popular music tags are rock, electronic, pop and hip-hop/rap. This again confirm the tendency to listen to the most well-known music genres (pop and rock), as well as the emerging hype of new ones (electronic and hip-hop/rap).

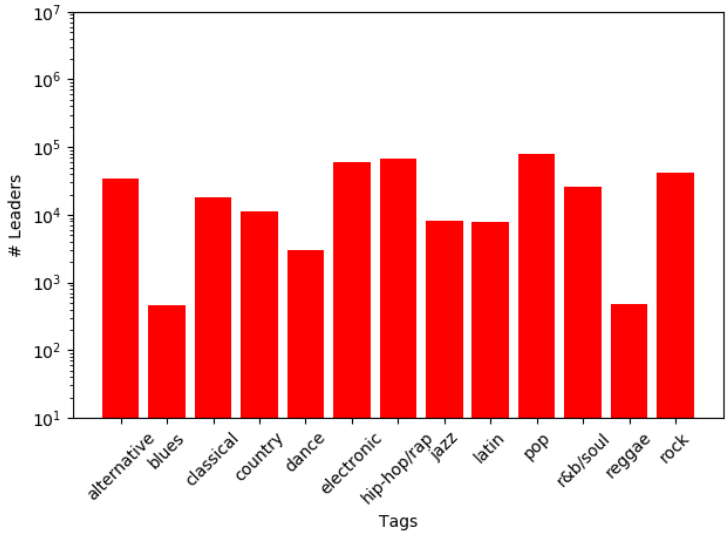

Figure 1.12: Distribution of main tags per leaders.

[16]Reggae is a music genre that originated in Jamaica in the late 1960s. The immediate origins of reggae were in ska and rocksteady; from the latter, reggae took over the use of the bass as a percussion instrument. The most famous key player in this sound is Bob Marley.

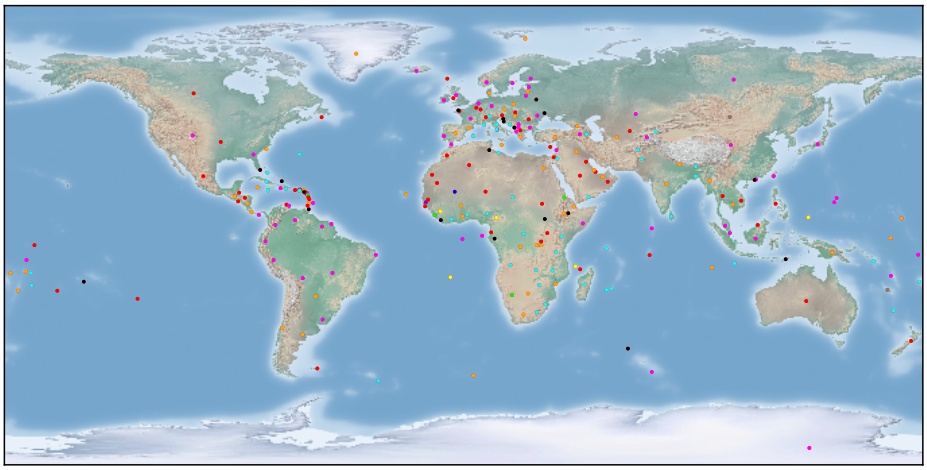

Figure 1.13: Geographical clustering based on users most listened music genre: alternative (red, 57 countries), blues (blue, 1 country), classical (orange, 52 countries), country (brown, 0 countries), dance (yellow, 5 countries), electronic (lime, 4 countries), hip-hop/rap (cyan, 46 countries), jazz (gray, 2 countries), latin (salmon, 0 countries), pop (fuchsia, 59 countries), hip-hop/rap (darkviolet, 1 country), reggae (darkgreen, 0 countries), rock (black, 21 countries).

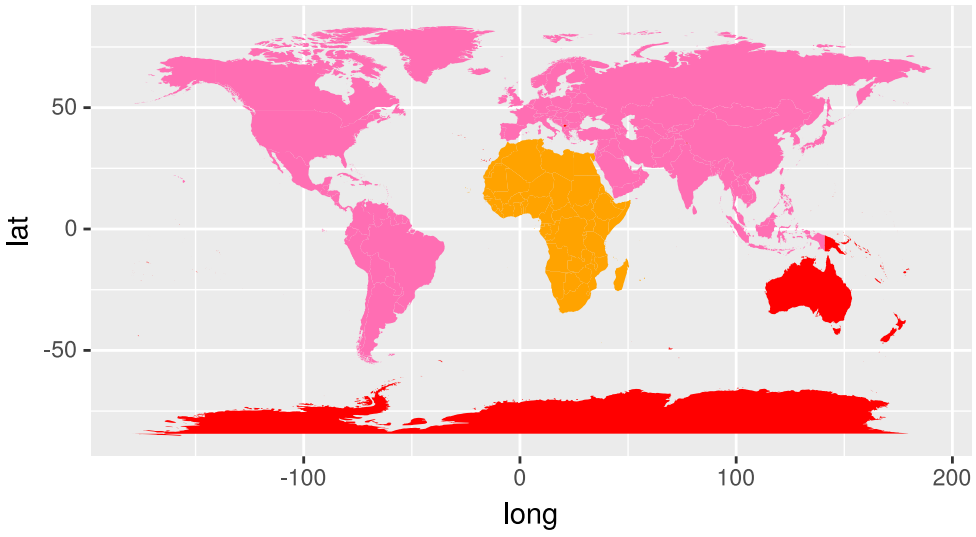

Figure 1.14: Continents clustering based on users most listened music genre: Europe (pop, 16,939 users out of 88,473), America (pop, 10,847 users out of 54,144), Africa (classical, 814 users out of 3,319), Asia (pop, 1,591 users out of 6,000), Oceania (alternative, 811 users out of 4,111), None(7,807 users out of 37,361).

Regarding the topological characteristics of Leaders per tag, Figure 1.12 shows the number of leaders (358,578 not uni-vocal leaders) influencing their neighbors for a given amount of tags. We can notice that the tags which have the greatest number of leaders are pop with 78,158 leaders and hip-hop/rap with 68,5942 leaders. This changes a little but the previous insight, where rock and electronic predominated the artists founded. Here we can respectively find 42,407 rock leaders and 59,158 electronic leaders, meaning that among the leaders that influenced their tribe with this music genres, fewer artists were in common. On the other side, pop and hip-hop/rap leaders influence many more users, but the spread artists are most of the time the same. This explains why, we depicted a lower number of artists per this tags. The least number of leaders per tag can again be found for reggae (479 leaders) and blues (464 leaders). The other tags fluctuate between 3,008 (dance) and 34,803 (alternative) leaders. Lastly, Figures 1.13 and 1.14 display countries and continents clustering, based on users' main music genre.

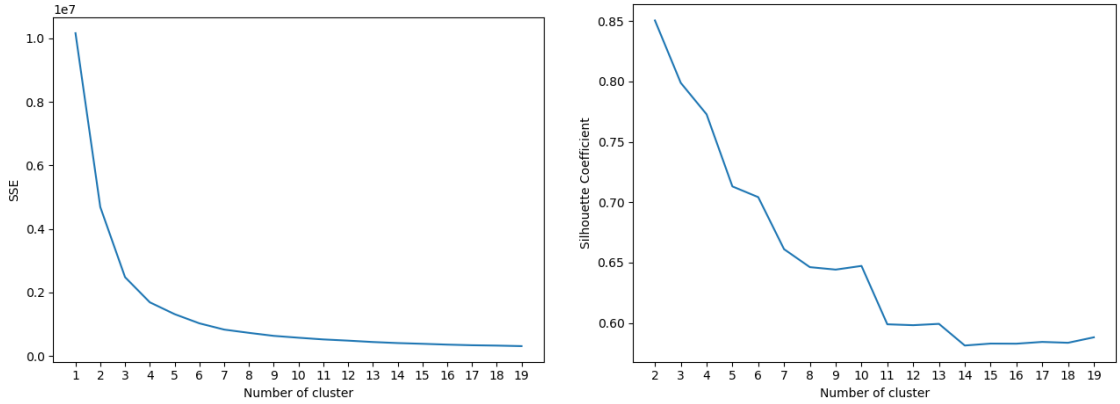

Figure 1.15: Elbow Criterion Method and Silhouette Coefficient Method to determine the optimal number of clusters for K-Means.

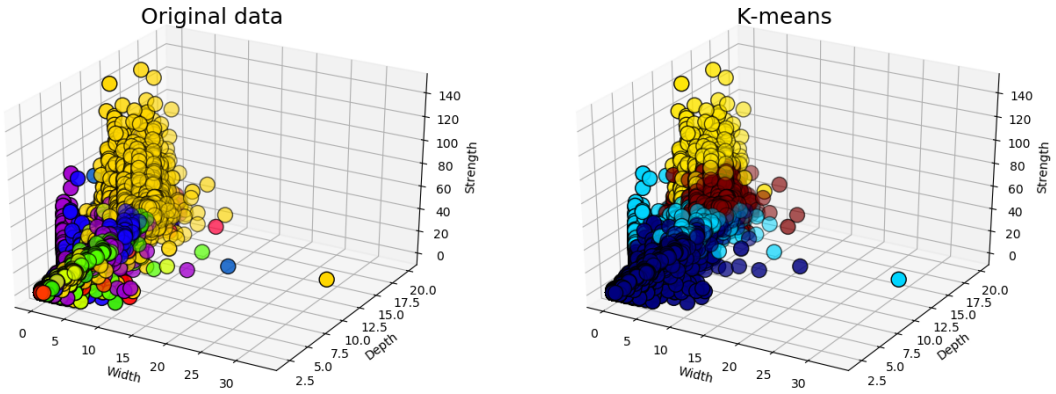

Figure 1.16: Before and after K-Means' data clustering.

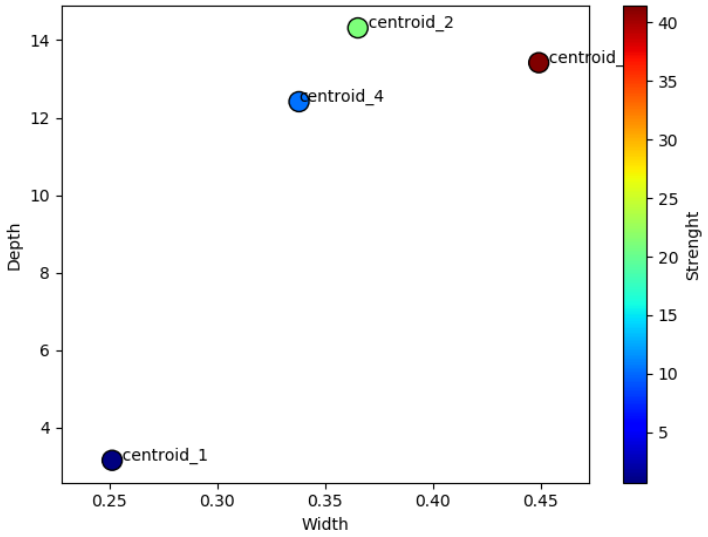

Figure 1.17: The centroids of the K-Means' clusters.

As already mentioned in Section 3, for each couple leader l and object ψ, I've calculated the Depth, Width and Strength values, as well as $T_{l,\psi}$'s cardinality $|T_{l,\psi}|$. To characterize the typical values of Width, Depth and Strength for each tag I cannot use the average or the median, because Strength and Width values are skewed (respectively by $L : T(l, \psi) \times N \to R$ and Γ_l), and it is the combination of the three measures that really characterizes the leaders. As Figure 1.16 anticipates, I cluster leaders using as features their Width, Depth, Strength and $|T_{l,\psi}|$ values. In Figure 1.15, my study of the best k to be used in K-Means[40] with the Sum of Squared Errors (SSE)[41] and Silhouette Coefficient[42] methodologies resulted in a optimal number of clusters falling in a range between 3 and 4. From Figure 1.17, instead, we can identify the clusters characterized by the highest and lowest Strength (3 and 1 respectively); by the highest and lowest Depth (3 and 1 respectively); and by the highest and lowest Width (3 and 1 respectively), as well as a mean among these three measures in cluster 4 and 2.

	cluster1 (342,639 leaders)	cluster2 (5,279 leaders)	cluster3 (1,128 leaders)	cluster4 (9,532 leaders)
alternative	**1.05**	0.00	0.00	0.001
blues	**1.05**	0.00	0.00	0.00
classical	1.04	0.00	0.00	0.09
country	1.04	0.00	0.00	0.19
dance	1.04	0.00	0.00	0.04
electronic	1.04	0.00	0.00	0.04
hip-hop/rap	0.91	**3.82**	**4.47**	2.20
jazz	1.05	0.00	0.00	0.00
latin	0.99	0.03	0.03	1.80
pop	0.99	0.48	0.47	1.53
r&b/soul	0.94	2.20	0.60	**2.42**
reggae	**1.05**	0.00	0.00	0.00
rock	1.04	0.003	0.00	0.09

Table 1.2: The RCA scores of the presence of each tag in each cluster

In Table 1.2, I report a presence score for each tag in each cluster, showing the expected number of leaders with the given tag in the cluster, a measure known as Revealed Comparative Advantage:

$$RCA = \frac{freq_{i,j}}{freq_{i,*}} / \frac{freq_{*,j}}{freq_{*,*}}$$

where i is a tag, j is a cluster, $freq_{i,j}$ is the number of leaders who spread an artist tagged with tag i that is present in cluster j.

From Table 1.2 we can see that first cluster's highest unexpected presence are alternative, blues, jazz and reggae. Almost all the diffusion trees labeled with these tags are present here, meaning that almost all alternative, blues, jazz and reggae leaders have a small tribe (16.52), depth (3.16), width (0.25), and strength (0.71), suggesting that for these music genres is not easy to be prominent. In the fourth and second clusters hip-hop stands out meaning that

among all the other music genres it is the one with the tribe (882.12 and 2363.30), depth (12.41 and 14.32), width (0.36 and 0.38) and strength (10.29 and 21.07) more on average. The third and last cluster is dominated by r&b/soul, which overtakes hip-hop/rap's power-ship. All the diffusion trees belonging to leaders who spread r&b/soul artists had the most tribe (5038.14), depth (13.42), width (0.45) and strength (41.42) influence power. As a result, we can conclude that prominent leaders for r&b/soul artists are embedded in groups of users very engaged with the new artist and likely they will be prominent among their friends too. To verify this intuition, using again leaders' Minimum Diffusion Trees $T_{l,\psi}$, I extract some patterns that help me to obtain a complementary point of view over Leader's prominence for different music genres.

		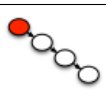	
alternative	7.42% (30.56%)	14.32% (58.98%)	15.63% (64.40%)
blues	8.62% (41.00%)	8.84% (40.00%)	10.99% (51.00%)
classical	7.39% (30.31%)	15.35% (62.95%)	16.55% (67.90%)
country	13.87% (36.50%)	26.55% (69.85%)	28.05% (73.78%)
dance	13.60% (40.61%)	24.37 (72.79%)	25.36% (75.77%)
electronic	9.89% (34.96%)	18.03% (63.74%)	19.10% (67.51%)
hip-hop/rap	**38.84% (38.84%)**	79.96% (79.96%)	79.36% (79.36%)
jazz	8.13% (33.37%)	14.30% (58.66%)	15.66% (64.26%)
latin	17.71% (36.48%)	39.19% (80.71%)	39.03% (80.39%)
pop	19.21% (40.95%)	37.78% (80.52%)	37.84% 80.64%)
r&b/soul	37.96% (37.96%)	**81.98% (81.98%)**	**80.71% (80.71%)**
reggae	8.35% (33.90%)	10.85% (44.07%)	13.15% (53.39%)
rock	9.64% (34.19%)	16.82% (59.63%)	18.33% (64.99%)

Table 1.3: Presence of different diffusion patterns per tag

I mine a graph dataset composed of all diffusion trees $T_{l,\psi}$ with the VF2 algorithm[38]. I used the VF2 algorithm's query processing to find all graphs in my graph collection that contain similar subgraphs to the following query graphs:

(a) a star-like pattern where a leader influences three of its neighbors;

(b) a chain where each node is prominent for (at least) one neighbor;

(c) a split where the leader is prominent for a node, which itself is prominent for two other neighbors.

For each diffusion tree $T_{l,\psi}$, when analyzed, I've used a simple pruning technique, to reduce the VF2's search space:

(a) for the star-like pattern I hold the nodes until $T_{l,\psi}$'s second level[17];

(b) for the chain pattern I hold the nodes until $T_{l,\psi}$'s fourth level;

(c) for the split pattern I hold the nodes until $T_{l,\psi}$'s third level.

In Table 1.3 I report the results of mining the three patterns described above. Two values are associated to each pattern and tag pair: the relative frequency considering all the trees clustered with the same music genre and the relative frequency considering only the trees with at least four nodes (in parentheses), clustered always with the same music genre. As Table 1.3 highlights, r&b/soul leaders achieve long and strong cascades (predominance in the chain and split patterns), confirming the dominance in the third clustered depicted in Table 1.2. Similar to r&b/soul leaders are the hip-hop/rap ones, placing second in the chain and split patterns, but first in the star pattern (they influence more directed friends than r&b/soul predominant users). However, despite a broader friendship network influence, hip-hop/rap leaders achieve less longer and powerful cascades. This phenomenon suggests us that r&b/soul leaders tend to be prominent for nodes strongly devoted to r&b/soul, inducing them to spread the music to their neighbors. Hip-hop/rap leaders, on the other hand, affect more neighbors, presumably flooding their ego networks with the songs they like, but the influence among the network is a little bit less powerful. Considering the others music tags, we can see that there is no necessary relation between the patterns and Width, Depth and Strength measures. For example we can observe that blues, jazz and reggae (which are completely absent in the fourth cluster) and alternative, classical, dance, electronic and rock (only slightly present in the same most "in-fluent" cluster), although having leaders in the fourth and second mean valued clusters, are mostly predominant in the first cluster. Moreover they show how even having a low Width, Depth and Strength still makes possible to achieve chains (chain and split motifs, respectively) and to influence at least three directed neighbors (star motif). This makes clear, again, how it is the combination of the three dimensions of social prominence that really characterizes the prestige of a node in a network.

[17]The level of a node is defined by 1 plus the number of connections between the node and the root.

Chapter 2

Hit-Savvies detection

5 Hit-Savvies detection

My second contribution in the leader detection quest is depicted from [3]. The paper focuses on finding this peculiar class of users by looking at their adoptions and measuring their tendency to adopt Hits (successful items) and Flops (unsuccessful items). As the common intuition suggests, the greater number of Hits a user adopts, the more likely important he is, but to be influential it is mandatory to do so before others do. Furthermore, since an item's success definition is not easy to quantify[1], first of all is important to fix an uni-vocal definition. I've experimented and compared a few of them:

- success based on artists' seed user listeners;
- success based on artists' seed user play-counts;
- success based on artists' Last.fm listeners;
- success based on artists' Last.fm play-counts;
- success based on artists' Google searches[2];
- success based on seed users' actions[3];
- success based on how artists' adoptions spread over time.

The first six definitions are volume based, while the seventh is more data-driven and describes how items' adoption unfolds through time. We can define it as:

Definition 6 (*Adoption trend*):
Given an item g its adoption trend τ is a time series in which $\tau(t)$ identifies the percentage x of the total adoptions of g occurred at time t.

[1] Several factors both exogenous (i.e., artist's social impact) and endogenous (i.e., purchases' volumes.) can be used to measure it.

[2] I've scrapped the information from *Google Trends* [20].

[3] Counting each user adoption slot.

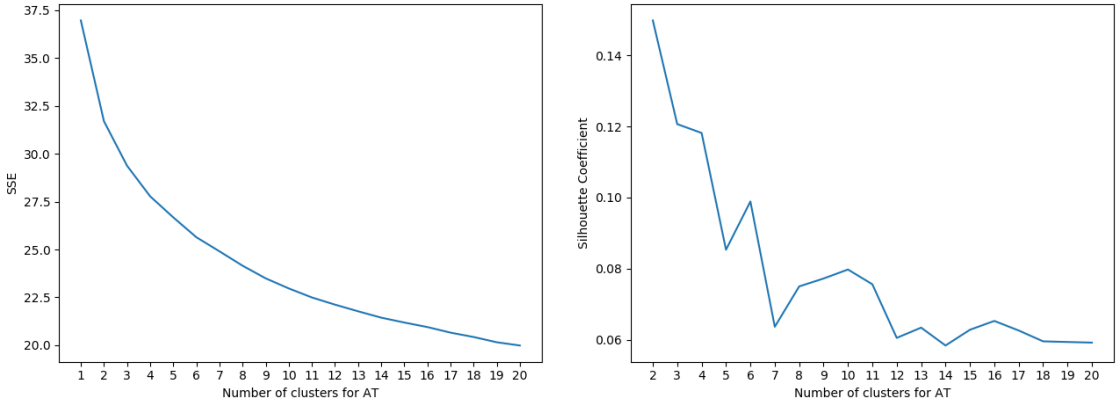

Figure 2.1: Elbow Criterion Method and Silhouette Coefficient Method to determine the optimal number of clusters for K-Means with Dynamic Time Warping.

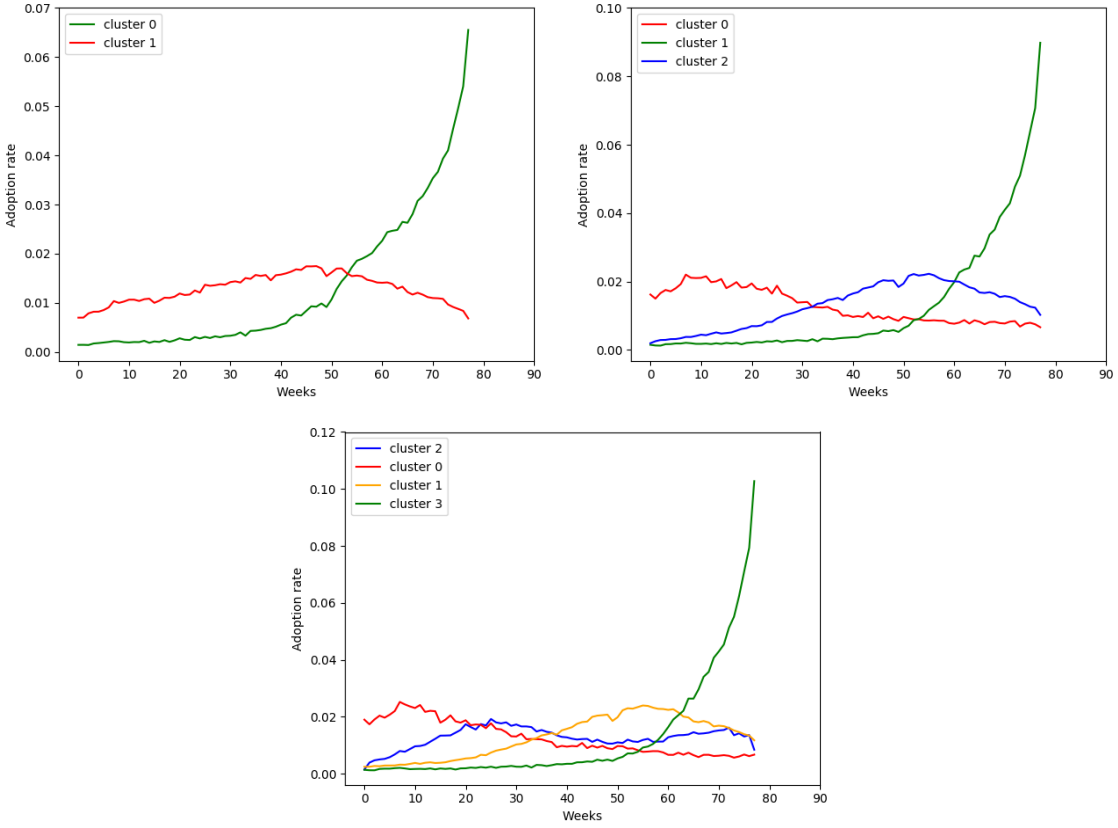

Figure 2.2: Adoption trends cluster medoids for k = 2, k = 3 and k = 4.

For an artist its adoption trend describes, for each week, the percentage of listeners that listened to him for the first time. I clustered each item's observed adoption trend τ with K-Means to profile them and identify recurrent patterns. I've used K-Means clustering with Dynamic Time Warping[43] as the distance function, since the Euclidean distances between alignments are much less susceptible to pessimistic similarity measurements due to distortion in the time axis. From Figure 2.1, we see that the study of cluster quality through SSE identifies k = 2 as the optimal K-Means parameter value. The medoids of the two clusters describe peculiar shapes: one expressing a drop of the adoption rate, the other capturing an expanding trend. Figure 2.2 shows the medoids obtained and in the same figure, I also report the medoids for k = 3 and k = 4. In the latter scenarios, the additional profiles identify only specializations of the decreasing medoid obtained with the optimal solution. Besides, to avoid biases while computing trend clusters, in the first phase I employed only those items that were continuously adopted for at least half of the observed period (half of 18 months, i.e. 9 months). Still, since the resulting number of increasing trend items was very low, I proceeded to cluster all the items without observing changes in the metoids shape.

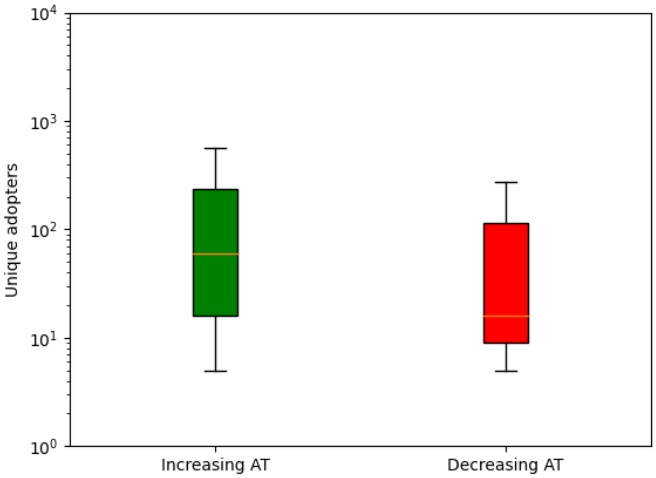

Figure 2.3: Comparison of volumes of expanding and contracting adoption trends.

To understand if exists a correlation between the shape of the adoption trend of one item and its adoption volume, I also analyzed how volumes distribute in the two clusters. Figure 2.3 shows how items having expanding trends and items having decreasing trends don't differ much in the adoption volume, but if the first increases as the time goes by, the second manifest a sudden decreasing. In conclusion, the data-driven definition of success discriminates successful and unsuccessful items by the shape of their adoption trends:

Definition 7 (*Successful trends*):
Successful trends are the ones describing an increase of adoption rate through time capturing an expansion of individual items' adopter base.

Definition 8 (*Unsuccessful trends*):
Unsuccessful trends are the ones in which the adoption rate does not increase considerably over time or even reach an early maximum only to start to decrease rapidly.

An important thing to underline is that the definition of success based on an increasing or decreasing of adoption trend over time, can be considered universal and transferable across different contexts, without taking into account the items and their semantic correlation. Independently from the success definition used, in the following, I will call **Hits** the successful items and **Flops** the unsuccessful ones. Moreover, I identify with $\mathbf{H} \subset \Psi$ the set of Hits and with $\mathbf{F} \subset \Psi$ the set of Flops. Returning on the problem of segmenting the customers of an item to identify its early adopters, I thus define:

Definition 9 (*Innovators*):
Given an adoption trend τ of an item g we consider innovators all those adopters that adopt g strictly before the global maxima of τ.

Once identified the innovators, to exploit the predictive model proposed in [3], I need to quantify their intrinsic individual propensity to adopt Hits or Flops. To do so I define for each user its HF-propensity as:

Definition 10 (*HF-propensity*):
Given an adopter a, f the number of Flops he adopted, h the number of the Hits he adopted as innovator and k the number of Hits he adopted after they reach their global maxima, the HF-propensity of a is:

$$HF(a) = \frac{h - k - f}{h + k + f}$$

HF-propensity lies between [-1, 1]. Its value is maximised when a adopts only Hits as innovator, minimised when his adoptions regards only Flops or latecomer Hits . Adopters having HF = 0 are considered neutral signals since they do not show any special propensity toward neither Hits nor Flops. In the following analysis, I will discard neutral adopters since they cannot be considered discriminatory indicators. Indeed, since the HF-measure is not weighted on the number of adoptions, an adopter that has adopted a single successful (unsuccessful) item will receive the highest (lowest) value in the range. An adopter must be considered an innovator even if he only adopted a single item among thousands available if, later on, such item reached success.

Definition 11 (*Hit-Savvy*):
Given a set of successful items $H \subset \Psi$ adopted by some adopters $a \in V' \subset V$, I define Hit-Savvy all those adopters $a \in V'' \subset V' \subset V$ for which HF(a) > 0.

I'll address with **HT** the set of Hit-Savvy and with **FT** the set of Flop-adopters (e.g. adopters having HF < 0).

Success def.	$\|HT\|$	$\|H\|$	Coverage
seed-user listeners	2,338 (1.21%)	2,046 (9.30%)	24.40%
seed-user play-counts	2,307 (1.19%)	2,002 (9.10%)	23.87%
google trends	1,072 (0.55%)	952 (4,33%)	11.35%
last.fm listeners	1,864 (0.96%)	1,386 (6,30%)	16.53%
last.fm play-counts	1,933 (1.0%)	1,417 (6,44%)	16.90%
AT	822 (0.42%)	715 (3,25%)	8.53%
adopters' actions	2,346 (1.21%)	2,077 (9,44%)	24.77%

Table 2.1: Hit-Savvies statistics for each success definition: number of Hit-Savvies $|HT|$, number of successful items $|H|$ and the coverage induced on them by the Hit-Savvies. Within bracket are reported their percentage over all the adopters and items respectively.

To better highlight the impact different success definitions have on the HT set, in Table 2.1, I compared the seven success definitions previously mentioned: six based on volume, that define as Hits respectively the top-38 %[4] most adopted items based on the number of seed play-counts, seed listeners, Last.fm play-counts, Last.fm listeners, adopters' actions, Google searches and the last base on the AT, the data-driven trend-based previously introduced. We can observe that, disregarding the data set and the success definition used, HF-propensity often identifies as Hit-Savvies less than 1% of the total adopters and the induced coverage fluctuate across 8,53% and 24,77%. This low Coverage percentage brings out an interesting finding: among all success definitions, the majority of hits $\in$ H $\subset \Psi$ aren't adopted by Hit-Savvies, but instead by Flop-adopters. This means that many Flop-adopters are initially potential Hit-Savvy, but by adopting to a greater extent late Hits and/or Flops, they reverse their attitude and become Flop-adopters. This also results in a Flop-adopters' less strong HF-propensity, a scenario which will lead to a predictive model not so accurate, as we will later discover.

[4]38 % was chosen due to relative percentage of items with increasing AT over all the items adopted and found with the previously described clustering.

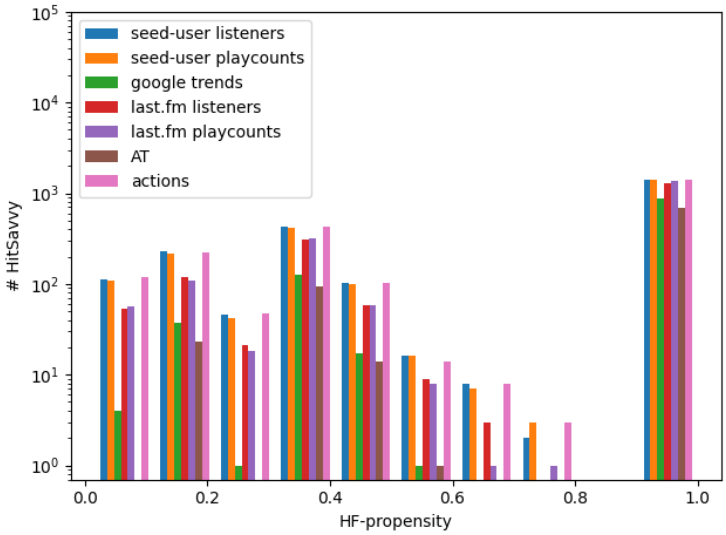

Figure 2.4: Hit-Savvies HF-propensity distribution.

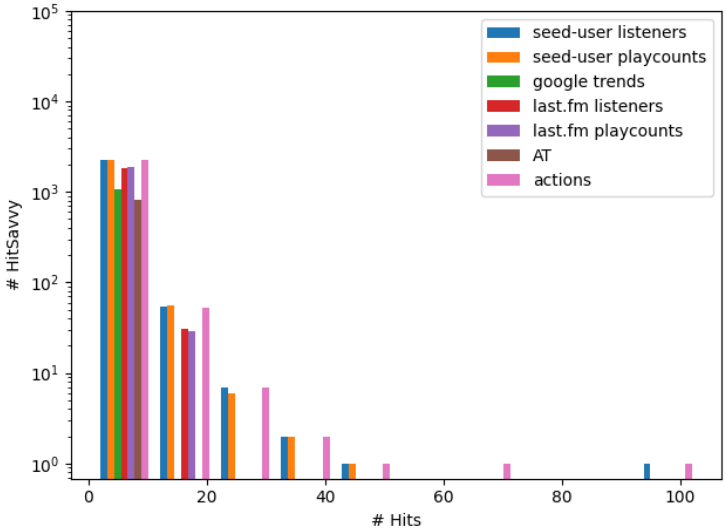

Figure 2.5: Hit-Savvies hits adoption distribution.

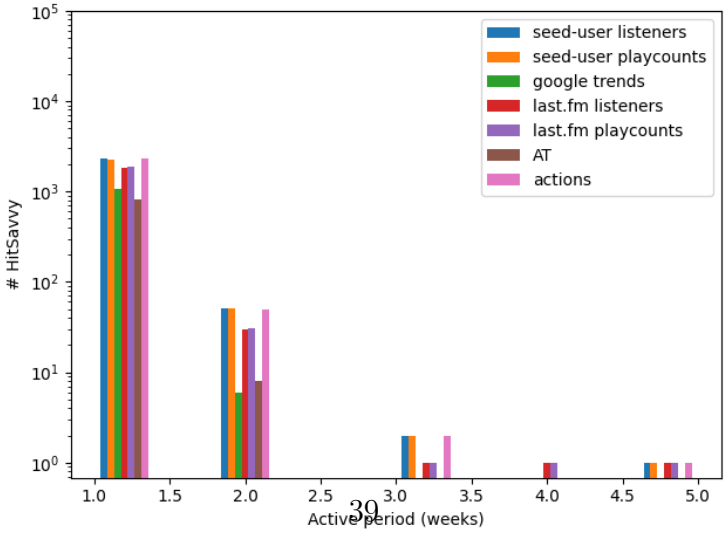

Figure 2.6: Hit-Savvies active period distribution.

In Fig 2.4 I show the distributions of the HF-propensity for the identified Hit-Savvies while varying the success definition. A neat peak for high HF-propensity scores emerges (in the range (0.9, 1.0]), implying that it is quite common to identify within Hit-Savvies that adopt exclusively Hits, but like the Figure 2.5 depicts this scenario is possible only because Hit-Savvies adopt a low number of successful items. In fact most Hit-Savvies adopt between 1 to 3 successful items and as we will later see, this will again negatively impact the predictive model. To understand if a Hit-Savvy adopts Hits only a few random times across his history or stably performs prevalent Hit adoptions as an innovator, I computed his HF-propensity on a weekly basis.

Definition 12 (*Hit-Savvy's active period*):
Given an Hit-Savvy a, I define his active period as the average number of consecutive weeks having HF-propensity greater than 0:

$$HF(a) = [HF_{t_0}(a), ..., HF_{t_n}(a)]$$

with [t_0 , . . ., t_n] the ordered weeks in which a makes at least an adoption.

Figure 2.6 shows the probability distribution of Hit-Savvies' active periods computed varying the success definitions. Two clear patterns emerge:

1. the majority of Hit-Savvies aren't consistent and make a few, sparse adoption over time. This, in mixture with the low number of global adoptions, results in an low average active period of one week;

2. only the success definition based on listeners, play-counts and actions are able to guarantee the identification of Hit-Savvies with longer active periods (even if still restricted up until 5 consecutive weeks).

	Def1	**Def2**	**Def3**	**Def4**	**Def5**	**Def6**	**Def7**
Def1	-	0.87	0.22	0.52	0.50	0.19	0.94
Def2	0.87	-	0.22	0.53	0.53	0.20	0.89
Def3	0.22	0.22	-	0.24	0.25	0.18	0.22
Def4	0.52	0.53	0.24	-	0.69	0.19	0.53
Def5	0.50	0.53	0.25	0.69	-	0.20	0.51
Def6	0.19	0.20	0.18	0.19	0.20	-	0.19
Def7	0.94	0.89	0.22	0.52	0.51	0.19	-

Table 2.2: Jaccard similarity among Hit-Savvies found using different success definitions

	Def1	Def2	Def3	Def4	Def5	Def6	Def7
Def1	-	0.80	0.16	0.33	0.50	0.15	0.91
Def2	0.80	-	0.16	0.37	0.36	0.16	0.82
Def3	0.16	0.16	-	0.21	0.22	0.16	0.16
Def4	0.36	0.37	0.21	-	0.58	0.17	0.37
Def5	0.33	0.36	0.21	0.58	-	0.18	0.34
Def6	0.15	0.16	0.16	0.17	0.18	-	0.16
Def7	0.91	0.82	0.16	0.37	0.34	0.16	-

Table 2.3: Jaccard similarity among Hits covered by the Hit-Savvies found using different success definitions.

Lastly, Tables 2.2 and 2.3 compare the set of Hit-Savvies and Hits covered by the different success definitions, encoded from latter on as following:

- **Def1**: seed-user listeners;
- **Def2**: seed-user play-counts;
- **Def3**: google trends;
- **Def4**: last.fm listeners;
- **Def5**: last.fm pray-counts;
- **Def6**: AT;
- **Def7**: adopters' actions.

We can conclude asserting that different definitions of success generate different sets of Hit-Savvies and Hits that vary both in volume and stability. Volume-based definitions are quite similar among them, with the exception of the Def3 (Google trend based). On the other hand, the previously introduced AT method generates the most different set of Hit-Savvies and Hits, but unfortunately it shares the common drawback among all the definition analyzed: the reduced set of positive, successful items they define causes the identification of very few and volatile Hit-Savvy. Such reduced set of special users are usually not enough to make high accurate predictions, as I'll explain in the next section.

6 Hits&Flops: Forecast model

In this Section, I will try to exploit the previously found Hit-Savvies, to understand if they can be fruitfully used to forecast novel items success. So far I've measured adopters' propensity towards Hits and Flops and thus transformed them into indicators. For my first forecasting approach I've used directly the HF-propensity scores, building a simple classifier that exploits these indicators and combining them linearly: in such scenario, a novel item is labeled as Hit if the sum of the HF-propensities of its adopters is greater than 0, Flop otherwise.

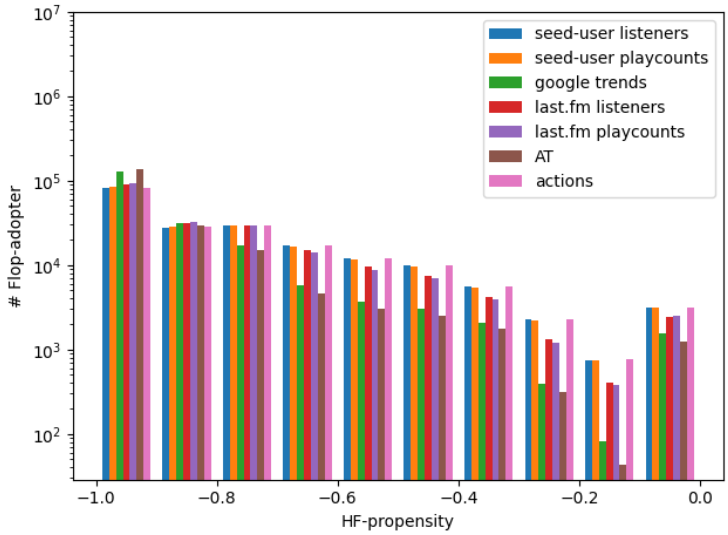

Figure 2.7: Flop-Adopters HF-propensity distribution.

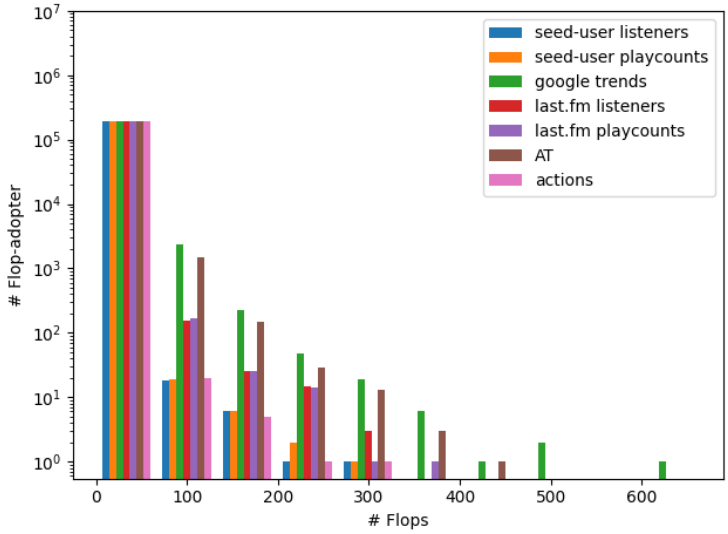

Figure 2.8: Flop-Adopters flops adoption distribution.

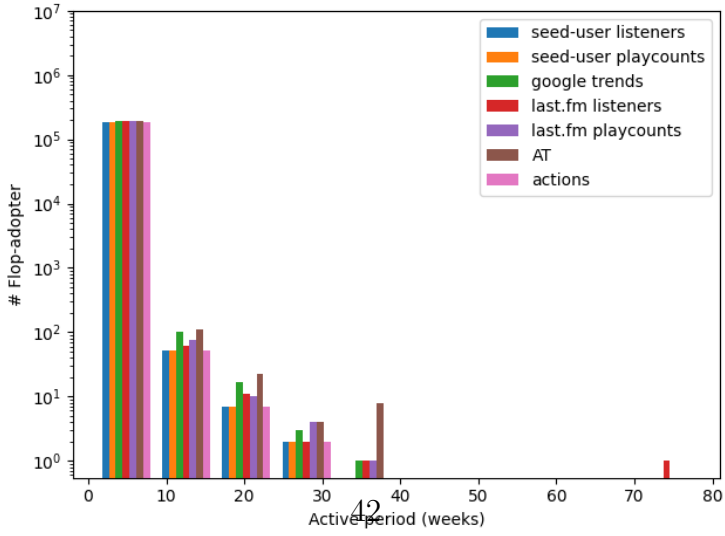

Figure 2.9: Flop-adopters active period distribution.

Unfortunately, such approach is not able to make predictions for Hits due to the highly unbalance among the computed indicators (Hit-Savvies are always around 1% of total adopters). Such low performances are due to the HF-propensity distribution for Floppers which tends, conversely from Hitters, to range over all the values, to adopt on average more flop-items and in general to be more active (they are constantly adopting artists, which are listened only by them or by a restrict circle and for a short period of time, otherwise they'll be classified as Hits), as shown in Figures 2.7, 2.8 and 2.9. [3] addresses this imbalance, targeting the minimum sets of Hit-Savvies as well as of Flop-adopters able to discriminate both Hits and Flops, to reduce noises. The task of identifying such subsets can be formulated as a Weighted Multi-Set Coverage (WMSC) problem[44]. To comply with WMSC, I built two bipartite graphs: one connecting each Hit-Savvy in HT to the successful items he adopts, and the other connecting each Flop-adopter in FT to his unsuccessful items. On each of such topologies, I then searched for the subset of adopters that guaranteed a constrained minimum coverage of the connected items, according to the following WMSC formulation:

$$
\begin{aligned}
& \mathbf{min}_{x_i,\ldots,x_{|X|}} && \sum_i x_i \\
& \textbf{\textit{subject to}} && \forall\ i,\ j : x_i = e_{i,j} \\
& && \forall j : \sum_i e_{i,j} \geq \gamma_j \beta \\
& && \sum_j y_j \geq \alpha |Y| \\
& \textbf{\textit{where}} && x_i,\ y_j,\ e_{i,j} \in \{0,1\}, \alpha, \beta \in (0,1]
\end{aligned}
$$

where x_i , y_j , $e_{i,j}$ are binary variables modelling each potential adopter (belonging to the adopter set X), item adopted (belonging to the item set Y), and the edge of the bipartite graph. The first constraint ensures that a selected adopter contributes to the coverage of each of the item he adopts. The second constraint ensures that selected adopters cover at least β% of the sum of all incoming edges to each item. The third constraint ensures that the selected adopters collectively cover at least α% of the set of covered items. The greedy approach to solve WMSC (Algorithm 1) selects a subset of adopters which satisfies the α and β thresholds. It also takes care to sort the adopters to be selected for the covering:

- maximizing their degree and HF-propensity to cover Hits;
- maximizing their degree and minimizing their HF-propensity to cover Flops.

To facilitate readability, in the pseudo-code I make use of the following notation:

- $a \in$ V identifies an adopter;
- $g \in \Psi$ identifies an adopted item;
- $A_* \subset$ V and $G_* \subset \Psi$ two sets;
- ω(a, G_*) identifies the set of items in G_* adopted by $a \in$ V;

- θ(g, A_*) identifies the adopters of $g \in \Psi$ within A_*;
- c is a sorting strategy based on degree and HF-Propensity;
- α is a global coverage threshold;
- β is a local coverage threshold.

Listing 2.1: WMSC greedy

```
Indicators = []   covered_items = []
A_s = sort(A, c)
for a ∈ A_s do
    if |covered_items| < α |G_*| then
        G_a = ω(a, G_*)
        for g ∈ G_a do
            if |θ(g, Indicators)| < β degree(g) then
                if a ∉ Indicators then
                    Indicators.add(a)
                if g ∉ covered_items then
                    covered_items.add(g)
    else
        return Indicators
return []
```

Since the α and β parameters deeply affect the coverage produced by WMSC, to identify the optimal HT and FT subsets I instantiate multiple runs, ranging from 10% to 100%. Such strategy results in two sets of solutions:

1. one for Hit-coverages $HC = \{HT_0, \ldots, HT_n\}$, with $HT_0, \ldots, HT_n \subset HT$;
2. the other for Flop-coverages $FC = \{FT_0, \ldots, FT_n\}$, with $FT_0, \ldots, FT_n \subset FT$.

Given a valid solution $HT_i \in HC$, let $H_i \subset H$ be the subset of items covered by adopters in HT_i . Symmetrically, given a valid solution $FT_j \in FC$, let $F_j \subset$ F be the subset of items covered by adopters in FT_j. Among the solutions within HC and FC, I select HT_i and FT_j that maximise $|H_i|$ and $|F_j|$ while having minimal cardinality: I call them $\widehat{HT}$ and $\widehat{FT}$. I call **Hitters** all the filtered and reliable Hit-Savvies within $\widehat{HT}$ and symmetrically **Floppers** all the Flop-adopters in $\widehat{FT}$. Once learned the Hitters and Floppers to predict whether a novel item will be either a Hit or a Flop I proceed as follows:

1. for each successful item g $\in$ H let $\pi(g)$ be the number of its adopters in $\widehat{HT}$ and Π the probability distribution of $\pi(g)$ across all the items in H;
2. symmetrically, for each unsuccessful item g $\in$ F let $\delta(g)$ be the number of its adopters in $\widehat{FT}$ and Δ the probability distribution of $\delta(g)$ across all the items in F.

We call $\widehat{m}$ ($\widehat{n}$) the mean values of the two distributions Π and Δ, used as threshold over Π and Δ by the predictive model Hits&Flops, defined by this two step procedures:

1. **Specialized classification**. Two distinct classifiers are built:
   - one to forecast Hits (leveraging $\widehat{HT}$ set and Π distribution);
   - one to forecast Flops (leveraging $\widehat{FT}$ set and Δ distribution)
2. **Synthesis**. A meta-classifier combines the predictions of specialized classifiers and generates the final prediction for each novel item.

Let g_* be a previously unseen item and A_* the set of its adopters during a limited time window starting from its first appearance: the rationale behind the specialized predictors is the following:

Definition 13 (*Hits-classifier*):
The hit classifier H_p returns a positive class (Hit) iff $\pi(g_*) > \widehat{m}$, where $\pi(g_*)$ is computed over $A_* \cap \widehat{HT}$.

Definition 14 (*Flops-classifier*):
The flop classifier F_p returns a positive class (Flop) iff $\delta(g_*) > \widehat{n}$, where $\delta(g_*)$ is computed over $A_* \cap \widehat{FT}$.

Since those classifiers give only partial predictions their outcomes need to be combined by a meta-classifier:

Definition 15 (*Hits&Flops*):
Given the Hits and Flops predictors H_p and F_p, we define the rule based meta-classifier as:

1. *if $H_p(g_*) \rightarrow$ Hit and $F_p(g_*) \rightarrow$ Hit, g_* is a future Hit;*
2. *if $H_p(g_*) \rightarrow$ Flop and $F_p(g_*) \rightarrow$ Flop, g_* is a future Flop;*
3. *if $H_p(g_*) \rightarrow$ Hit and $F_p(g_*) \rightarrow$ Flop:*
   (a) *if $\pi(g_*) - \widehat{m} > \delta(g_*) - \widehat{n}$, g_* is a future Hit;*
   (b) *if $\pi(g_*) - \widehat{m} = \delta(g_*) - \widehat{n}$, the observed adoptions are still not enough to make a forecast;*
   (c) *otherwise, g_* is a future Flop;*
4. *otherwise, the observed adoptions are still not enough to make a forecast.*

The first (second) rule identifies an item g_* having $\pi(g_*)$ at the right (left) of $\widehat{m}$, and $\delta(g_*)$ at the left (right) of $\widehat{n}$. In this cases the meta-classifier returns the positive class identified by the corresponding classifier. The third rule models the situation in which both the classifiers return a positive class for the item g_*. In this case, the meta-classifier will return the class identified by the greater distance between the thresholds and the corresponding value $\pi(g_*)$ and $\delta(g_*)$ and in case the two distances are equivalent, it will not return a classification. Finally, the fourth rule models the case in which both the classifiers do not return a positive guess. In this case, a classification is not provided: since we are dealing with partial observations I choose not to classify the instances for which we do not have enough data.

7 Hits&Flops: Evaluation

Now that I've introduced the theory behind the forecast model, I'll describe how I actually exploited it to forecast the success of novel, unseen artists. Most importantly, in next Section, I'll illustrate a parallelism and some common traits among the leaders and Hit_Savvies found, giving a more in depth understanding of the insight behind the influence and predictive power of such significant users. First of all, to make reliable predictions, I've studied the effect that time had on the predictive model, to understand for how long I needed to observe the listeners of new items to make a reliable prediction. I then instantiated multiple runs with the predictive model Hits&Flops over my data set, applying increasingly longer observation periods. I managed these multiple runs using an incremental temporal window: since my data set had almost 18 months available, I've used 3, 6, 9, 12 and 15 months to learn the model on those artists (and related listening) that appeared during this time and forecast the success of the artists listened for the first time in the remaining period (15, 12, 9, 6 and 3 months respectively). The forecast on the artists adopted in the test set (remaining period) was done ranging the observation period from one month to a year, to establish the best predictive window for each split. My results can be found here [5], where with the following notation I define respectively:

$$PPV = (PositivePredictiveValue/precision) = \frac{TP}{TP + FP}$$

$$NPV = (NegativePredictiveValue) = \frac{TN}{TN + FN}$$

$$TPR = (TruePositiveRate/recall) = \frac{TP}{TP + FN}$$

$$TNR = (TrueNegativeRate/specificity) = \frac{TN}{TN + FP}$$

with the binary classification defined as TP = True Positive, FP = False Positive, TN = True Negative , FN = False Negative.

More over, I identify my splits as following:

- **split a**: 3 months of training, 15 months of test
- **split b**: 6 months of training, 12 months of test
- **split c**: 9 months of training, 9 months of test
- **split d**: 6 months of training, 6 months of test
- **split e**: 15 months of training, 3 months of test

	Def1	**Def2**	**Def3**	**Def4**	**Def5**	**Def6**	**Def7**
Def1	-	0.04	0.33	0.19	0.21	0.24	0.02
Def2	0.04	-	0.32	0.20	0.19	0.24	0.03
Def3	0.33	0.32	-	0.28	0.28	0.30	0.32
Def4	0.19	0.20	0.28	-	0.09	0.27	0.29
Def5	0.21	0.19	0.28	0.09	-	0.27	0.20
Def6	0.24	0.24	0.30	0.27	0.27	-	0.24
Def7	0.02	0.03	0.32	0.19	0.20	0.24	-

Table 2.4: Jaccard similarity among Hit set and Flop set defined using different success definitions.

	Def1	**Def2**	**Def3**	**Def4**	**Def5**	**Def6**	**Def7**
Def1	-	0.80	0.21	0.40	0.38	0.33	0.90
Def2	0.80	-	0.22	0.39	0.40	0.33	0.85
Def3	0.21	0.22	-	0.27	0.27	0.24	0.24
Def4	0.40	0.39	0.27	-	0.63	0.28	0.40
Def5	0.38	0.40	0.27	0.63	-	0.28	0.39
Def6	0.33	0.33	0.24	0.28	0.28	-	0.32
Def7	0.90	0.85	0.24	0.40	0.39	0.32	-

Table 2.5: Jaccard similarity among Hit sets defined using different success definitions.

	Def1	**Def2**	**Def3**	**Def4**	**Def5**	**Def6**	**Def7**
Def1	-	0.88	0.43	0.58	0.56	0.52	0.94
Def2	0.88	-	0.43	0.58	0.58	0.53	0.90
Def3	0.43	0.43	-	0.48	0.48	0.45	0.43
Def4	0.40	0.58	0.48	-	0.76	0.48	0.58
Def5	0.38	0.58	0.48	0.76	-	0.49	0.57
Def6	0.33	0.53	0.45	0.48	0.49	-	0.52
Def7	0.90	0.90	0.43	0.58	0.57	0.52	-

Table 2.6: Jaccard similarity among Flop sets defined using different success definitions.

We can clearly notice how the success definition based on Google's greatest number of searches, as well as the increasing AT don't act very well in respect of the predictive model. The Google Trend success definition don't fit the real adoptions made by the user in my data set and the low scores underline this pattern[5]. On the other hand, using the AT success definition, I had expected quite good results since the increasing number of adopters over time could well fit the idea of a successful item, but contrarily to my initial forecasts the definition still poorly performed. I suppose that behind these low scores we can find the following issues:

1. recalling Figure 2.3, I didn't found a huge difference among increasing and decreasing AT adoption volumes, meaning the only discriminant for an item to be considered a Hit or a Flop is determined by the adoption rate curve over time, continuously increasing for increasing AT (and so for Hits), increasing and at a certain point decreasing for decreasing AT (and so for Flops), with the number of new adopters reaming quite similar. This makes me suggest that maybe the decreasing AT artists are wrongly classified as Flops, because they can still have a wide range of occasional new listeners, which for a period of weeks and months made them reach notoriety[6]. We can notice, for instance, observing the column relative to the Def6 in Table 2.4, how a percentage of Flops identified by the AT success definition are present in the Hits set of the other success definitions. Always looking at this Table, we can underline how artists' classification based on Last.fm number of listeners and play-counts (Def4 and Def5, respectively) are similar among them (this is confirmed also by Tables 2.5 and 2.6) but quite different from the ones related to my adoption log (Def1, Def2 and Def7), similar in their turn among them: this discrepancy with my adoption data makes them perform not with accurate scores, too.

2. the cardinality of the HitSavvy set may be too small (0,42% out of all adopters) to make meaningful Hit predictions since the low number of Hitters (even if we select them among all as the whole HitSavvy set) are swollen by the greatest cardinality of the Flop-adopters and by its subset of Floppers;

3. Hit-Savvies' adoption rate (and consequently Hitters adoption rate) is quite sparse and fragmented[7] and by looking at their Hits adoption distribution in Figure 2.5 and active period distribution in Figure 2.6 (which are extremely left-screwed and converging to one) we can detect how they may be poor success indicators. On the counterpart, Flop-adopters (and Floppers as a consequence) adopt on average more Flops (adopted

[5]What users had searched with the search engine isn't related with the fame of my Last.fm artists.

[6]Maybe decreasing AT artists are singers or bands whose releases are sporadic and drive the listeners only for a short period of time, like for example the summer hits or the seasonal songs. Since I didn't retrieved neither artists' songs or albums, judging only from their classification in a main music genre, the majority of them are related to electronic (2,988 out of 13,616), pop (1,365 out of 13,616) and hip-hop/rap (1,218 out of 13,616), which are genres whose seasonality is quite a discriminator for success.

[7]More on this topic is discussed in the next section, where I've clustered all the users of my data set using a PLDM (Personal Listening Data Model.), to identify their personal preferences and patterns of music consumption and from which resulted that the majority of Hit-Savvies make an average of one single adoption in the period that I've analyzed

mainly by them, otherwise they would be classified as Hits) and remain active for consecutive weeks in a row;

4. the majority of the items of my training sets reach a global maximum in the first weeks of my training. Having an item's threshold within the first month of the training set may be a heavy discriminant, since an adopter must have listened to an item within this temporal range. Otherwise isn't classified as an innovator and on the contrary, if he continues to adopt *late Hits*, he will result in a negative HF-score and will be labeled as Flop-adopter and later used as a Flopper[8]. This may lead to the unpleasant drawback of cutting out potential Hit-Savvies, shrinking the potential Hitters pool and compromising the classification of unseen items.

Points 2), 3) and 4) together with a low TPR score due to the higher number of FN in comparison to TP, are shared among all the other success definitions, too. I've tried to investigate the reasons behind the high FN rate and the conclusions which aroused are the same. The low number of Hit-Savvies (and as a consequence Hitters) combined with their occasional adoption rate make them poorly predict. This explains the lower number of TP in respect of the number of FN[9]. The high number of FN is given by the mixture of Hitters smaller number, Hitters' sporadic adoption rate, Floppers higher presence, Floppers more concise adoption rate and by the fact that Floppers are not pure (adopt only Flop items) as depicted from Figure 2.7. As a consequence, they still adopt Hits and *late Hits* and given an unseen item, if few Hitters (or all the Hitters, but less than the number of Floppers and I've underlined how this is easily achieved) adopt it, the item will be classified with high probability as a Flop, even if it is, in reality, a Hit (globally adopted by the majority of my users). Lastly, I've tried to find a better model tuning to achieve more accurate and homogeneous predictive scores, by running and combining the following ideas:

- postponing goods' threshold after the first month, to try to broaden the Hit-Savvies pool and as a consequence the Hitters selected[10];
- using all the Hit-Savvies as Hitters[11];
- cutting out all the impure Flop-adopters (adopters who adopted only *late Hits* or a number of *late Hits* higher than the number of Flops), experimenting even in using only extreme pure Flop-adopters (adopters who adopted only *Flops*), to use only the most relevant Flop indicators;

[8]I've rerun the predictive model modifying the Flopper selection method, by imposing that only proper Flop-adopters (adopters who adopted at least one flop) had to be selected. This, however, didn't change the scores that I got, since even the previous Flopper selection method took in account this: the Flop-adopter are sorted by highest Flop Coverage and lowest HF-propensity: for all the success definition only about the 33% of the total number of Flop-adopters were chosen as Floppers, never reaching the *late Hits* adopters (in the sorting criterion, they had been placed in the last positions since they covered a small number of Flops (or even zero)).

[9]I obtain a signifying TPR increase above 0.5 only with the success definition Def1, Def2 and Def7 and the splits b, c and d, while maintaining an adequately good PPV.

[10]This solution actually massed up the prediction model, due to the fact that I intentionally modified early adoption's patterns.

[11]Among all the success definitions, Hitters were choose all the time with a percentage near 100% among Hit-Savvies, so my attempt hadn't brought variations in Hitters selection.

- shifting the time granularity from a month based to a weekly and narrowing the training and test set time slots[12].

All my experiment didn't produce significant changes in the scores, since the initial selected Hit-Savvy set remained unchanged and I could play only in broadening it, narrowing the Flop-adopters set and making it more concise and truthful. The occasional adoption rate and the limited number of adoptions made by each Hit-Savvy persisted and made the prediction hard to stun. Only the experiments with Flop-adopter's purity gave me a significant FN lowering rate, but on the counterpart led to a greater number of unclassified items (increased cases in which $\pi(g_*) < \widehat{m}$ and $\delta(g_*) < \widehat{n}$). As a proof, I've showed the scores obtained using pure Flop-adopters here [5] (extreme pure Flop-adopters' scores are similar, too).

	a (10)	**b (10)**	**c (7)**	**d (4)**	**e (1)**
PPV	0.43	0.53	0.60	0.62	0.53
NPV	0.99	0.99	0.99	0.99	0.96
TPR	0.17	0.77	0.83	0.64	0.35
TNR	0.99	0.99	0.99	0.99	0.98

Table 2.7: Hits&Flops predictive performances when dealing with different temporal splits for success definition 1.

	a (10)	**b (12)**	**c (7)**	**d (6)**	**e (1)**
PPV	0.78	0.70	0.71	0.71	0.59
NPV	0.99	0.99	0.99	0.99	0.96
TPR	0.23	0.70	0.73	0.87	0.36
TNR	0.99	0.99	0.99	0.99	0.98

Table 2.8: Hits&Flops predictive performances when dealing with different temporal splits for success definition 2

	a (10)	**b (9)**	**c (6)**	**d (4)**	**e (1)**
PPV	0.52	0.60	0.67	0.64	0.56
NPV	0.99	0.99	0.99	0.99	0.97
TPR	0.21	0.60	0.69	0.67	0.37
TNR	0.99	0.99	0.99	0.99	0.99

Table 2.9: Hits&Flops predictive performances when dealing with different temporal splits for success definition 7.

As a consequence I'll stick with the scores first introduced, focusing most on the success definitions that showed the best results: Def1, Def2 and Def7. Tables 2.7, 2.8 and 2.9 summarize and compare their predictive performances when dealing with different temporal

[12]The problems with Hit-Savvy restricted set, their sparse adoption patterns and the counterpart Flop-adopter's behaviour still remained unchanged.

splits. Each column identifies a different split of observed 18 months, while within bracket it's indeed encoded the best observation widow performed on the test set. As we can see among the three definitions the one that returns better PPV and TPR scores is Def2, which takes into account the number of play-counts made by my seed users. This underlines how for my data it's not the number of listeners or actions that make an item successful or not, but the total number of play-counts earned. Since my Hitters are occasionally active this success definition suits them. Taking a few actions but with a great counter of play-counts can make them overcome their minority and less frequent adoption rate in comparison to Floppers and so transform them in fruitful success indicators, lowering as well the number of FP encountered.

8 Hit-Savvies and Leaders comparison

Lastly, I've compared the Hit-Savvies found by my different success definitions with the Leaders first introduced in this paper, to understand not only the predictive power of these important users, but also how their influence spread among the network. I've undergone this analysis for Def3, Def4, Def5 and Def6, too. Even if they didn't perform well with the predictive model, it's still useful to understand their Hit-Savvy's set pattern of diffusion. The comparison between Hit-Savvies and Leaders was made using K-Means with k=4,[13] taking into account that a Leader spreads more than one item over the network. For this reason, for each one of them, I've performed a mean among its $T_{l,\psi}$'s Tribe, Width, Depth, Strength and spread music genres, to achieve an unequivocal characterization. Among the 91,652 uni-vocal leaders their distribution in the 4 clusters is the following:

- Cluster A's members = 83,604
- Cluster B's members = 1,723
- Cluster C's members = 425
- Cluster D's members = 5,900

[13] Best k according to Sum of Squared Error[41] in range k ∈ [2, 21].

	#Hit-Savvies	#H-S_leaders	#H-S_influenced	#H-S_neutral
Def1	2,338	489 (21%)	413 (183 H-S_leaders)	1,436 (61,42%)
Def2	2,307	495 (21%)	421 (180 H-S_leaders)	1,391 (60,29%)
Def3	1,072	98 (8%)	75 (15 H-S_leader)	905 (84,42%)
Def4	1,864	298 (16%)	249 (88 H-S_leaders)	1,317 (70,65%)
Def5	1,605	328 (17%)	253 (93 H-S_ leaders)	1,352 (84,24%)
Def6	822	86 (10%)	65 (12 H-S_ leaders)	671 (81,63%)
Def7	2,346	493 (21%)	416 (183 H-S_leaders)	1,437 (61,25%)

Table 2.10: Number of Hit-Savvies which are at the same time Leaders, number of Hit-Savvies influenced by Leaders (Leader Hit-Savvies or canonical Leaders) and number of Hit-Savvies which are neither influenced or influencers.

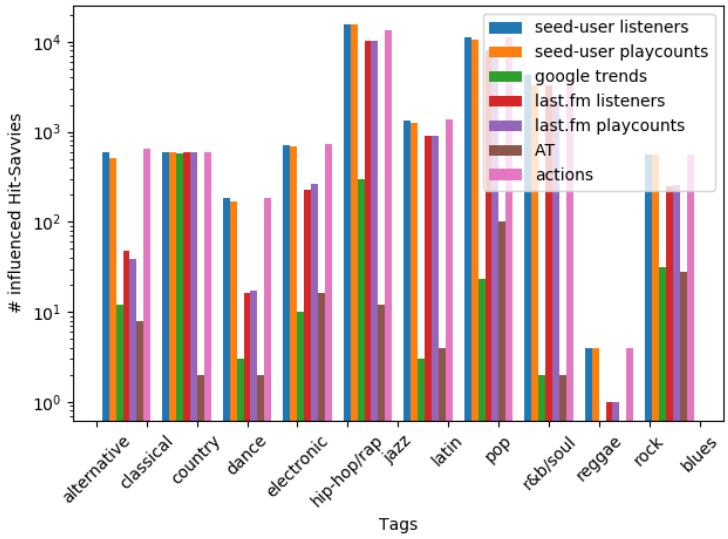

Figure 2.10: Music genres with which influenced Hit-Savvies are compelled.

Taking into account only the Hit-Savvies, from Table 2.10 emerges an interesting scenario. Hit-Savvies aren't in their social circle the absolute first to adopt new, successfully items or if they are, they however don't influence their tribe within the temporal parameter δ. As a result, Hit-Savvies are from my data set point of view poor influential spreaders. From Table 2.11 and Figure 2.10, we can see, instead, how some Leader Hit-Savvies receive the influence of other Leaders, too. The main music genres with which these Leader Hit-Savvies are influenced are hip-hop/rap, pop, r&b/soul and rock, which in accordance with Table 1.3, underlines the diffusive power of r&b/soul and hip-hop/rap and the tendency to listen to the most well-known music genres(pop and rock), as well as the emerging hype of new ones (electronic and hip-hop/rap) depicted in Chapter 1, Section 4.

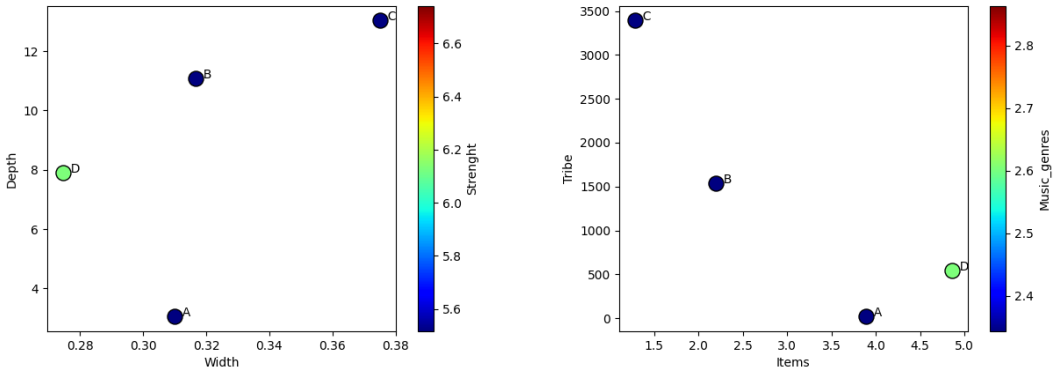

Figure 2.11: Leader clusters' centroids.

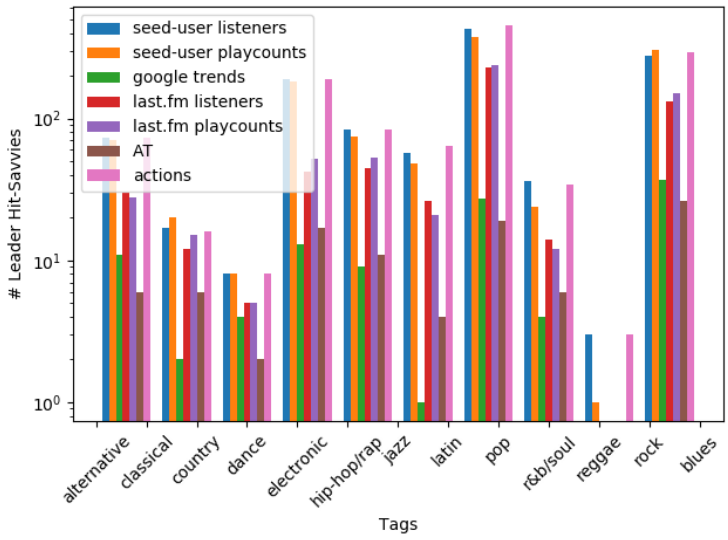

Figure 2.12: Music genres which Leader Hit-Savvies spread.

	Cluster A	Cluster B	Cluster C	Cluster D
Def1	0.973	0.020	0.004	0.002
Def2	0.976	0.028	0.004	0.002
Def3	1.0	-	-	-
Def4	0.966	0.027	0.003	0.003
Def5	0.976	0.018	0.003	0.003
Def6	1.0	-	-	-
Def7	0.975	0.028	0.004	0.002

Table 2.11: Hit-Savvies presence in the leader clusters.

Moreover, taking into account Table 2.11 and Figures 2.11 and 2.12, since almost all Leader Hit-Savvies are located in cluster A, they:

- influence near a third of their direct followers;
- reach a depth near 3 and a mean depth near 2, implying that in some case the third part of their direct followers spread the received influence at least to the next $T_{l,\psi}$'s level. On average $T_{l,\psi}$'s tribe is composed by 20 nodes, distributed among leader's direct followers and followers' followers;
- the strength with which their influence is spread among $T_{l,\psi}$ is low, implying that the influenced users don't adopt the item with which they are compelled. Presumably they make a single play-count on ψ, driven by curiosity and by the possibility to discover a new, unknown artist but in the end they don't adopt the novelty;
- spread a mean of 4 artists, within a mean of 2 genres. Rock, pop and electronic result the most diffused ones, similar to the leaders' global distributed discovered in Figure 1.12.

Chapter 3

PLDM Model

In this Chapter I'll try to better characterize user's personal preferences and music consumption, building a listening profile as the one described in [21] and [22]. By applying the following definitions and functions, it is possible to build for each user a listening model, which from latter on will be called *Personal Listening Data Model (PLDM).*

Definition 16 (Listening):
Given a user u we define $L_u = \{$*(artist, genre, slot, quantity)*$\}$ *as the set of listening performed by u.*

From the set of listening L_u I can extract the set of artists A_u, music genres G_u, weeks L_u and play-counts Q_u for each user. More formally:

- $A_u = \{(..., \text{artist}, ...) \in L_u\}$
- $G_u = \{(..., \text{genre}, ...) \in L_u\}$
- $S_u = \{(..., \text{slot}, ...) \in L_u\}$
- $Q_u = \{(..., \text{quantity}, ...) \in L_u\}$

Besides the sizes of these sets, a valuable summary of the user behavior can be realized through frequencies dictionary indicating the support (i.e. the relative number of occurrences) of each feature of the listening.

Definition 17 (Support):
The support function returns the frequency dictionary of (item, support) where the support of an item is obtained as the number of occurring items on the number of listening:

$$supp(X, L) = \{(x, |Y|/|L|) | x \in X, Y \subset L, \forall l \in Y, x \in l\}$$

We define the following frequency dictionaries:

- $a_u = \text{supp}(A_u, L_u)$
- $g_u = \text{supp}(G_u, L_u)$
- $s_u = \text{supp}(S_u, L_u)$

- $q_u = \text{supp}(Q_u, L_u)$

These dictionaries can be exploited to extract indicators.

Definition 18 (Entropy):
The entropy function returns the normalized entropy in [0, 1] of a dictionary X. It is defined as:

$$entropy(X) = -\sum_{i=1}^{n} P(x_i) log_b \frac{P(x_i)}{log_b n}$$

The entropy tends to 0 when the user behavior with respect to the observed variable in a systematic way, tends to 1 when the behavior is not predictable. I define the entropy for artists, genres, weeks and quantities as:

- $e_{a_u} = \text{entropy}(a_u)$
- $e_{g_u} = \text{entropy}(g_u)$
- $e_{s_u} = \text{entropy}(s_u)$
- $e_{q_u} = \text{entropy}(q_u)$

Definition 19 (Top):
The top function returns the most supported item in a dictionary. It is defined as:

$$top(X) = argmax(x, y) \in X(y)$$

I define the top for artists, genres, slots[1] and quantities[2] as:

- $\hat{a_u} = \text{top}(a_u)$
- $\hat{g_u} = \text{top}(g_u)$
- $\hat{s_u} = \text{top}(s_u)$
- $\hat{q_u} = \text{top}(q_u)$

Moreover, I want to consider for each user the set of representatives, i.e. significantly most listened, subsets of artists, genres, slots and quantities.

Definition 20 (Representative):
The repr function returns the most representative supported items in a dictionary. It is defined as:

$$repr(X) = knee_{(x,y)\in X}(y)$$

[1]Corresponds to an user's most active week (the week in which he has taken the greatest number of actions.)

[2]Corresponds to an user's most frequent play-count's counter.

This result is achieved by employing a technique known as "knee method" represented by the function knee($\cdot$). It sorts the vector according to the supports, and it returns as most representative the couples with support greater or equal than the support corresponding to the knee in the curve of the ordered frequencies. I define the most representative for artists, genres, slots and quantities as:

- $\tilde{a_u} = \text{repr}(a_u)$
- $\tilde{g_u} = \text{repr}(g_u)$
- $\tilde{s_u} = \text{repr}(s_u)$
- $\tilde{q_u} = \text{repr}(q_u)$

Finally, to understand how each user u is related in terms of preferences with his followers and followings, I define the following sets:

$$followers = \{v_1, ..., v_n\}\ s.t\ \forall\ v_i\ \exists\ (v_i, u)\ \in\ E$$

$$followings = \{v_1, ..., v_m\}\ s.t\ \forall\ v_i\ \exists\ (u, v_i)\ \in\ E$$

By applying the definitions and the functions described above on the user listening set L_u I've retrieved a Personal Listening Data Model (PLDM). The PLDM characterizes the listening behavior of a user by means of its indicators, frequencies and patterns:

Definition 21 (PLDM):
Given the listening L_u we define the personal listening data model as:

$$\begin{aligned} P_u = (&|L_u|, |A_u|, |G_u|, |S_u|, |Q_u|,\\ &e_{a_u}, e_{g_u}, e_{s_u}, e_{q_u},\\ &a_u, g_u, s_u, q_u,\\ &\hat{a_u}, \hat{g_u}, \hat{s_u}, \hat{q_u},\\ &\tilde{a_u}, \tilde{g_u}, \tilde{s_u}, \tilde{q_u},\\ &followers, followings) \end{aligned}$$

where the first two rows are indicators, the third frequencies, the fourth and the fifth patterns and the last one friends.

In particular I was interested in understanding if, and how, friendship ties affect the listening behavior and users' homophily. By exploiting the PLDM among two users I was able to calculate the similarity between them, by using 2 additional indicators μ and σ, computes respectively as the mean and the standard deviation of the cosine similarity calculated on *followers* and *followings* sets of a given user u. The higher is μ, the more homophilous is u with her friends w.r.t. a certain variable X. The higher is σ, the more various is the similarity between u and her friends w.r.t. a certain variable X. I then applied K-Means with k=3 resulting as the best split, to cluster my seed users and discoverer intrinsic common traits.

Figures 3.1, 3.2 and 3.3 display the normalized radar charts representing the three clusters' centroids. Table 3.1 indeed shows the percentage of Hit-Savvies in each cluster taking

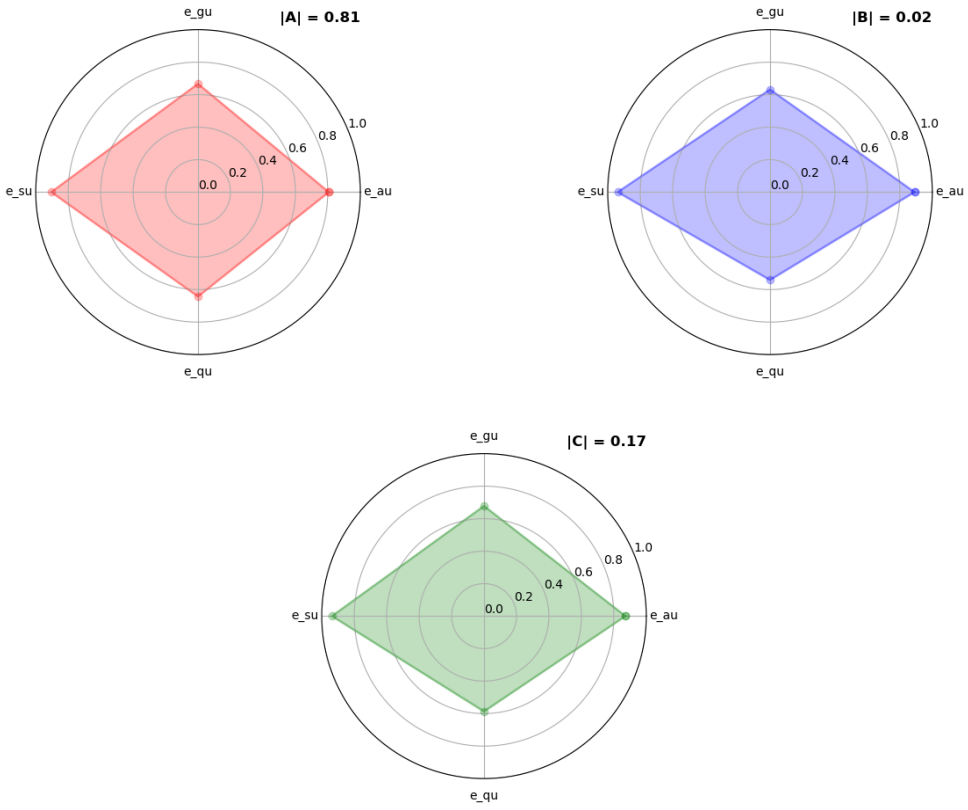

Figure 3.1: Radar charts for the centroids of the clusters extracted on the indicators of entropy based on PMDLs.

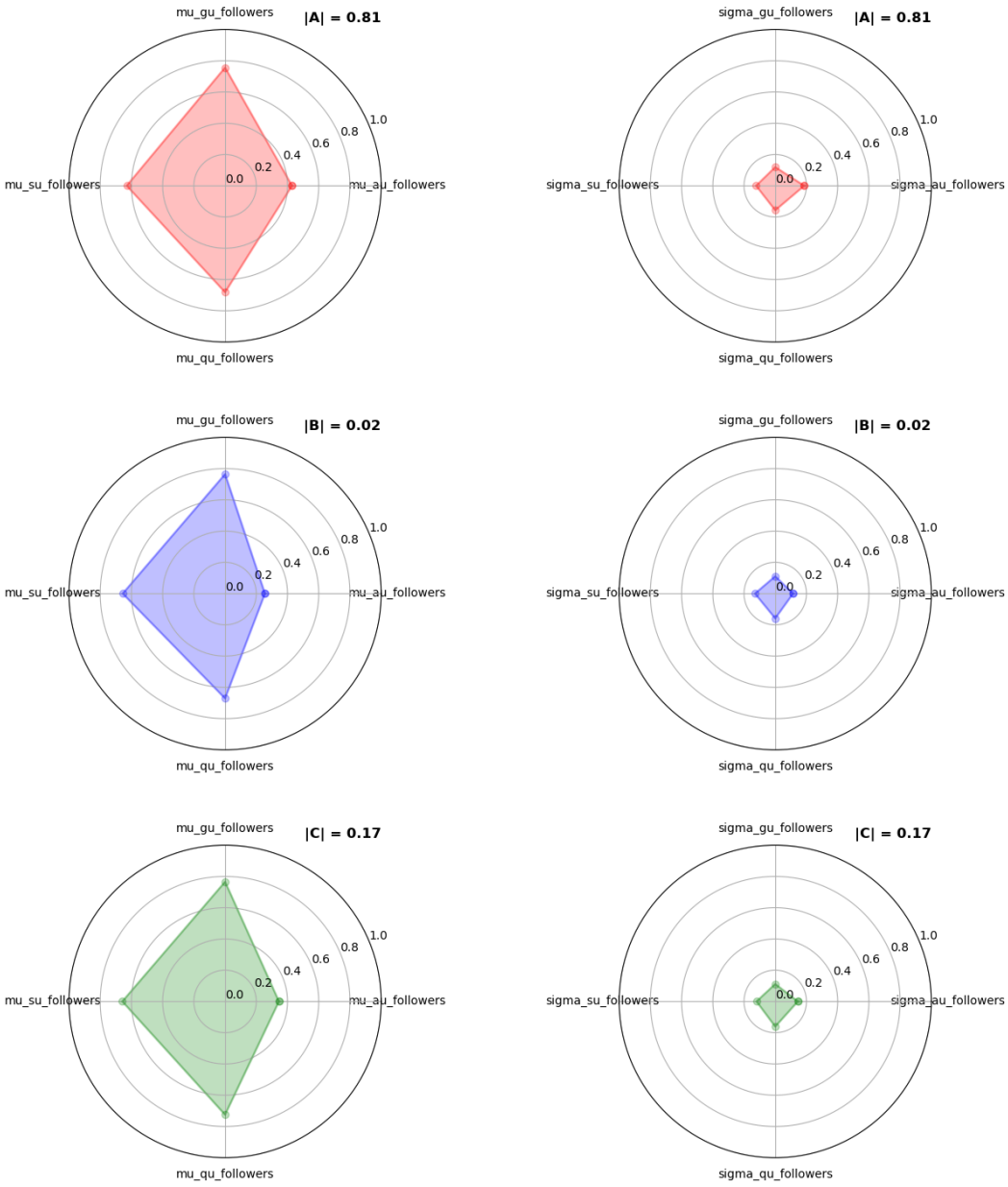

Figure 3.2: Radar charts for the centroids of the clusters extracted on the indicators of follower homophily based on PMDLs.

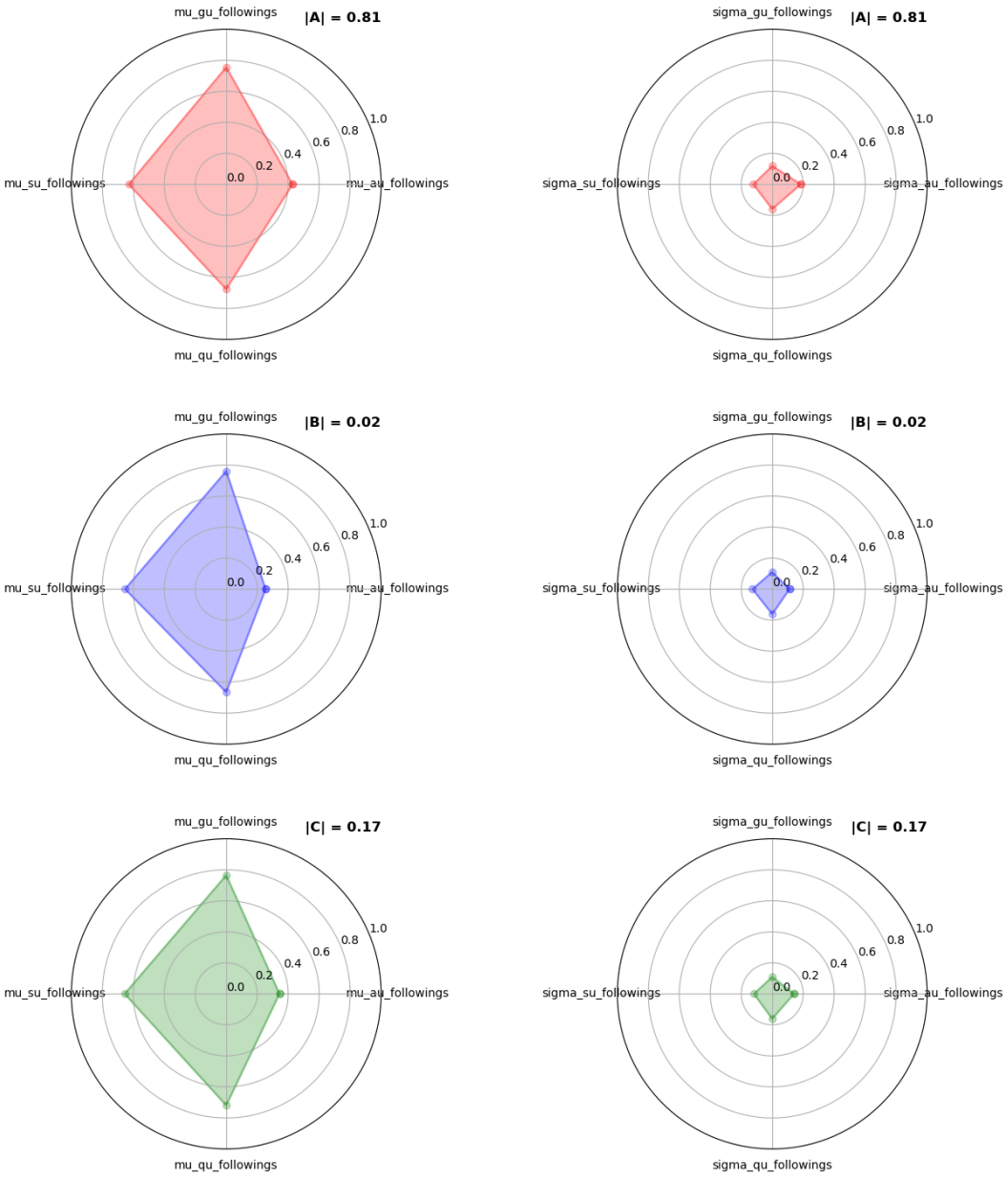

Figure 3.3: Radar charts for the centroids of the clusters extracted on the indicators of following homophily based on PMDLs.

	Cluster A	Cluster B	Cluster C
Def1	0.976	0.022	0.002
Def2	0.974	0.023	0.001
Def3	0.998	0.001	-
Def4	0.985	0.014	0.001
Def5	0.983	0.015	0.001
Def6	0.997	0.002	-
Def7	0.975	0.022	0.002

Table 3.1: Hit-Savvies presence in the LPDM clusters.

	Cluster A	Cluster B	Cluster C
# adoptions	28.86	552,82	163,36
# artists	12,47	116,64	50,84
# genres	4,28	8,82	7,57
# slots	16,26	60,37	45,83
# quantities	6,13	20,43	14,22

Table 3.2: Additional information's for the centroids of the clusters extracted on the indicators of following homophily based on PMDLs.

into account the different success definitions. We can notice that for all the success definitions, Hit-Savvies are clustered in cluster A, meaning that their are a lot alike. From Figure 3.1 we discover that actually all three clusters are quite similar and behave unpredictably. Cluster A is the most unpredictable in regard of the play-count patterns: one play-count and many more coexist in the adoption log of the users clustered here. By looking also at Table 3.2, where I recall a low number of total adoption rate, adopted artists and weeks, this makes me drove to the idea that the majority of these users don't use Last.fm service on a regularl basis. This maybe due to the fact the they shifted their actions over other platforms (Spotify, Apple music, SoundCloud, Deezer only to mention some) or to a sporadic usage of Last.fm. Another cause could be that these users just tried out the service, didn't found it suitable for their needs, but occasionally return to it. Whatever may be the reason behind this behaviour, it invests almost all my Hit-Savvy set, negatively impacting my predictive model performances. On the other hand, Cluster B and C are the most unpredictable in regard to the number of artists (Cluster B), genre (Cluster) and week (Cluster B and C). Contrarily from Cluster A, they are more diversified in the number of adoptions, artists and most importantly adoption's weeks, but since my Hit-Savvies all converge towards Cluster A, they can't help in the predictive model.

Taking into account the similarity between a users and its followers, from Figure 3.3 at once arises the artist dissimilarity: users that follow someone tend to adopt a number of artists as well as a variety of them quite different from the given adopter (low $\mu_{a_u}\_followers$ and $\sigma_{a_u}\_followers$). On the other hand, they tend to follow someone due to their common musical tastes and listening patterns, establishing a low variety among these traits (high $\mu_{g_u}\_followers$, $\mu_{s_u}\_followers$, $\mu_{q_u}\_followers$ and low $\sigma_{g_u}\_followers$, $\sigma_{s_u}\_followers$, $\sigma_{q_u}\_followers$). Taking into account the similarity between a users and its followings we fall

again in the previous findings. Users tend to follow other users because they share similar musical tastes and listening patterns (number of adopted weeks (and so listening periods) and number of play-counts (and so listening volume)). This inconsistency among a user's artists and his followers/followings' artists, may be at the core of the low predictive and diffusive process, too, since the success definitions are computed using artists as guidelines. More suitable success definitions, for instance, may take into account musical genres and listening patterns to define the success of the items present in the adoption log, and more generally, to incorporate topic modeling in the influence analysis.

Chapter 4

Communities discovery

Community[1] detection is often used to understand the structure of large and complex networks, but the absence of a unique, well-posed, definition of what a community in a complex network should represent is only one of the issues to face when approaching network clustering. In this section I'll use four Community Detection algorithms, to analyze if the PLDM clusters found in the previous section reflects the network structure, too[2]. To achieve this goal, I labeled each node with the identifier of the cluster it belongs to and then performed the CD algorithms, classified as:

- crisp communities (each node belongs to one and only one community):
  - Louvain;
  - Infomap;
  - Label Propagation;
- overlapping communities (any node can be assigned to more than one community):
  - Angel [23].

If the Louvain algorithm focuses on the modularity maximisation problem, Infomap's concern is to find the structures within a network that are significant with respect to how information or resources flow through that network. Label Propagation detects communities using network structure alone as its guide, and does not require a pre-defined objective function or prior information about the communities. As a drawback, many nodes could be clustered in the same community, leading to few, huge clusters which may not fit the real nodes' cohesivity. Finally, Angel is a node-centric bottom-up community discovery algorithm, which leverages ego-network structures and overlapping label propagation to identify micro-scale communities that are subsequently merged in mesoscale ones.

[1] Cluster of nodes which form relatively dense groups: we can say that community discovery can be seen as a network variant of traditional data clustering.

[2] More informally, are community formed by users belonging to the same PLDM cluster?

9 Louvain

The algorithm optimises modularity in two elementary phases:

1. local moving of nodes;
2. aggregation of the network.

First, each node in the network is assigned to its own community. Then for each node i, the change in modularity is calculated for removing i from its own community and moving it into the community of each neighbor j of i. This value is easily calculated by two steps:

1.1 removing i from its original community;

1.2 inserting i to the community of j.

The equation for step *1.2* is:

$$\Delta Q = \left[\frac{\Sigma_{in} + 2k_{i,in}}{2m} - \left(\frac{\Sigma_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right]$$

where Σ_{in} is sum of all the links inside the community i is moving into, Σ_{tot} is the sum of all the links to nodes in the community i is moving into, k_i is the degree of i, $k_{i,in}$ is the sum of the links between i and other nodes in the community that i is moving into, and m is the sum of all links in the network. Then, once this value is calculated for all communities i is connected to, i is placed into the community that resulted in the greatest modularity increase. If no increase is possible, i remains in its original community. This process is applied sequentially to all nodes until no modularity increase can occur. In the second phase of the algorithm, Louvain algorithm groups together all the nodes of the individuated communities, building a new network where nodes are the communities from the previous phase. Any links between nodes of the same community are now represented by self-loops on the new community node and links from multiple nodes in the same community to a node in a different community are represented by weighted edges between communities. Once the new network is created, the second phase has ended and the first phase can be re-applied to the new network.

10 Infomap

Infomap is a flow-based method, which realises on the *Map Equation*. The *Map Equation* is built on a flow-based and information-theoretic foundation, taking advantage of the duality between finding community structure in networks and minimizing the description length of a random walker's movements on the network. Indeed, if we want to understand network structures, we need to understand the flow of information on it. We then identify the modules that compose the network by finding an efficiently description of how information flows on the network. A group of nodes among which information flows quickly and easily can be aggregated and described as a single module; the links between modules, on the other hand, capture the avenues of information flows between those modules. Describing information flow is a **coding/compressing problem**. The key idea in coding theory is that a *data*

stream can be compressed using a code that exploits regularities in the stream. With the help of a random walk used as a proxy for the information flow, we can do the same thing on a network. First of all we need the *data stream*, which in our case translates in describing the trajectory of a random walk on the network. A basic approach is to give unique names to every node in the network. The Hoffman code[47] is an efficient way to do so, because Huffman codes save space by assigning short codewords to common events or objects and long codewords to rare one. In addition, we use also the Shannon's source coding theorem, which implies that when you use n codewords to describe the n states of a random variable X that occur with frequencies p_i, the average length of a codeword can be no less than the entropy of the random variable X itself. This theorem provides us with the necessary apparatus to see that, in our Huffman illustration, the average number of bits needed to describe a single step in the random walk is bounded below by the entropy H(P), where P $= \{p_i\}$, $\sum_i p_1 = 1$, is the random walker's distribution of visit frequencies to the nodes on the network. So defining this lower bound on code length to be L, we have that:

$$L(P) = H(P) = -\sum_{1}^{n} p_i log_2 p_i$$

with the logarithm taken in base 2 to measure the codelength in bits. Matching the length of codewords to the frequencies of their use gives us efficient codewords for the nodes, but no map. To make a map, we need to separate the important structures from the insignificant details. We therefore retain unique names for large-scale objects, the clusters or modules to be identified within our network, but we reuse the names associated with fine-grain details, the individual nodes within each module. Thus, we assign unique names to coarse-grain structures but reuse the names associated with fine-grain details. A two-level description allows us to describe the path in fewer bits than we could do with a one-level description. We capitalize on the network's structure and, in particular, on the fact that a random walker is statistically likely to spend long periods of time within certain clusters of nodes. A special codeword, the exit code, is chosen as part of the within-cluster Huffman coding and indicates that the walk is leaving the current cluster. The exit code is always followed by the "name" or module code of the new module into which the random walker is moving.

Herein lies the duality between finding community structure in networks and the coding problem: to find an efficient code, we look for a module partition $\mathbf{M}$ of n nodes into m modules so as to minimize the expected description length of a random walk. That is, for module partition $\mathbf{M}$ of n nodes into m modules, the lower bound of the average length of the code describing a step of the random is given by the following *Map Equation*:

$$L(\mathbf{M}) = q_{\curvearrowright} H(Q) + \sum_{i}^{m} p^i_{\circlearrowright} H(P^i)$$

This equation comprises two terms:

1. first is the entropy of the movement between modules;
2. second is the entropy of movements within modules, including the exit code for module i.

Each is weighted by the frequency with which it occurs in the particular partitioning. Here:

- $q_{i,\curvearrowright}$ probability to exit module i;
- $q_{\curvearrowright} = \sum_{i=1}^{m} q_{i,\curvearrowright}$ is the probability that the random walker enters any of m modules on any given step;
- $H(Q) = -\sum_{i=1}^{m}(q_{i,\curvearrowright}/q_{\curvearrowright})log_2(q_{i,\curvearrowright}/q_{\curvearrowright})$ is the entropy of the module names, i.e., the entropy of the underlined codewords;
- p_α probability to visit node α;
- $H(P^i) = -(q_{i,\curvearrowright}/p^i_{\circlearrowright})log_2(q_{i,\curvearrowright}/p^i_{\circlearrowright}) - \sum_{\alpha \in i}(p_\alpha/p^i_{\circlearrowright})log_2(p_\alpha/p^i_{\circlearrowright})$ is the entropy of the within-module movements, including the exit code for module i;
- $p^i_{\circlearrowright} = \sum_{\alpha \in i} p_\alpha + q_{i\curvearrowright}$ is the fraction of within-module movements that occur in module i, plus the probability of exiting module i.

That is, for a given modular partition of the network, there is an associated information cost for describing the movements of the random walker, or of empirical flow if available. The partition with the shortest description length is the one that best captures the community structure of the network with respect to the dynamics on the network. The underlying code structure of the *Map Equation* is designed such that the description can be compressed if the network has regions in which the random walker tends to stay for a long time. Therefore, with a random walker as a proxy for real flow, minimizing the *Map Equation* over all possible network partitions reveals important aspects of network structure with respect to the dynamics on the network.

The core of the Infomap algorithm follows the Louvain method closely: neighboring nodes are joined into modules, which subsequently are joined into supermodules and so on. First, each node is assigned to its own module. Then, in random sequential order, each node is moved to the neighboring module that results in the largest decrease of the map equation. If no movement results in a decrease of the map equation, the node stays in its original module. This procedure is repeated, each time in a new random sequential order, until no move generates a decrease of the map equation. Then the network is rebuilt, with the modules of the last level forming the nodes at this level, and, exactly as at the previous level, the nodes are joined into modules. This hierarchical rebuilding of the network is repeated until the map equation cannot be further reduced.

11 Label Propagation

The Label Propagation algorithm (LP) works by propagating labels throughout the network and forming communities based on this process of label propagation. The intuition behind the algorithm is that a single label can quickly become dominant in a densely connected group of nodes, but will have trouble crossing a sparsely connected region. Labels will get trapped inside a densely connected group of nodes, and those nodes that end up with the same label when the algorithms finishes can be considered part of the same community. The algorithm works as follows:

- every node is initialized with a unique community label;
- labels are propagate through the network;
- at every iteration of propagation, each node updates its label to the one that the maximum numbers of its neighbours belongs to;
- LP reaches convergence when each node has the majority label of its neighbours.

At the end of the propagation only, a few labels will remain - most will have disappeared. Nodes that have the same community label at convergence are said to belong to the same community.

12 Angel

Angel is the faster successor of Demon algorithm (Democratic Estimate of the Modular Organization of a Network) [24]. Demon, and as a consequence, Angel, aims to extract the ego network of each node and apply a Label Propagation Algorithm on this structure, ignoring the presence of the ego itself that will be judged by its peers neighbors.Otherwise this would lead to noise in the subsequent steps of Demon, since by definition of the local community, the nodes would be put in the same community if they are close to each other. Obviously, a single node connecting the entire sub-graph will make all nodes very close, even if they are not in the same community, and for this reason the ego node is removed from its own ego network. For every *EgoMinusEgo*[3] subgraph such obtained, then is computed the communities within, using as guidance the Label Propagation Algorithm. This choice is made for the following reasons:

- LP shares with this work the definition of what is a community;
- LP is known as the least complex algorithm in the literature, reaching a quasi-linear time complexity in terms of nodes[4];
- LP will return results of a quality comparable to more complex algorithms [25] .

Suppose that a node $v \in$ V has neighbors $v_1, v_2, ..., v_k$ and that each neighbor carries a label denoting the community that it belongs to. Then v determines its community based on the labels of its neighbors, assuming that each node in the network chooses to join the community to which the maximum number of its neighbors belong. Clearly, a node with an equal maximum number of neighbors in two or more communities can belong to both communities, thus identifying possible overlapping communities. So the main difference between Demon and Angel versus LP are:

1. the democratic bottom-up mining approach: in turn, each node gives the perspective of the communities surrounding it and then all the different perspectives are merged together in an overlapping structure;

[3]Node v's ego-network, with $v \in$ V, from which the vertex v and all edges attached to v are removed.
[4]The other community discovery algorithms proposed reach all at least a quadratic time complexity.

2. the possibility to have overlapping communities.

The result of the LP is a set of local communities, according to the perspective of node v (which at the end of the Label Propagation algorithm is reintroduced, in each local communities). These communities are likely to be incomplete. Thus, the next step is to merge each local community of C to obtain the result set. The Merge operation is defined as follows: two communities C and I are merged if and only if at most the ϵ % of the smaller one is not included in the bigger one; in this case, C and I are removed from $C = Max(C(v),\ v \in V)$, with $C(v)$ denoting a v's community, and their union is added to the result set. The ϵ factor is introduced to vary the percentage of common elements provided from each couple of communities:

- $\epsilon = 0$ ensure that two communities are merged only if one of them is a proper subset of the other;
- $\epsilon = 1$ even communities that do not share a single node are merged together.

The democratic algorithm is incremental, allowing to recompute the communities only for the new incoming nodes and edges in an evolving network.

13 Communities evaluation

The results that I've obtained with the above's community detection algorithms are summarized in Table 4.1 and depicted in Figure 4.1:

	Louvain	Infomap	Label Propagation	Angel
num. communities	2,639	12,622	9,146	818
largest community	32,071	13,323	143,162	73,632
overlap	False	False	False	True
coverage	1.0	1.0	1.0	4.41
mean size	70.71	14.78	20.40	94.95
mean density	0.86	0.43	0.82	0.75
mean degree	8.32	6.60	10.37	21.98
mean clustering	0.12	0.16	0.10	0.22
mean conductance[52]	0.04	0.43	0.28	0.44
mean modularity[51]	1.99	2.03	1.66	5.69
mean edges inside C	370.80	61.08	134.90	365.78
mean edges pointing to C	232.88	74.16	25.32	216.89
mean edges pointing outside C	155.35	50.52	16.89	160.11

Table 4.1: Communities' general info

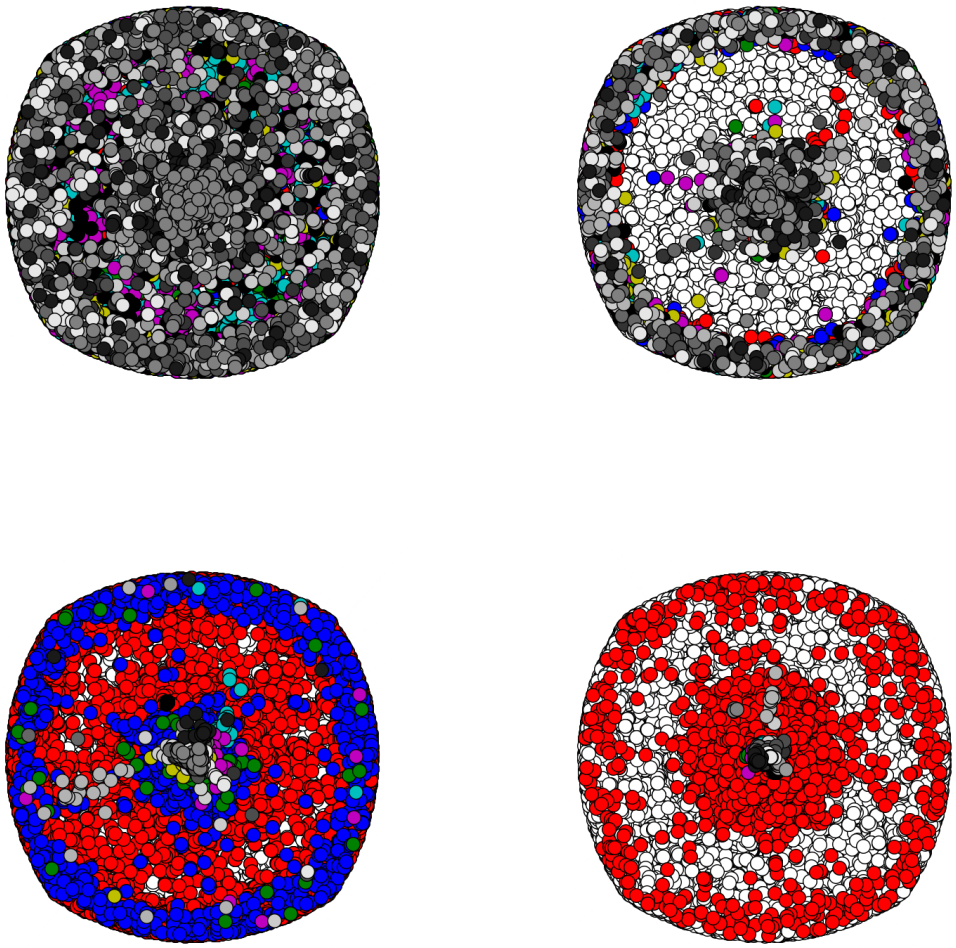

Figure 4.1: Louvain (top left), Infomap (top right), Label Propagation (bottom left) and Angel (bottom right) communities.

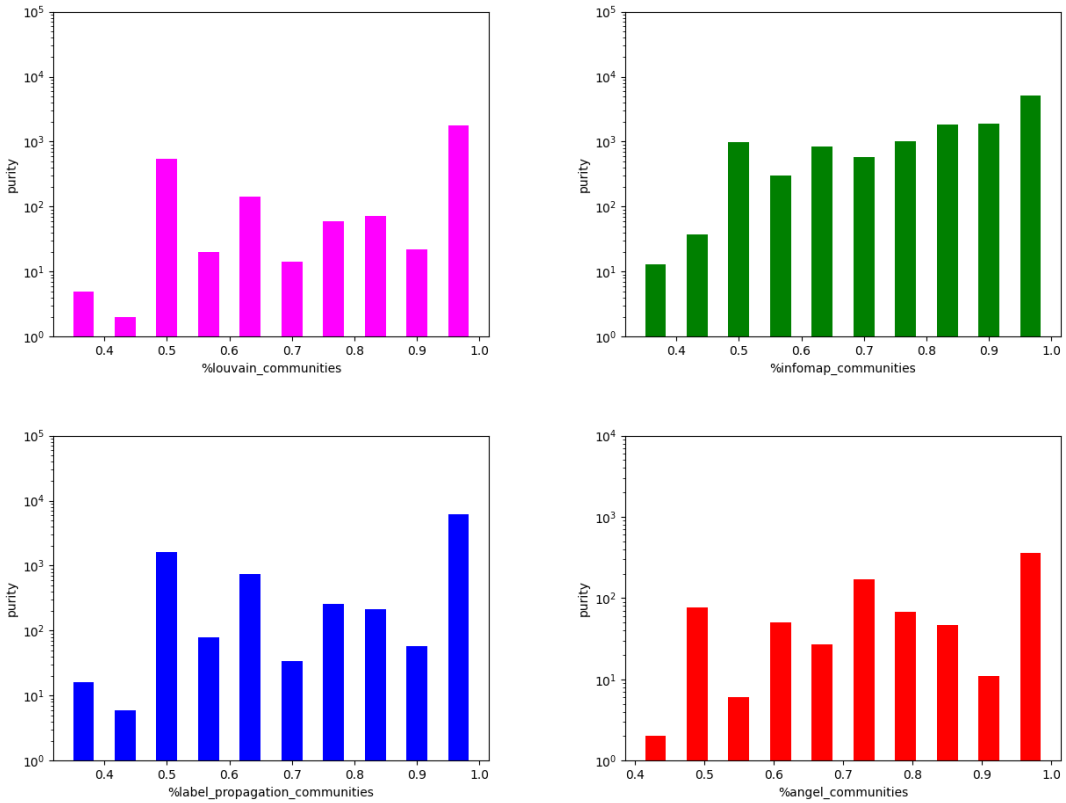

Figure 4.2: Louvain (top left), Infomap (top right), Label Propagation (bottom left) and Angel (bottom right) purity distribution.

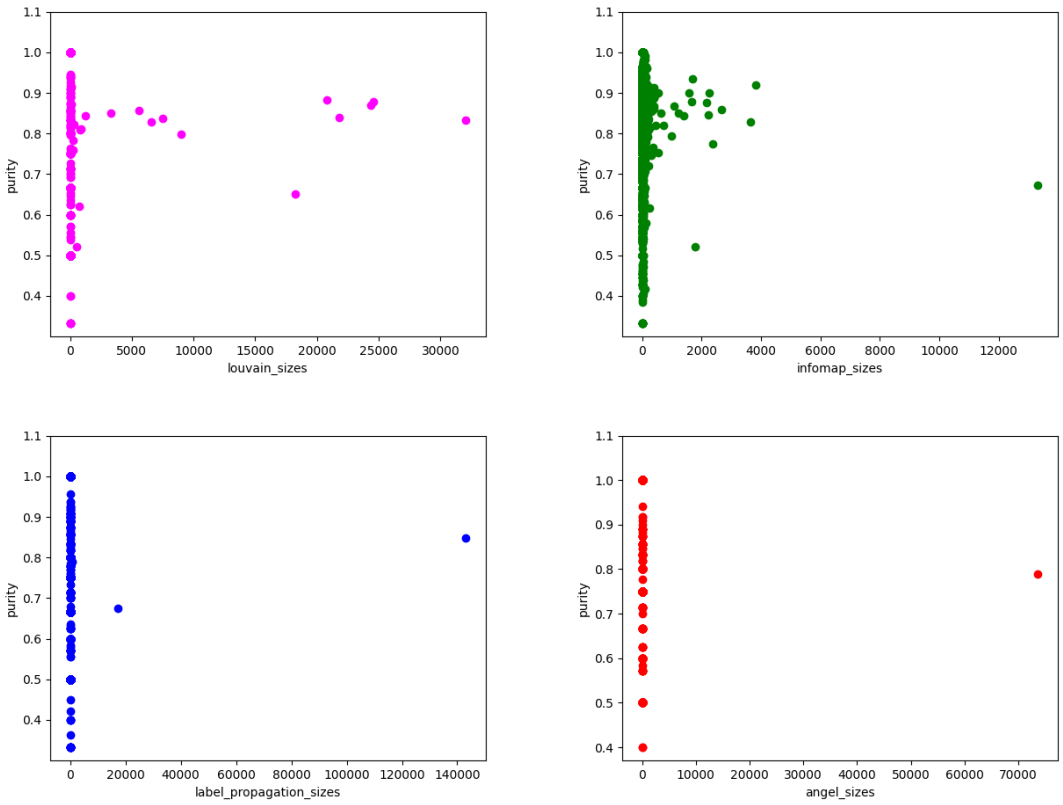

Figure 4.3: Louvain (top left), Infomap (top right), Label Propagation (bottom left) and Angel (bottom right) size compared to purity.

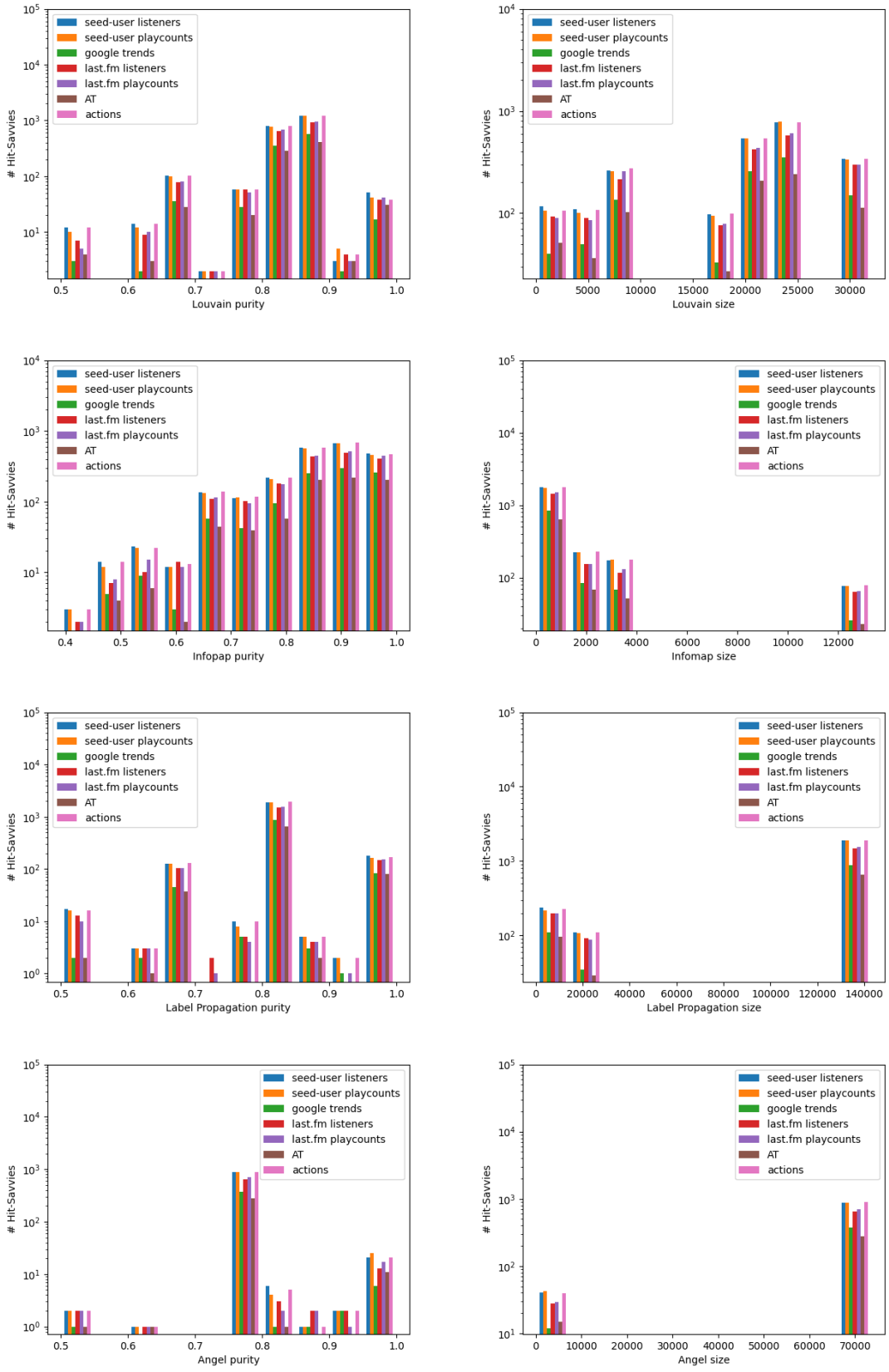

Figure 4.4: Louvain (top left), Infomap (top right), Label Propagation (bottom left) and Angel (bottom right) size compared to purity and size for my Hit-Savvies.

My main interest in the community detection is related to the evaluation of the level of purity of each community, to understand if similar nodes clustered by the CD algorithms, by sharing common network structure characteristics, also shared the PLDM common traits identified with the previous Section's clustering. Figure 4.2 reports the purity distribution among the identified communities. A good amount of communities are 100% pure with respect to the clustering labels: 1,755 communities for Louvain, 5,143 for InfoMap, 6,124 for Label Propagation and 362 for Angel. Still, all the communities are quite mixed, but almost never drop above the 30% of purity. This underlines how the nodes that share the same community have, on average, the same characteristics as at least the 30% of the other nodes. Figure 4.3 compares purity in regard to communities size. From the clustering on the left, we can notice that the majority of communities have a small size and vary a lot on the purity axes. On the right, we observer a few larger communities, which never reach a 100% purity score due to their high heterogeneity. As the common intuition suggests, increasing the number of nodes within a community, broaden the diversity among them, modifying the initially purity score and since the purity was computing labeling the nodes with the LPDM model, this increase the variability of the adopters within the community, introducing potential users that share different musical tastes from the peer circle. To verified if my Hit-Savvies endured this scenario, I've kept track of the purity of the communities in which the Hit-Savvies were clustered. Figure 4.4 illustrates my results. With Louvain communities my Hit-Savvies are part of quite pure communities and since their size is significantly high, these communities are formed by users who share a common music pattern. On the other hand, the communities identified by Infomap are even strongly pure, but this time adopters are classified in small communities, still remaining within alike. Label propagation and Angel, on the other hand, mix large clusters with reduced one, undergoing an oscillatory purity distribution. The small communities could be the niche of listeners devoted to uncommon music genres or artists, as well as users less virtually and socially active. In contrast, the largest communities may count on users socially very active, with many followers and followings, who follow the latest trends or who just conform to the mass. Nevertheless, all these communities underline an intrinsic mixture of adoption patterns, music tastes and played artists who, even by taking into account their topological characteristics, are quite difficult to universally monitor, analyze and classify.

Chapter 5

Conclusions

The music panorama is a bundled maze, where is easy to enter but not as easy to find the escape. Not by chance it's one of the most exploited, studied and profitable businesses. Its compelling power attracts many and very different adopters, among which is not easy to identify strong patterns and exploit them to make predictions. With this first data analysis attempt and the increased experience endured with my work, I've tried to discover some fruitful traits that could unlock a better understanding of those special adopters having a propensity to adopt successful items. My main goal was to exploit them to achieve predictive performance and to evaluate their influential power over the network. Given the obtained results I could conclude that among a sparse adoption log:

1. it is not easy to quantify the success of an item by using an adoption trend approach[1], especially if expanding trends don't show a broader diffusion in comparison with contracting ones;

2. the identification of strong and reliable Hit-Savvies[2] is quite difficult, since they tend to be less than 1% of the total adopters and for the most part they are labeled as such thanks to the adoption of a handful of Hits through their entire adoption history;

3. in contrast, Flop-adopters definition[3] may be too wide and in arrangement with higher activity on Flop-adopters part, this may lead to a high specialisation in Flop-prediction and as a result in a Hit-prediction penalty;

4. surprisingly, the majority of Hitters aren't Leaders[4]. They continue to adopt innovations before they reach their global maxima, but they aren't the absolute first to spot these novelties[5], underling how influence and prediction aren't always correlated: a user may be useful to predict the success of unseen items but not to spread them among the network; instead a user may be a great item influencer, but lacking the

[1] Increasing or decreasing over time of item's number of new adopters.

[2] Niche of individuals by chance or luck, are constantly involved in the adoption of successful items.

[3] Niche of individuals by chance or luck, are involved in the adoption of mainly unsuccessful items or later Hits.

[4] Node with a great influence power through the network.

[5] In many cases they are influenced by their followings, who may or may not be Hit-Savvies on their counterpart.

propensity to adopt mainly successful items and so to be used as Hit-Savvy by the predictive model. As a suggestion, the HF-propensity computation could take into account Leaders labeled as Flop-adopters and reassigned them as Hit-Savvies if they show a greater propensity to adopt Hits and later Hits in comparison to Flops;

5. the predictive methodology is concerned with the selection of Hit-Savvies in one go, before the predictive period starts. In this case, if any one of the selected seeds does not perform up to expectation, then the number of predicted successful items will be lesser than expected. Considering this case, a framework of multi-phase Hit-Savvies selection could be implemented;

6. the restricted set of Hit-Savvies as well as Leaders, is composed by user's with low Width, Depth and Strength power, whose influence range is confined to mutual friendship boundaries, among which musical testes and listening patterns are the driven building agents, rather than the number and the variety of the same artists played;

7. in real-world social networks, users have specific topics of choice. So, one user will be influenced by other users if both of them have similar choices. Keeping 'topic' into consideration[6], the spread of influence may be increased even more.

[6]This problem is refereed as topic aware influence maximization[23]

Appendices

Definition 22 (*Social Influence Maximisation problem (SIM problem)*):
The Social Influence Maximisation problem has as goal the finding among a social network of a seed set of user k, such that by targeting these, the expected influence spread (defined as the expecting number of influenced users) is maximized. The social network is abstracted as a graph with the users as the vertex set and social ties among the users as the edge set. It is also assumed that the diffusion threshold (a measurement of how hard to influence the user and given in a numerical scale; more the value, more hard to influence the user) is given as the vertex weight and influence probability between two users as edge weight. In this settings, the SIM Problem is stated as follows: for a given size k ($k \in \mathbb{Z}+$), choose the set S of k nodes, such that $\sigma(S)$ gets maximize, where $\sigma(S)$ is the social influence function. For any given seed S, $\sigma(S)$ returns the set of influenced nodes, when the diffusion process is over. This problem was proved to be non-deterministic polynomial (NP-hard).

Definition 23 (*Topic-aware Influence Maximisation problem (TIM problem)*):
In real-world social networks, users have their own interests (which can be represented as topics) and are more likely to be influenced by their friends (or friends' friends) with similar topics. We can increase the influence spread by taking into consideration topics. To address this problem, the SIM problem[22] is revisited as following: given a topic-aware influence maximization (TIM) query, finds k seeds from a social network such that the topic-aware influence spread of the k seeds is maximized.

Definition 24 (*Induced subgraph*):
Let G = (V, E) be a graph with a set of nodes V and set of edges E. If G'= (V', E') is an induced subgraph, then: N' $\subset$ N and E' $\subset$ E, where E' corresponds to the subset of edges in E relating to nodes in N'.

Definition 25 (*Minimum diffusion tree*):
Given a graph G*, an adopted object ψ, a set of action $a_{*,\psi}$ performed on ψ by some adopter *, the induced subgraph $G_\psi = (V_\psi, E_\psi)$ containing ψ's adopters, the directed connected component $C_\psi \subset G_\psi$ of ψ's leader $l \in L_\psi \subset V_\psi$, the minimum diffusion tree $T_{l,\psi} \subset C_\psi$ is the Minimum spanning tree (MST) having its root in* l *and built minimizing the temporal label assigned at the edges.*

Definition 26 (*Sturges estimator*):
Binning formula, where the number of bins is the base 2 log of dataset X's size n:

$$n_h = \lceil log_2 n + 1 \rceil$$

and binwidth is:

$$h = \left\lceil \frac{max(x) - min(x)}{n_h} \right\rceil$$

This estimator assumes normality of data and is too conservative for larger, non-normal datasets. This is the default method in R's hist method.

Definition 27 (*Doane estimator*):
Improved version of Sturges' binning formula[26] *that produces better estimates for non-normal dataset. This estimator attempts to account for the skew of the data X with cardinality* n*, resulting in:*

$$n_h = \left\lceil 1 + log_2(n) + log_2(1 + \frac{|g_1|}{\sigma_{g_1}} \right\rceil$$

number of bins, where:

$$g_1 = mean[\frac{x - \mu}{\sigma}]^3$$

$$\sigma_{g_1} = \sqrt{\frac{6(n-2)}{(n+1)(n+3)}}$$

is respectively X's skewness (third standardized moment $\tilde{\mu}_3$) and X's skewness standard deviation. The binwidth is found as:

$$h = \left\lceil \frac{max(x) - min(x)}{n_h} \right\rceil$$

Definition 28 (*In-degree*):
The in-degree is the number of incoming edges onto a node:

$$k_i^{in} = \sum_j a_{i,j}$$

in terms of the adjacency matrix A of a graph.

Definition 29 (*Out-degree*):
The out-degree is the number of outcoming edges onto a node:

$$k_i^{out} = \sum_j a_{j,i}$$

in terms of the adjacency matrix A of a graph.

Definition 30 (*Average neighbor in-degree*):
The average neighbor in-degree of a node i is:

$$k_{nn,i,in} = \frac{1}{|N(i)|} \sum_{j \in N(i)} k_j^{in}$$

where N(i) are the neighbors of node i and k_j^{in} is the in-degree of node j which belongs to N(i).

Definition 31 (*Average neighbor out-degree*):
The average neighbor out-degree of a node i is:

$$k_{nn,i,out} = \frac{1}{|N(i)|} \sum_{j \in N(i)} k_j^{out}$$

where N(i) are the neighbors of node i and k_j^{out} is the out-degree of node j which belongs to N(i).

Definition 32 (*Clustering*):
For unweighted graphs, the clustering of a node u is the fraction of possible triangles through that node that exist:

$$c_u = \frac{2T(u)}{deg(u)deg(u-1)}$$

where $T(u)$ is the number of triangles through node u and deg(u) is the degree of node u.

Definition 33 (*Betweenness centrality*):
Betweenness centrality of a node u is the sum of the fraction of all-pairs shortest paths that pass through u:

$$c_B = \sum_{s,t \in V} \frac{\sigma(s,t|u)}{\sigma(s,t)}$$

where V is the set of nodes, $\sigma(s,t)$ is the number of shortest (s, t)-paths, and $\sigma(s,t|u)$ is the number of those paths passing through some node u other than s, t. If $s = t$, $\sigma(s,t) = 1$, and if $u \in s, t$, $\sigma(s,t|u) = 0$.

Definition 34 (*Closeness centrality*):
Closeness centrality of a node u is the reciprocal of the sum of the shortest path distances from u to all n-1 other nodes. Since the sum of distances depends on the number of nodes in the graph, closeness is normalized by the sum of minimum possible distances n-1:

$$C(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)}$$

where d(v, u) is the shortest-path distance between v and u, and n is the number of nodes in the graph.

Definition 35 (*Pagerank*):
Pagerank computes a ranking of the nodes in the graph G based on this three distinct factors that of a node:

- *the number of links it receives: the more links a node attracts, the more important it is perceived;*
- *the link propensity of the linkers: links coming from parsimonious nodes are worthier than those emanated by spendthrift ones;*
- *the centrality of the linkers: links from important vertices are more valuable than those from obscure ones.*

It was originally designed as an algorithm to rank web pages. Let $A = (a_{i,j})$ be the adjacency matrix of a directed graph. The PageRank centrality x_i of node i is given by:

$$x_i = \alpha \sum_k \frac{a_{k,i}}{d_k} x_k + \beta$$

where α and β are constants and d_k is the out-degree of node k if such degree is positive, or $d_k = 1$ if the out-degree of k is null. In matrix form we have:

$$x = \alpha x D^{-1} A + \beta$$

where β is now a vector whose elements are all equal a given positive constant and D^{-1} is a diagonal matrix with i-th diagonal element equal to $1/d_i$. Notice that, PageRank is determined by an endogenous component that takes into consideration the network topology and by an exogenous component that is independent of the network structure. It follows that x can be computed as:

$$x = \beta(I - \alpha D^{-1}A)^{-1}$$

Definition 36 (*Pearson correlation coefficient*):
The Pearson correlation coefficient ρ also known as Pearson product-moment correlation coefficient (PPMCC) is a measure of the linear correlation between two variables X and Y. According to the Cauchy–Schwarz inequality it has a value between 1 and -1, where:

- *1 is total positive linear correlation, which means that for every positive increase in one variable, there is a positive increase of a fixed proportion in the other ($Y = a + bX$, $b > 0$);*
- *-1 is total negative linear correlation, which means that for every positive increase in one variable, there is a negative decrease of a fixed proportion in the other (($Y = a + bX$, $b < 0$));*
- *0 is no linear correlation, which means that for every increase, there isn't a positive or negative increase. The two variables just aren't related.*

The absolute value of the correlation coefficient gives indeed the relationship strength and the larger the number, the stronger the relationship is. It is commonly represented by the Greek letter ρ (rho) and given a pair of random variables (X, Y), the formula for ρ is:

$$\rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$$

where:

- *σ_{xy} is the covariance between X and Y;*
- *σ_x is the standard deviation of X;*
- *σ_y is the standard deviation of Y.*

When applied to a sample the Pearson correlation coefficient is commonly represented by r_{xy} and may be referred to as the sample correlation coefficient*. Given paired data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ consisting of* n *pairs, r_{xy} is defined as:*

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where:

- n *is sample size;*
- *x_i, y_i are the individual sample points indexed with* i*;*

- $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ *and* $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ *the sample mean of X and Y.*

which generalized corresponds to:

$$r_{xy} = \frac{n\sum xy - (\sum x)(\sum y)}{\sqrt{[n\sum x^2] - [n\sum y^2]}}$$

Definition 37 (*Subgraph Isomorphism*):
Given a query graph $Q(V_q,\ E_q)$ and a graph G(V, E), a subgraph $G'(V', E')$ of G [24] is a subgraph isomorphic match to Q if exists a bijective function f: $V_q \rightarrow V'$ such that: an edge (u, v) is in E_q if and only if (f(u), f(v)) is in E'.

Definition 38 (*VF2 algorithm*):
The VF2 algorithm is a pattern matching algorithm, which tries to resolve the subgraph isomorphism problem [26]. The objective of the subgraph isomorphism problem is to find all subgraph isomorphic matches of a query graph Q in a given graph G. This problem is known to be NP-hard [27], but with a fixed query size, can be computed in polynomial time with respect to the query size. The VF2 algorithm is a ***tree search based algorithm****, whose outline for a graph pattern matching can be summarized as following:*

1. *retrieve feasible matches for each vertex in Q based on semantic and/or neighborhood information in G. This results in sets $\phi(0), ..., \phi(|V_q| - 1)$, where $\phi(i)$ is the set of feasible matches for vertex $i \in V_q$;*

2. *reduce the search space globally, refining the set of feasible matches for each query vertex;*

3. *optimize the search order of vertices in the query;*

4. *search the remaining space in a* ***depth-first*** *manner.*

Definition 39 (*Unsupervised model*):
An unsupervised model, is a learning algorithm trained using data that consists only of input vectors, with no specific target vectors or output that is expected after the data is processed.

Definition 40 (*K-Means algorithm*):
K-Means clustering is an unsupervised learning algorithm[39] that, as the name hints, finds a fixed number (k) of clusters in a set of data. A cluster is a group of data points that are grouped together due to similarities in their features. When using a K-Means algorithm, a cluster is defined by a centroid, which is a point at the center of a cluster. Every point in a data set is part of the cluster whose centroid is most closely located. K-Means starts by randomly defining k centroids (the number of centroids k is require in order for the algorithm to proceed. It can be guess, for example, using the Elbow criterion[41] or the Silhouette Coefficient[42]). From there, it works in iterative steps to perform two tasks:

1. *assign each data point to the closest corresponding centroid, using the Euclidean distance;*

2. *for each centroid, calculate the mean of the values of all the points belonging to it. The mean value becomes the new value of the centroid.*

Once step 2 is complete, all of the centroids have new values that correspond to the means of all of their corresponding points. These new points are put through steps one and two producing yet another set of centroid values. This process is repeated over and over until there is no change in the centroid values, meaning that they have been accurately grouped. Or, the process can be stopped when a previously determined maximum number of steps has been met.

Definition 41 (*Elbow criterion*):
The elbow method is a heuristic method of interpretation and validation of consistency within cluster analysis designed to help find the appropriate number of clusters in a dataset. The idea of the elbow method is to run K-Means clustering on the dataset for a range of values of k and for each value of k calculate the sum of squared errors (SSE):

$$SSE = \sum_{i=1}^{n} (x_i - \bar{x})^2$$

where n is the number of observations, x_i is the value of the i-th observation and $\bar{x}$ is the mean of all the observations. The SSE is used as a measure of variation within a cluster (if all cases within a cluster are identical the SSE would then be equal to 0). With the Elbow criterion, we plot a line chart of the SSE for each value of k. If the line chart looks like an arm, then the "elbow" on the arm is the value of k that is the best. The idea is that we want a small SSE, but that the SSE tends to decrease toward 0 as we increase k (the SSE is 0 when k is equal to the number of data points in the dataset, because then each data point is its own cluster, and there is no error between it and the center of its cluster). So our goal is to choose a small value of k that still has a low SSE, and the elbow usually represents where we start to have diminishing returns by increasing k.

Definition 42 (*Silhouette Coefficient*):
The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1, to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. Assuming that the data have been clustered via any technique (such as K-Means) into k clusters, for data point $i \in C_i$ (data point i in the cluster C_i), let:

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i,\, i \neq j} d(i, j)$$

be the mean distance between i and all other data points in the same cluster, where d(i, j) is the distance (Euclidean distance or Manhattan distance) between data points i and j with $i \in C_i$, $j \in C_i$ (we divide by $|C_i| - 1$, because we do not include the distance d(i, i) in the sum) and a(i) can be interpreted as a measure of how well i is assigned to its cluster (the smaller the value, the better the assignment);

$$b(i) = \min_{k \in i} \frac{1}{C_k} \sum_{j \in C_k} d(i,j)$$

be the smallest mean distance of i to all points in any other cluster $C_k \neq C_i$. The cluster with this smallest mean dissimilarity is said to be the "neighboring cluster" of i because it is the next best fit cluster for point i. We now define the silhouette (value) of one data point i as:

$$s(i) = \frac{b(i) - a(i)}{\max((a(i), b(i))}$$

Definition 43 (*Dynamic Time Warping*):
Dynamic time warping finds the optimal non-linear alignment between two time series. Given two time series Q and C of the same length n where:

$$Q = q_1, q_2, ..., q_n$$

and

$$C = c_1, c_2, ..., c_n$$

construct an $n \times n$ matrix whose i,j-th element is the Euclidean distance between q_i and c_j. We want to find a path through this matrix that minimizes the cumulative distance. This path then determines the optimal alignment between the two time series. It should be noted that it is possible for one point in a time series to be mapped to multiple points in the other time series. Let's call the path W where:

$$W = w_1, w_2, ..., w_k$$

where each element of W represents the distance between a point i in Q and a point j in C:

$$w_k = (q_i - q_j)^2$$

So we want to find the path with the minimum Euclidean distance:

$$W^* = argsmin_W \sqrt{\sum_{k=1}^{K} w_k}$$

*The optimal path is found via **dynamic programming**, specifically the following recursive function:*

$$\gamma(i,j) = d(q_i, c_j) + min(\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1))$$

Definition 44 (*Weighted set cover problem*):
Given a set U of elements (called the universe), each one of them having associated a weight w_s and a collection S of m sets whose union equals the universe, the weighted set cover problem can be formulated as the following integer linear program:

$$minimize \sum_{s \in S} x_s$$
$$subject\ to \sum_{x \in S} x_s \geq 1 \forall e \in U$$
$$\forall x_s \in 0, 1 \forall s \in S$$

which corresponds respectively to:

- *minimize the weight of the elements in the sets;*
- *cover every element of the universe;*
- *every set is either in the set cover or not.*

Definition 45 (*Entropy*):
Given a random variable X, with possible outcomes x_i, each with probability $P_X(x_i)$, the entropy H(X) of X is defined as follows:

$$H(X) = -\sum_i P_X(x_i) \log_b P_X(x_i)$$

Definition 46 (*Shannon's source coding theorem*):
The source coding theorem establishes the limits to possible data compression. It shows that (in the limit, as the length of a stream of independent and identically-distributed random variable data tends to infinity) it is impossible to compress the data such that the code rate (average number of bits per symbol) is less than the Shannon entropy[45] of the source, without it being virtually certain that information will be lost. The source coding theorem for symbol codes places an upper and a lower bound on the minimal possible expected length of codewords as a function of the entropy of the input word (which is viewed as a random variable) and of the size of the target alphabet.

Definition 47 (*Huffman Coding*):
Huffman code is a particular type of optimal prefix code that is commonly used for lossless data compression.There are mainly two major parts in Huffman Coding:

1. *build a Huffman Tree from input characters;*
2. *traverse the Huffman Tree and assign codes to characters.*

Given an array of unique characters along with their frequency of occurrences as inputs, the steps to build the Huffman Tree are the follows:

1. *create a leaf node for each unique character and build a min heap of all leaf nodes (Min Heap is used as a priority queue. The value of frequency field is used to compare two nodes in min heap. Initially, the least frequent character is at root);*
2. *extract two nodes with the minimum frequency from the min heap;*
3. *create a new internal node with a frequency equal to the sum of the two nodes frequencies. Make the first extracted node as its left child and the other extracted node as its right child. Add this node to the min heap;*
4. *Repeat steps 2 and 3 until the heap contains only one node. The remaining node is the root node and the tree is complete.*

Once built the Hoffman Tree, in order to assign codewords to the nodes, we have to traverse the tree starting from the root and maintain an auxiliary array. While moving to the left child, write 0 to the array. While moving to the right child, write 1 to the array. When a leaf node is encountered we can print the array. The time complexity of the algorithm is O(nlogn), where n is the number of unique characters. If there are n nodes, extractMin() is called 2(n – 1) times in the min heap. Since extractMin() takes O(logn) time as it calles minHeapify().*

Definition 48 (*Random walker nodes and links visit rate*):
Given a network with n *nodes and weighted directed links $W_{\alpha\to\beta}$ between nodes $\alpha, \beta \in 1, 2, ..., n$ the conditional probability that a random walker steps from node α to node β is given by the relative link weight:*

$$p_{\alpha\to\beta} = \frac{W_{\alpha\to\beta}}{\sum_\beta W_{\alpha\to\beta}}$$

Assuming that the stationary distribution is given by p_α, it can be derived from the recursive system of equations:

$$p_\alpha = \sum_\beta p_\beta p_{\beta\to\alpha}$$

*However, to ensure a unique solution independent of where the random walker starts in the directed network, at a low rate τ the random walker teleports **to** a random node proportional to the total weights of the links to it. The stationary distribution is then given by:*

$$p_\alpha = (1-\tau)\sum_\beta p_\beta p_{\beta\to\alpha} + \tau\frac{\sum_\beta W_{\beta\to\alpha}}{\sum_{\alpha,\beta} W_{\beta\to\alpha}}$$

*To make the results even more robust to the teleportation parameter τ , it is also used an unrecorded teleportation steps, which add an extra step similar to the previous equation but with teleportation to a node proportional to the total weights of the links **from** the node:*

$$p^*_\alpha = (1-\tau)\sum_\beta p^*_\beta p_{\beta\to\alpha} + \tau\frac{\sum_\beta W_{\alpha\to\beta}}{\sum_{\alpha,\beta} W_{\beta\to\alpha}}$$

With this premises, the unrecorded visit rates on links $q_{\beta\to\alpha}$ and on nodes p_a can now be expressed as:

$$q_{\beta\to\alpha} = p^*_\beta p_{\beta\to\alpha}$$

$$p_\alpha = \sum_\beta q_{\beta\to\alpha}$$

This so called smart teleportation scheme ensures that the solution is independent of where the random walker starts in the directed networks with minimal impact on the results from the teleportation parameter:

- *a teleportation rate too close to 0 gives results that depend on how the random walker was initiated and should be avoided;*

- *a teleportation value equal to 1 corresponds to using the link weights as the stationary distribution.*

Definition 49 (*Jaccard similarity coefficient*):
The Jaccard coefficient measures similarity between two sample sets A and B, and is defined as the size of the intersection divided by the size of the union of the sample sets:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1, or, equivalently, by dividing the difference of the sizes of the union and the intersection of two sets by the size of the union:

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

Definition 50 (*Cosine similarity*):
Given two vectors of attributes A and B, the cosine similarity, cos(θ), is represented using a dot product and magnitude as:

$$similarity = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

where A_i and B_i are components of vector A and B respectively.

Definition 51 (*Modularity*):
Modularity measures the difference between the actual number of edges in a community and the expected number of such edges. We denote by e_c the actual number of edges in a community c*. The expected number of edges can be expressed as $\frac{K_c^2}{2m}$, where K_c is the sum of the degrees of the nodes in community* c *and* m *is the total number of edges in the network. This way of defining the expected number of edges is based on the so-called configuration model. Modularity is given then by:*

$$Q = \frac{1}{2m}\sum_{c}(e_c - \gamma\frac{K_c^2}{2m})$$

where $\gamma \not\geq 0$ is a resolution parameter. Higher resolutions lead to more communities, while lower resolutions lead to fewer communities. Optimising modularity is NP-hard [28], and consequentially many heuristic algorithms have been proposed.

Definition 52 (*Conductance*):
Conductance measures the fraction of total edge volume that points outside a graph community C:

$$\frac{b_c}{2e_c + b_c}$$

where b_c is the number of edges on C's boundaries and e_C is the number of edges internal to C.

Results

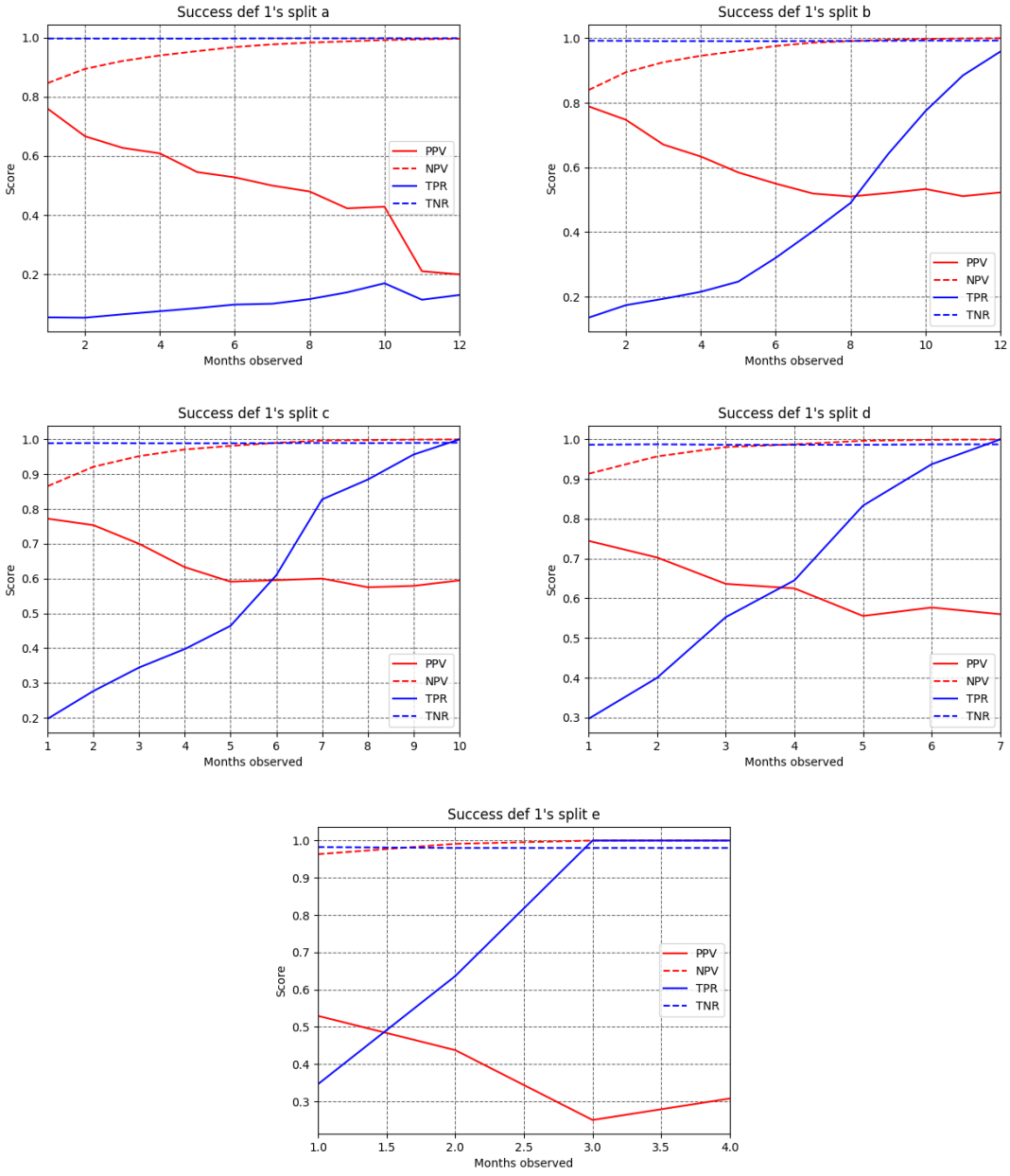

Figure A.1: Predictive accuracy varying the split and the observation period for the success definition 1.

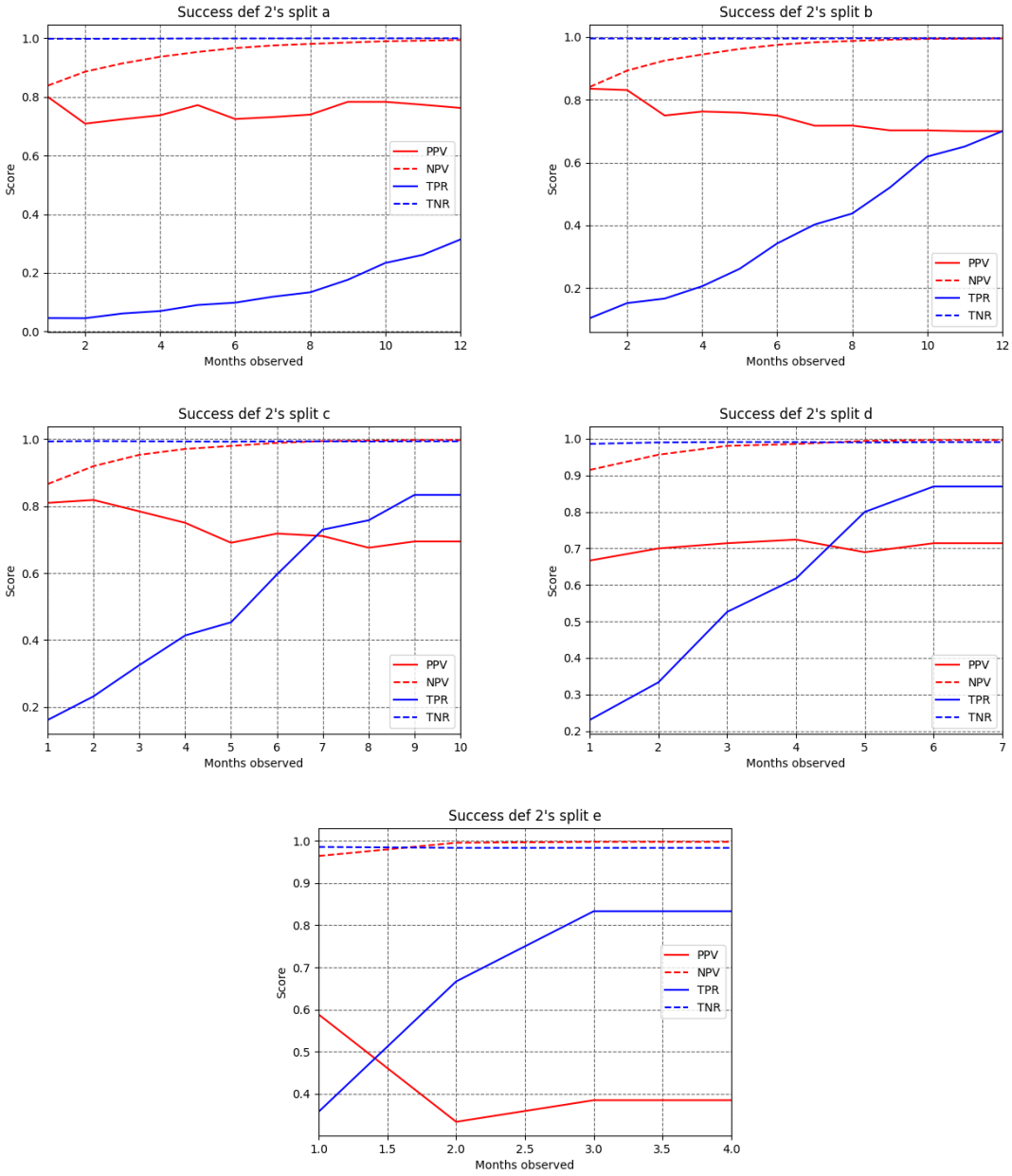

Figure A.2: Predictive accuracy varying the split and the observation period for the success definition 2.

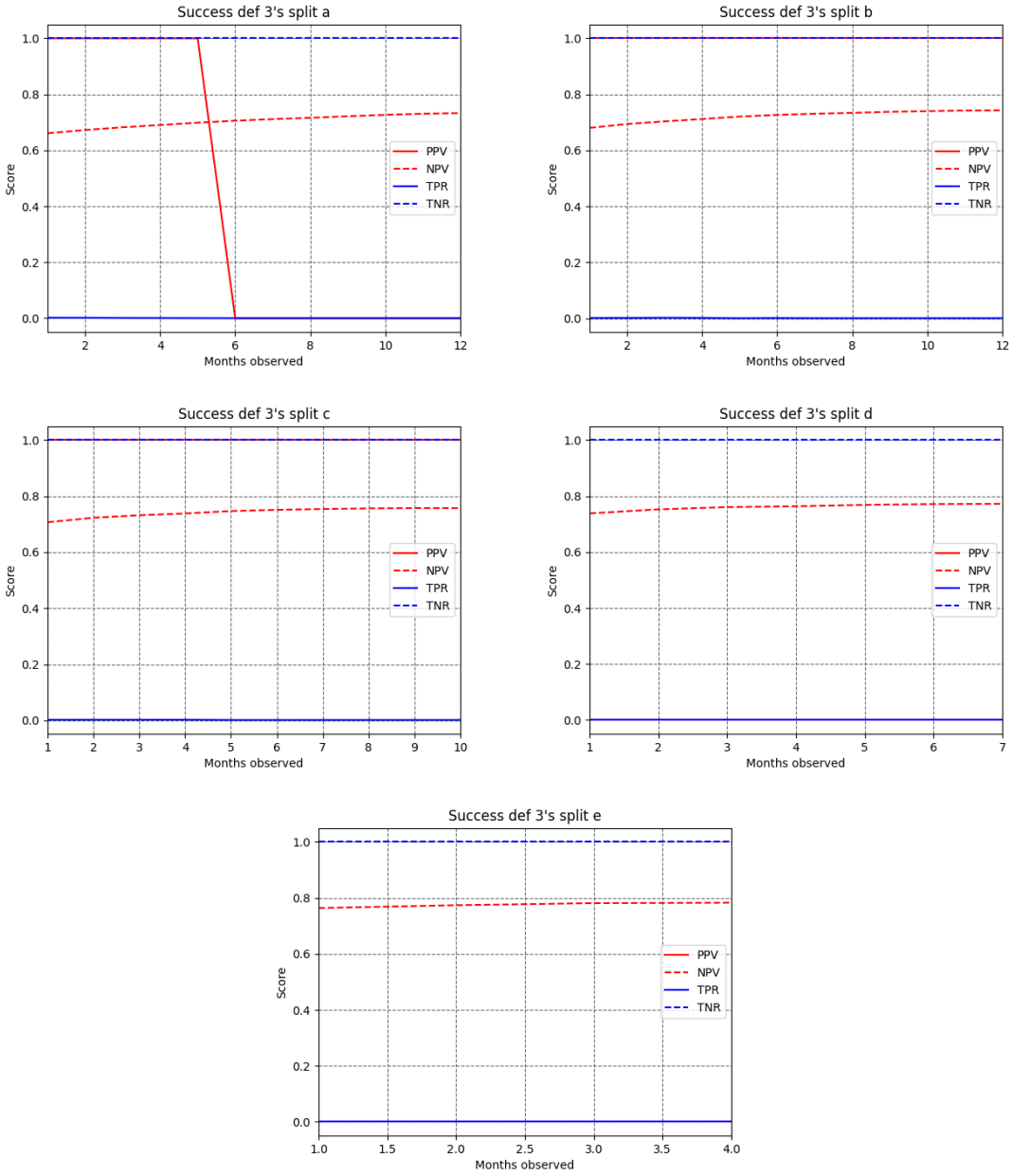

Figure A.3: Predictive accuracy varying the split and the observation period for the success definition 3.

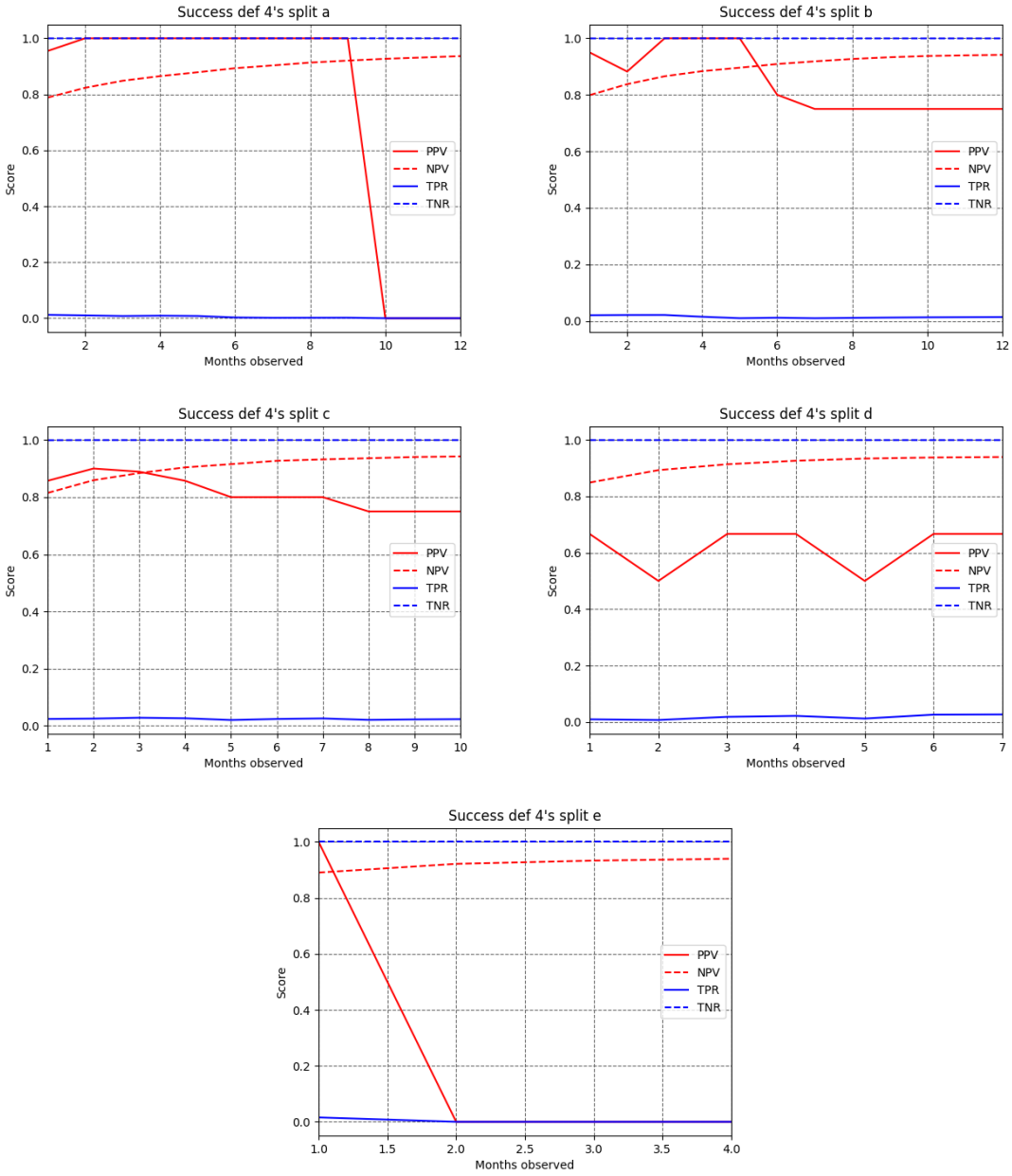

Figure A.4: Predictive accuracy varying the split and the observation period for the success definition 4.

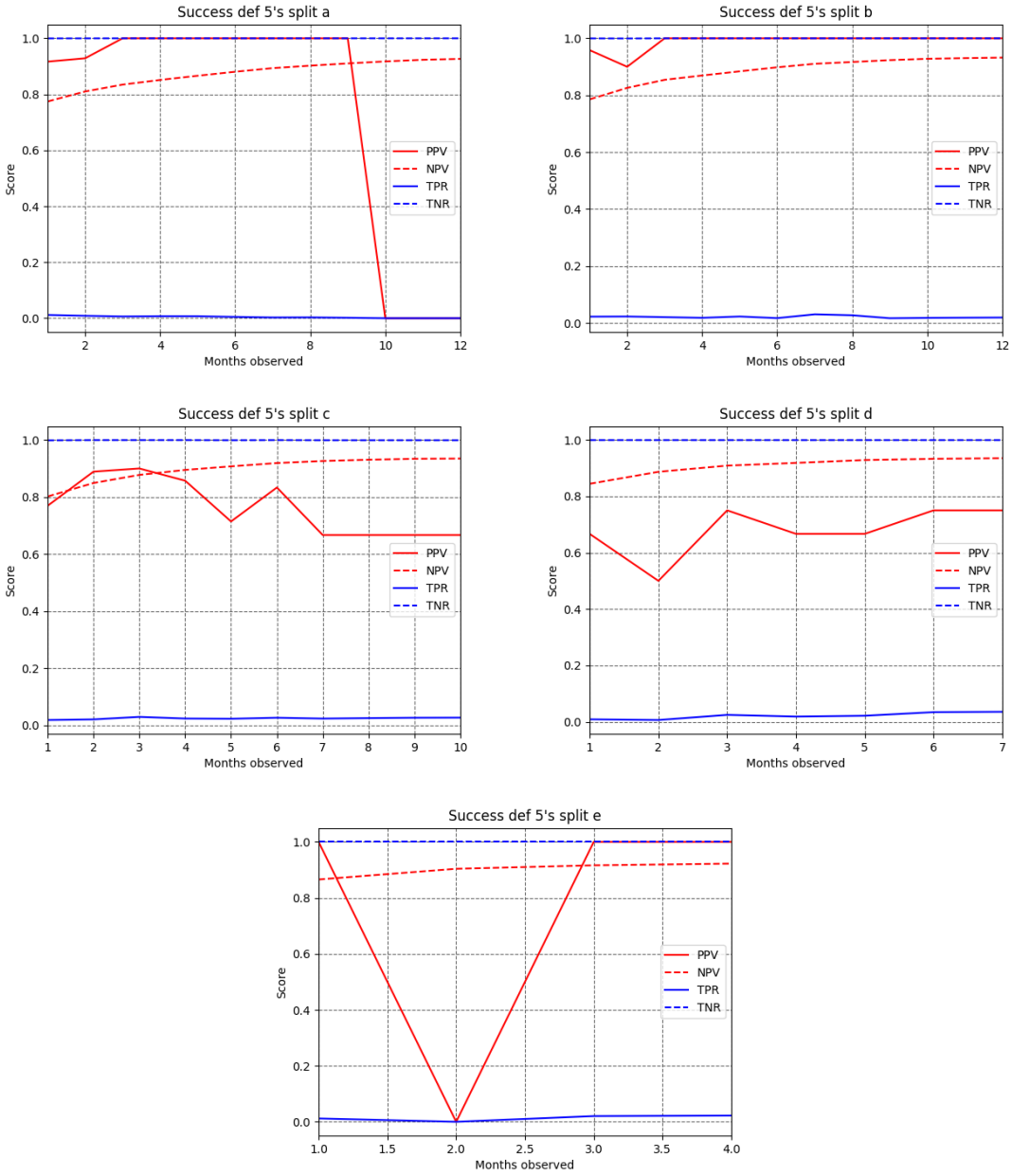

Figure A.5: Predictive accuracy varying the split and the observation period for the success definition 5.

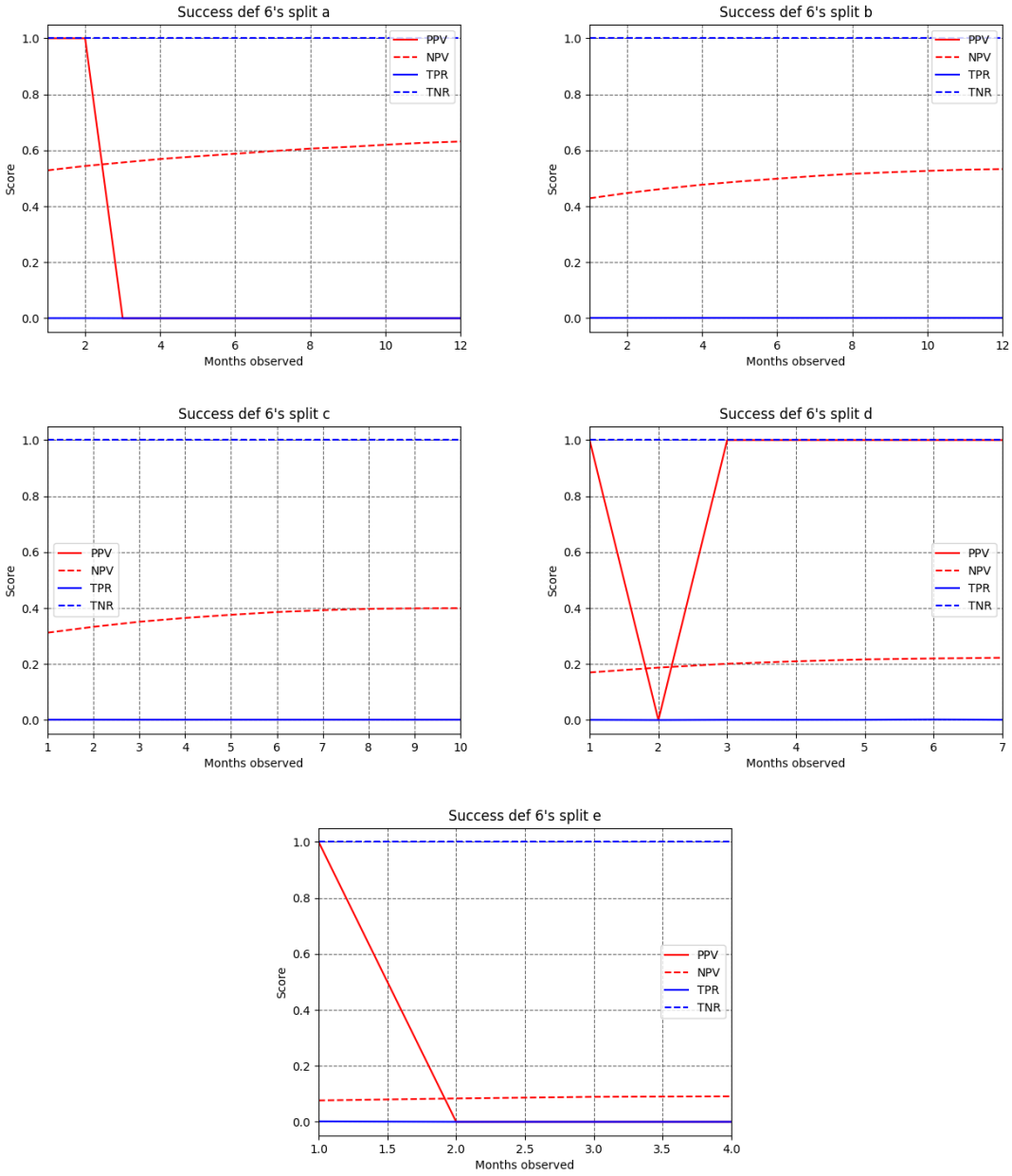

Figure A.6: Predictive accuracy varying the split and the observation period for the success definition 6.

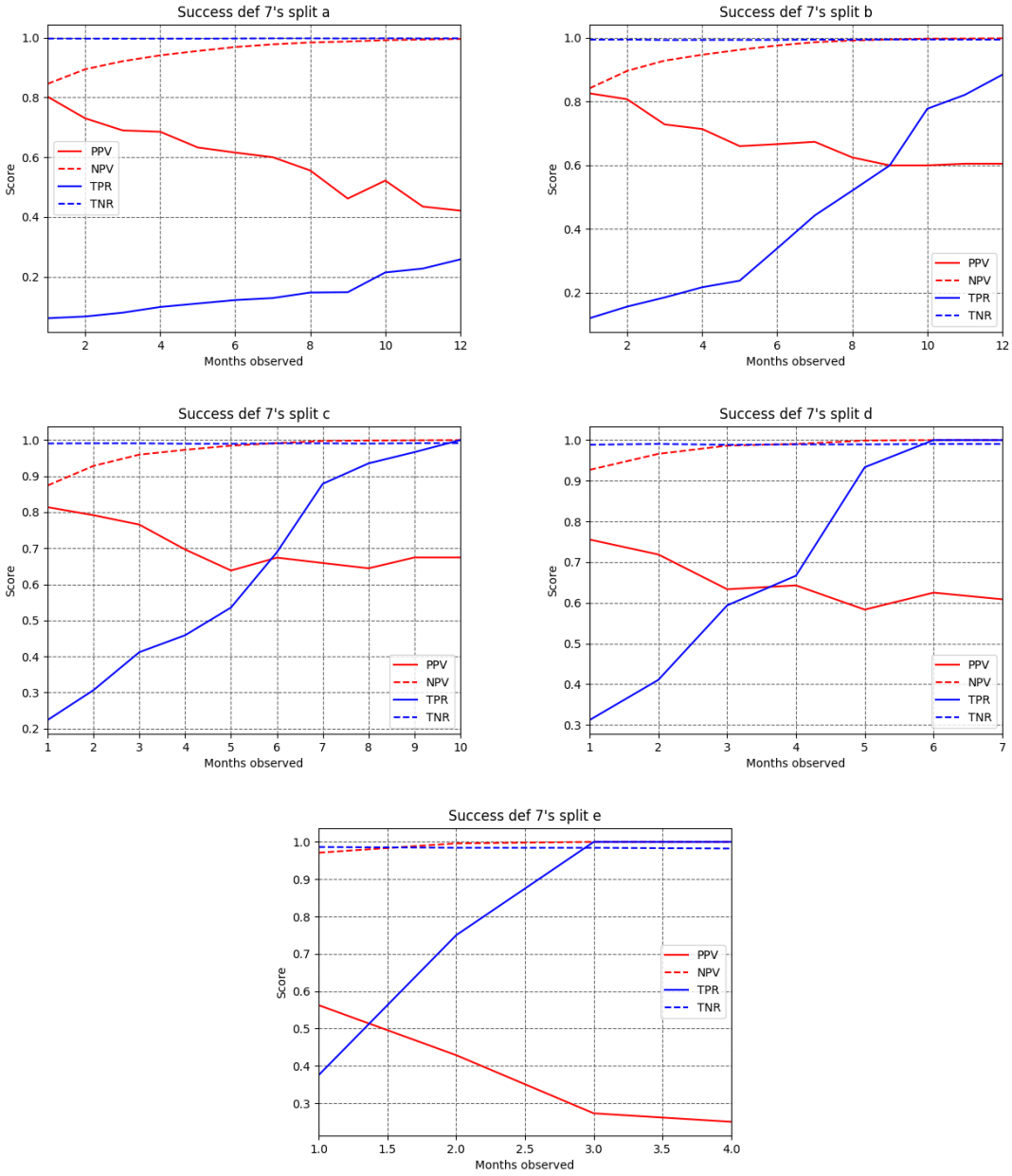

Figure A.7: Predictive accuracy varying the split and the observation period for the success definition 7.

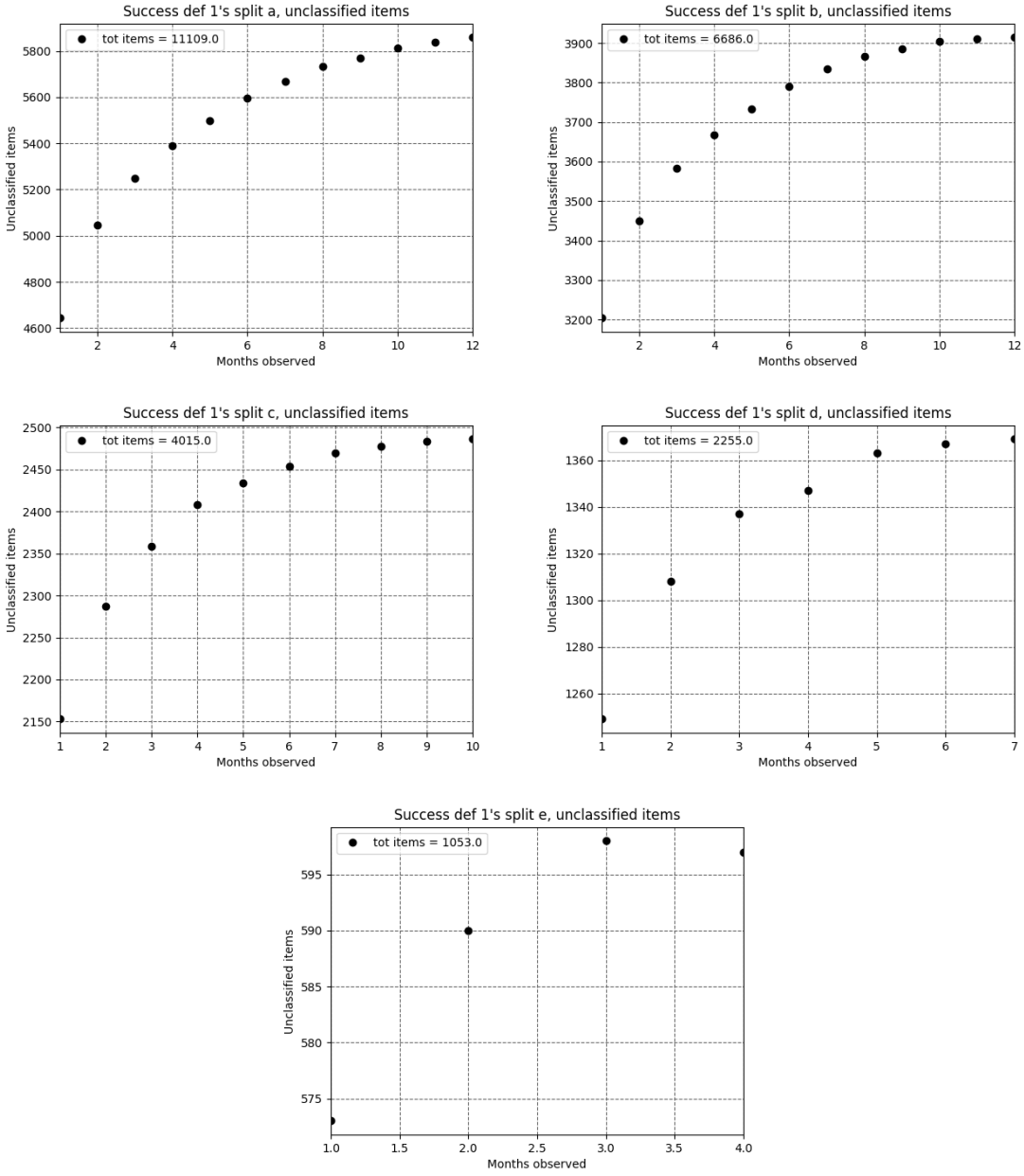

Figure A.8: Unclassified items for the success definition 1.

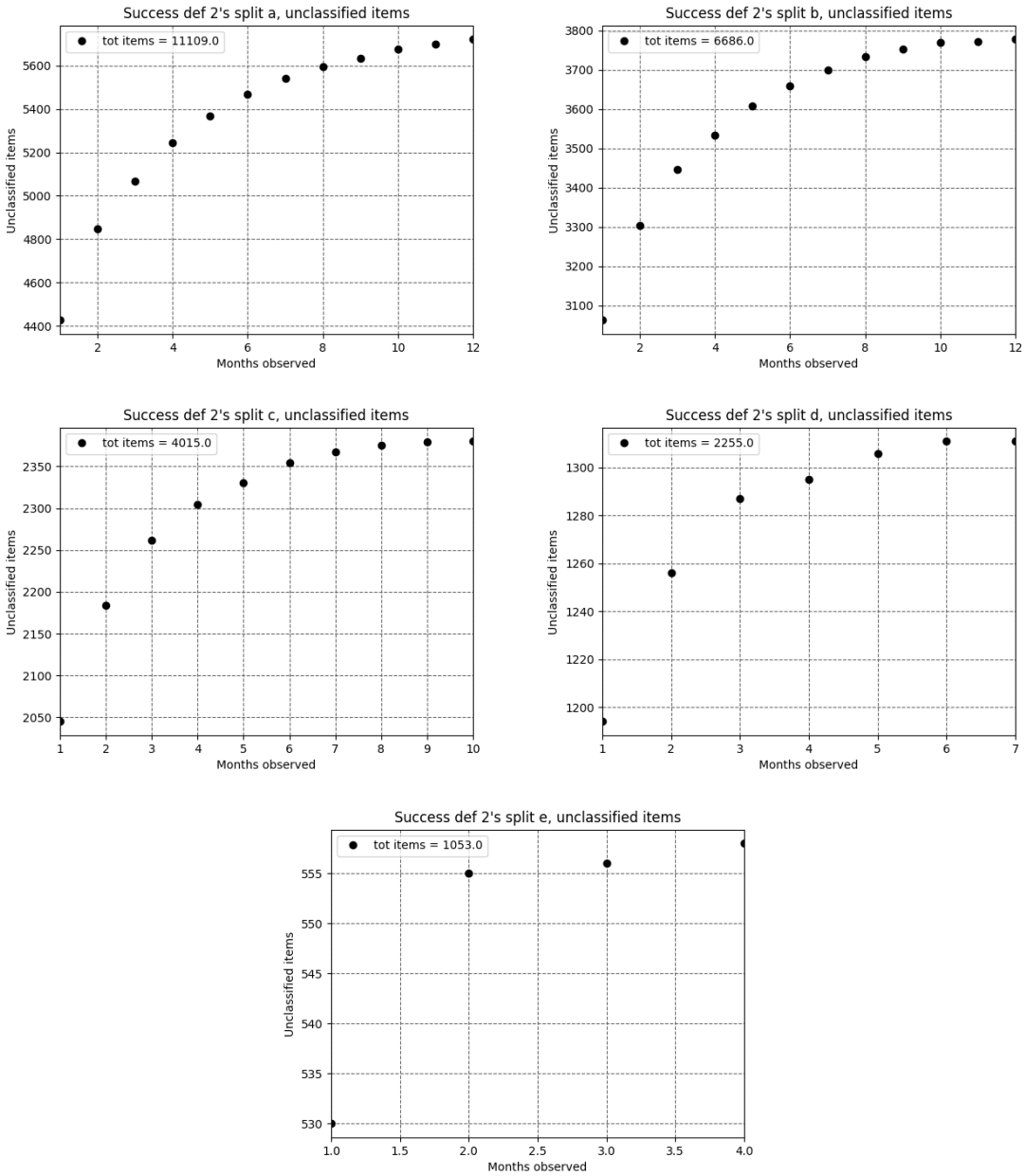

Figure A.9: Unclassified items for the success definition 2.

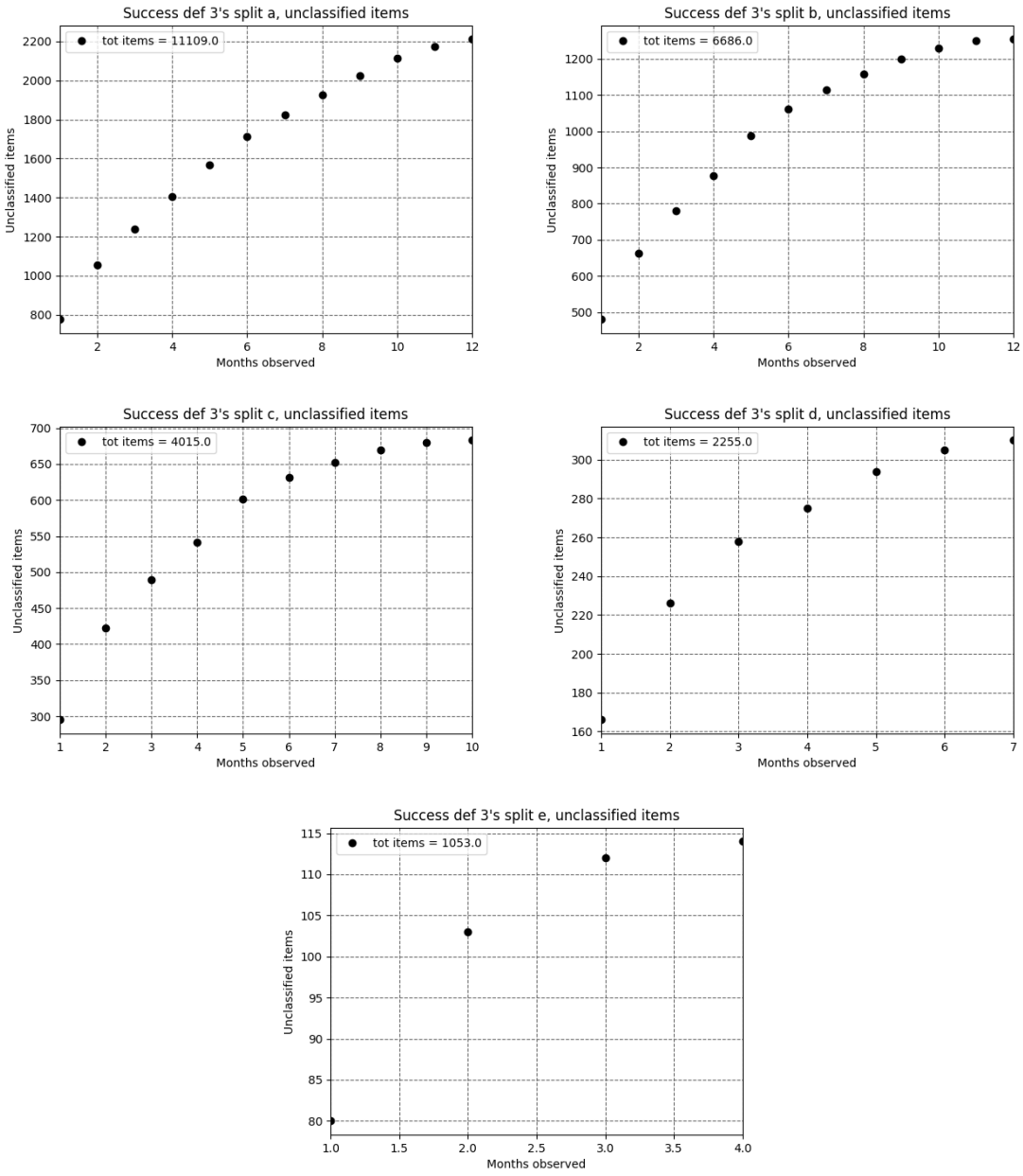

Figure A.10: Unclassified items for the success definition 3.

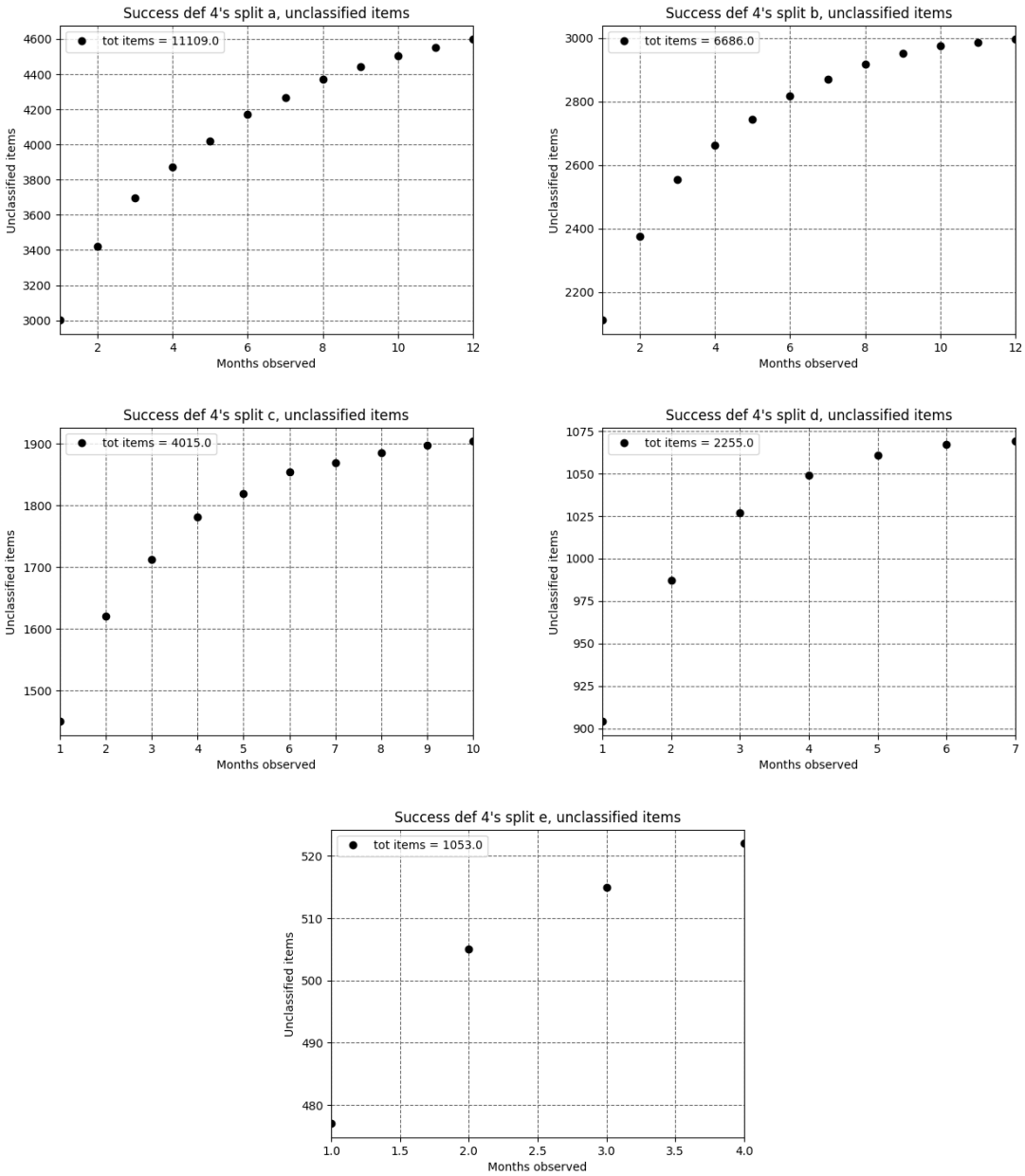

Figure A.11: Unclassified items for the success definition 4.

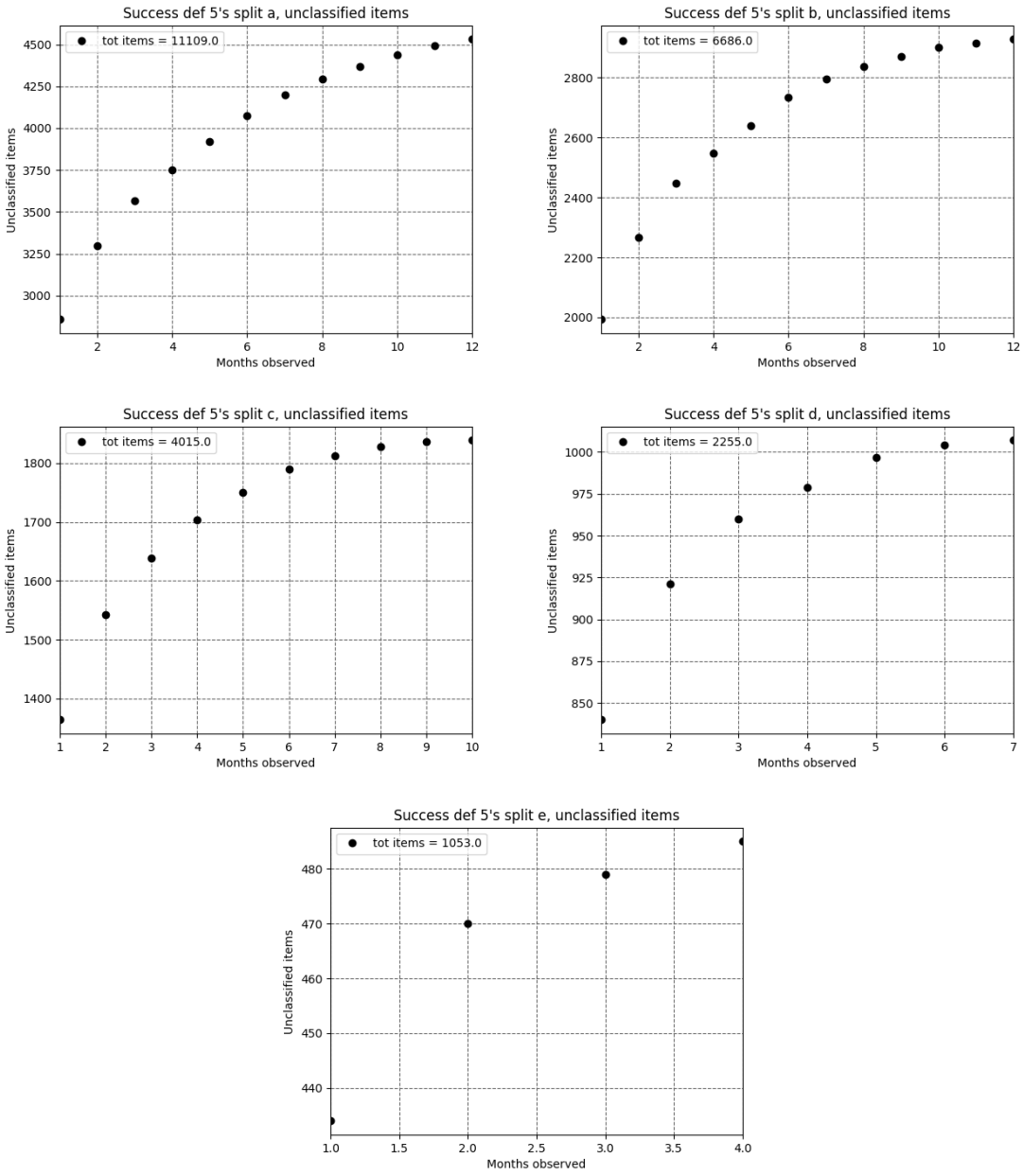

Figure A.12: Unclassified items for the success definition 5.

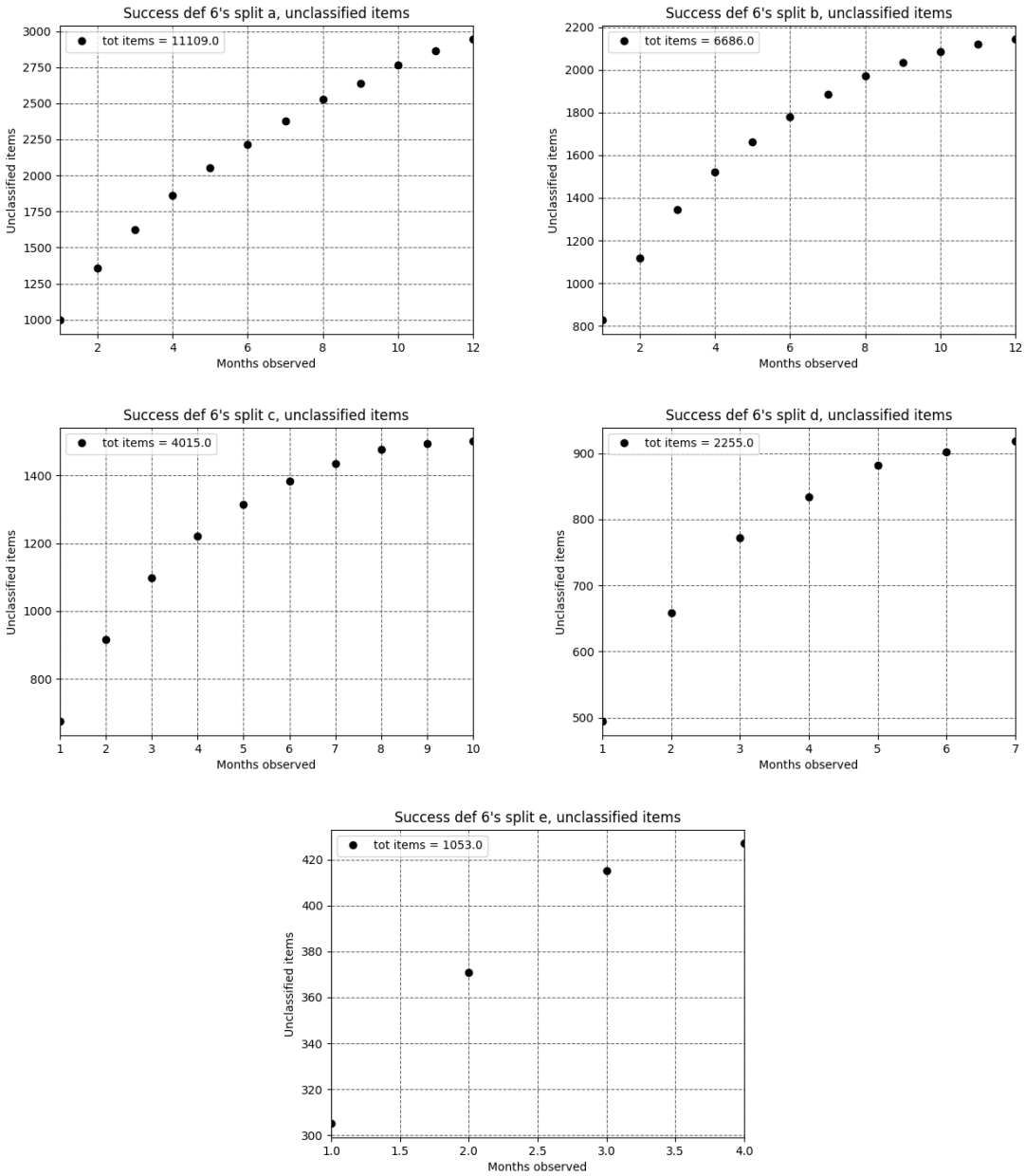

Figure A.13: Unclassified items for the success definition 6.

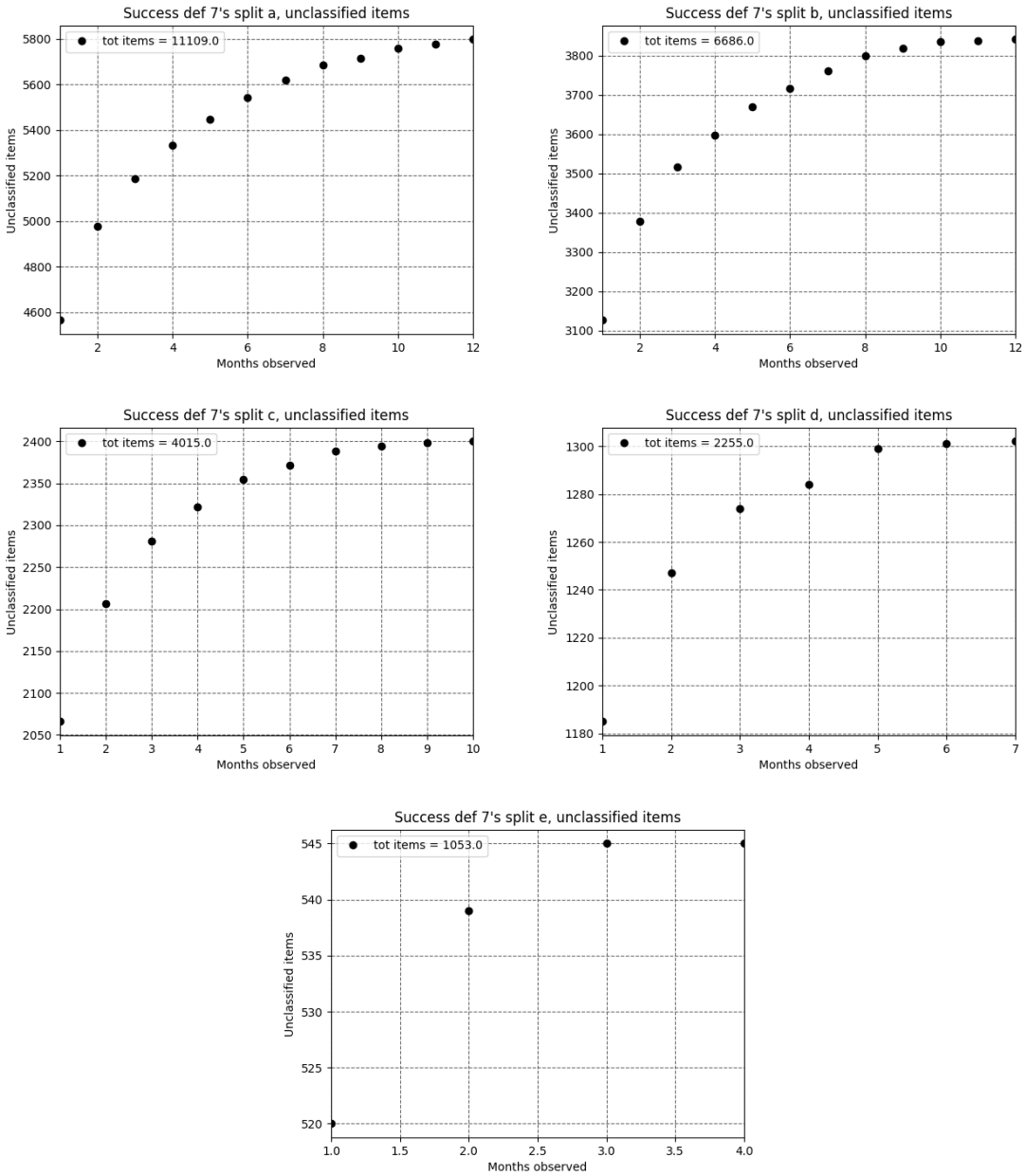

Figure A.14: Unclassified items for the success definition 7.

Pure Flop-adopters experiment

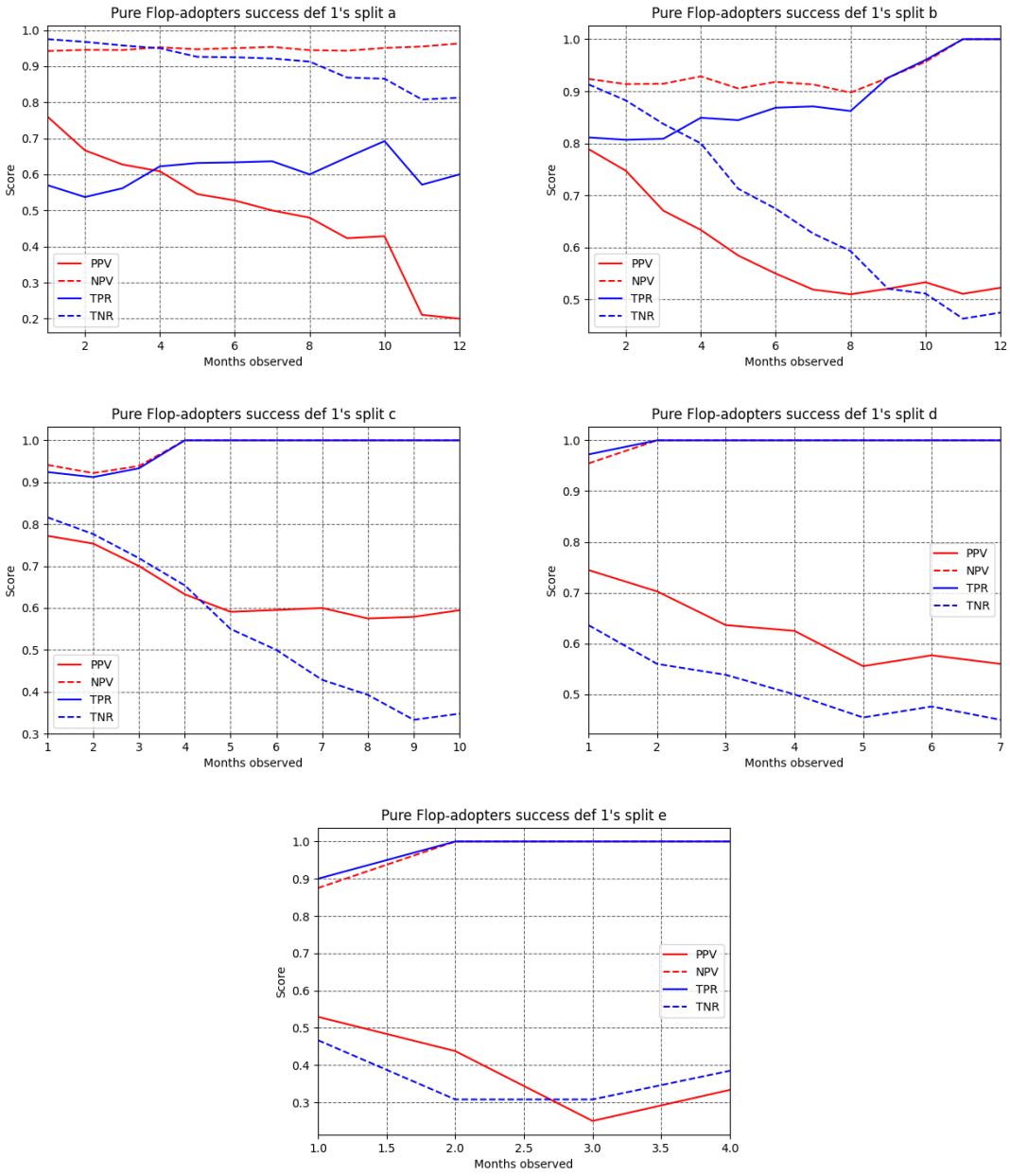

Figure A.15: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 1.

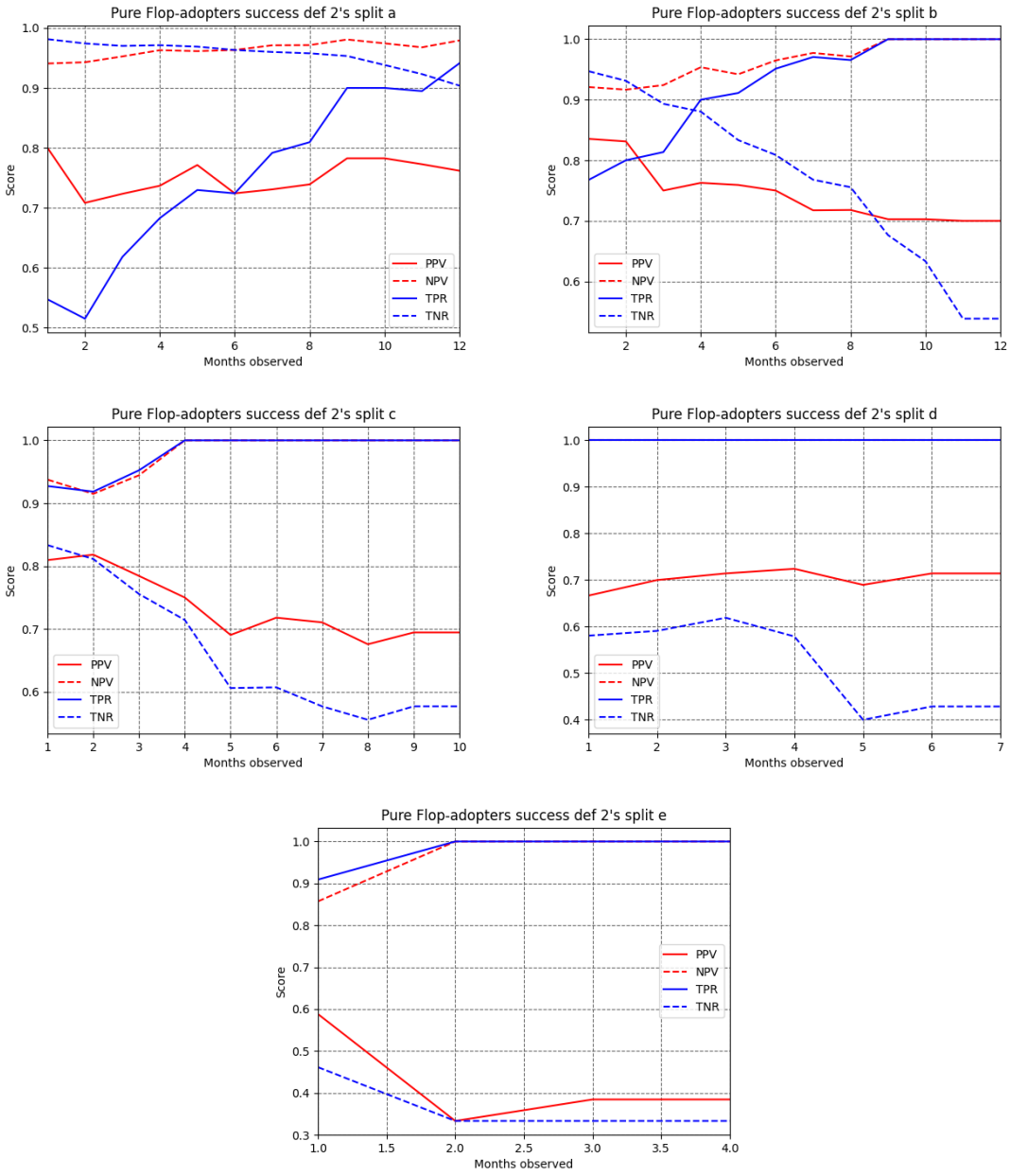

Figure A.16: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 2.

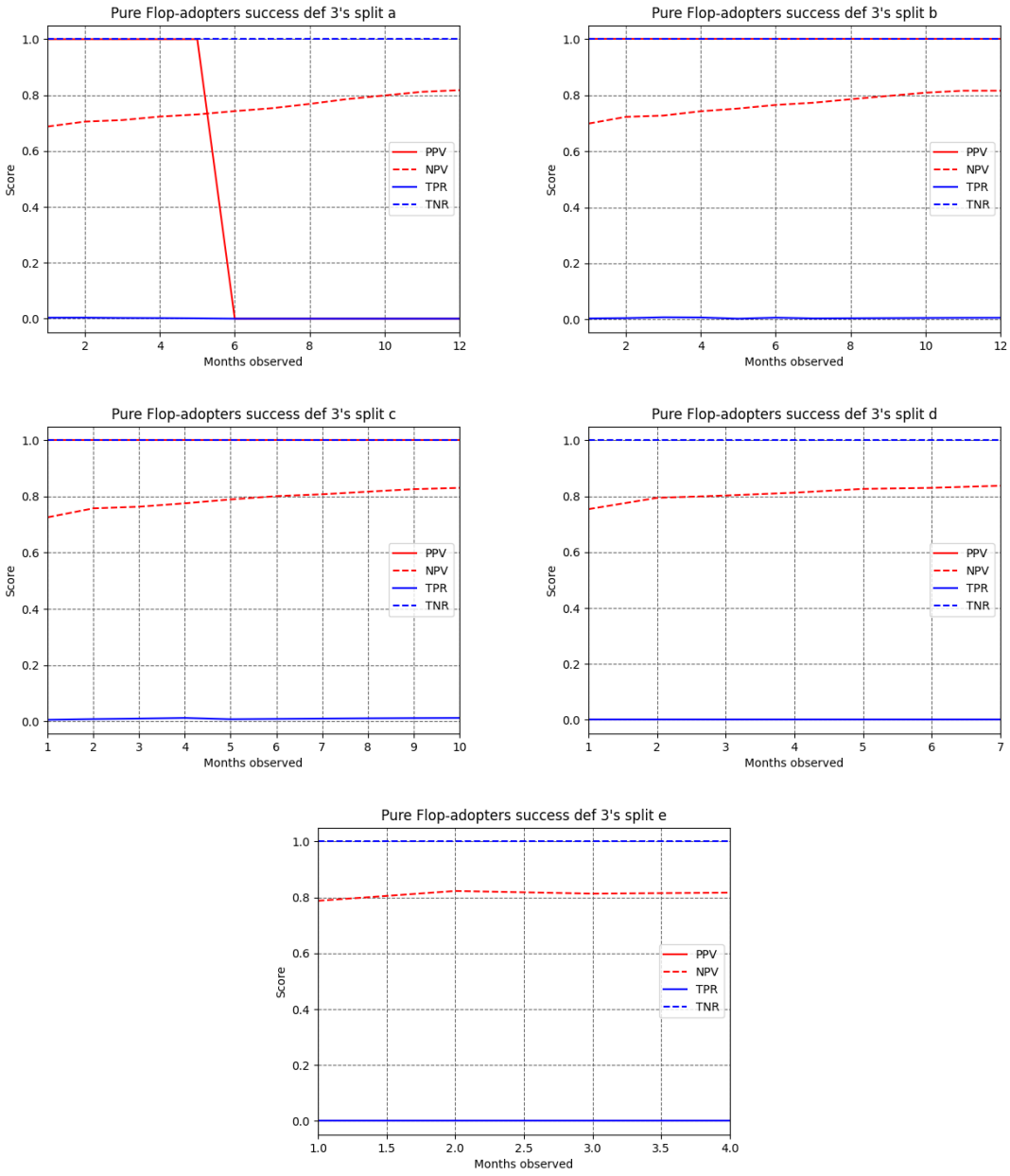

Figure A.17: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 3.

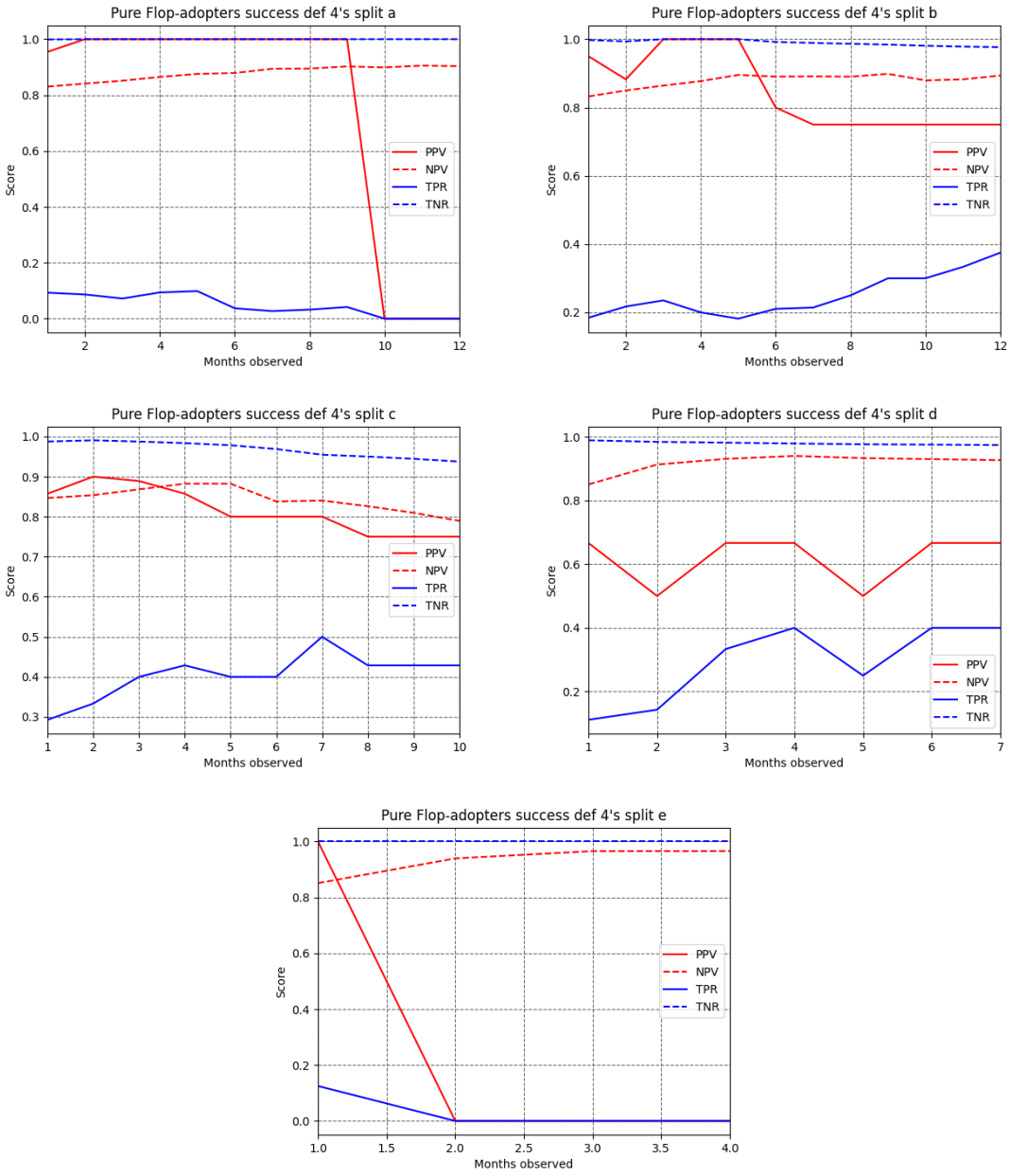

Figure A.18: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 4.

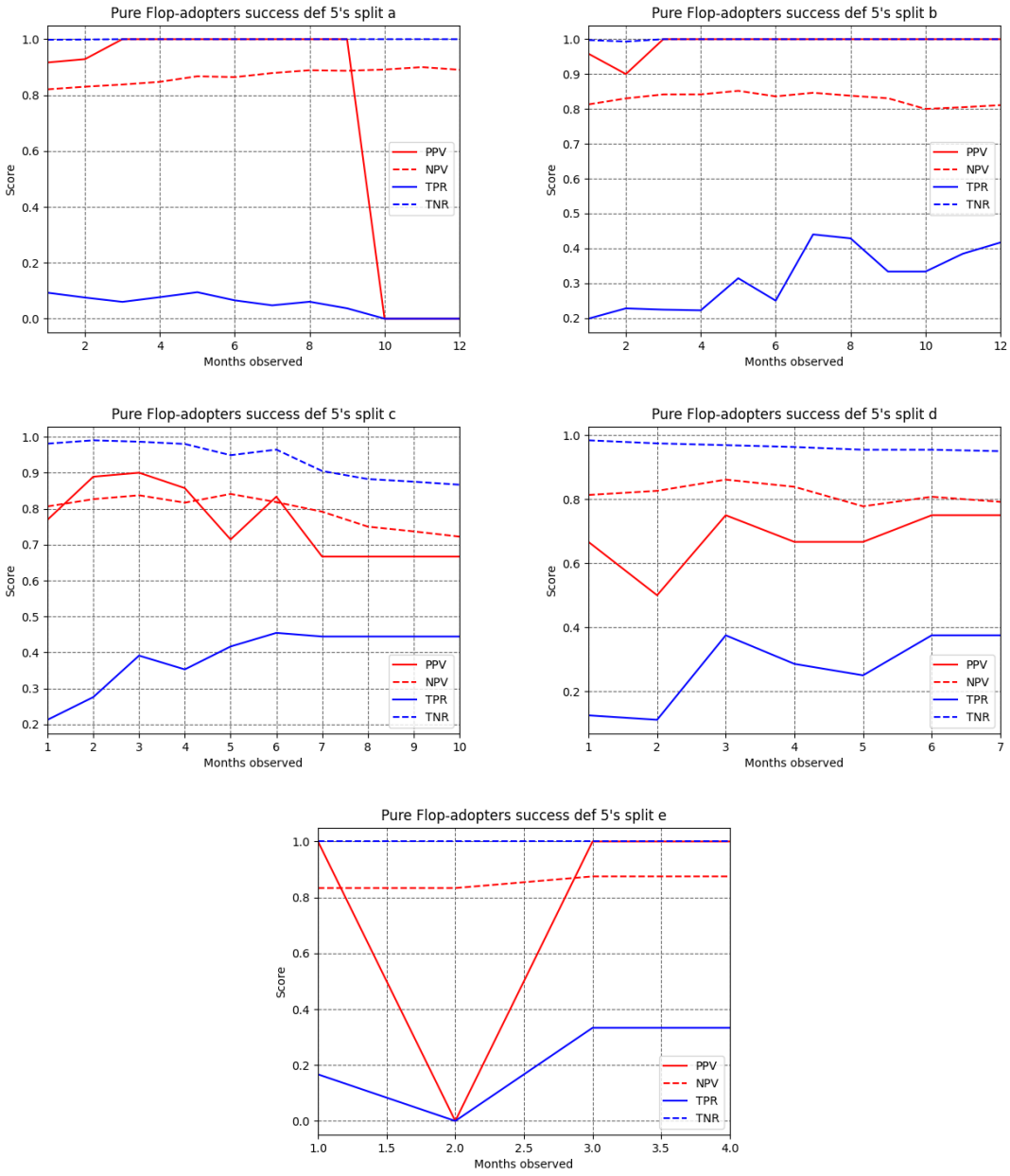

Figure A.19: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 5.

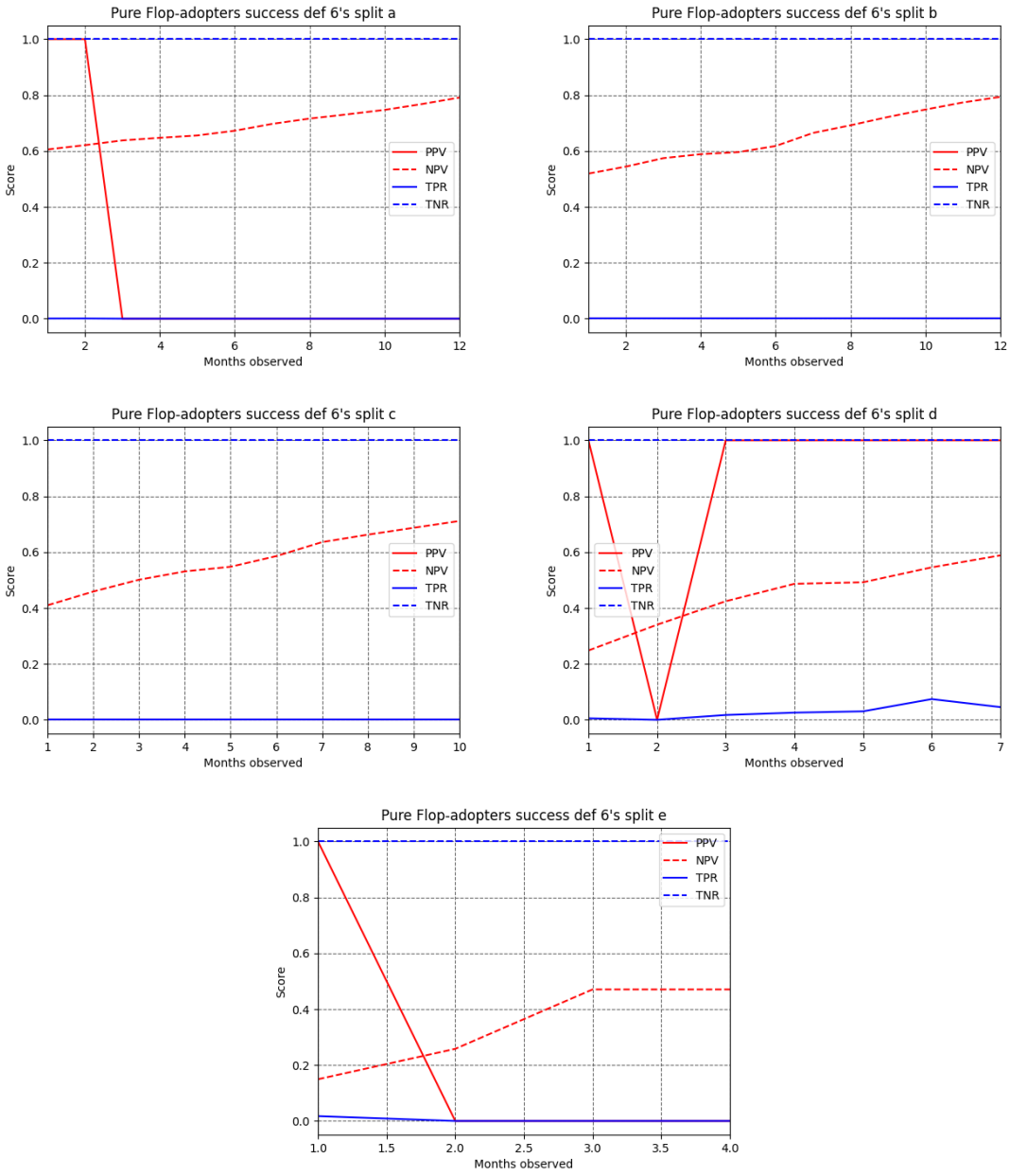

Figure A.20: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 6.

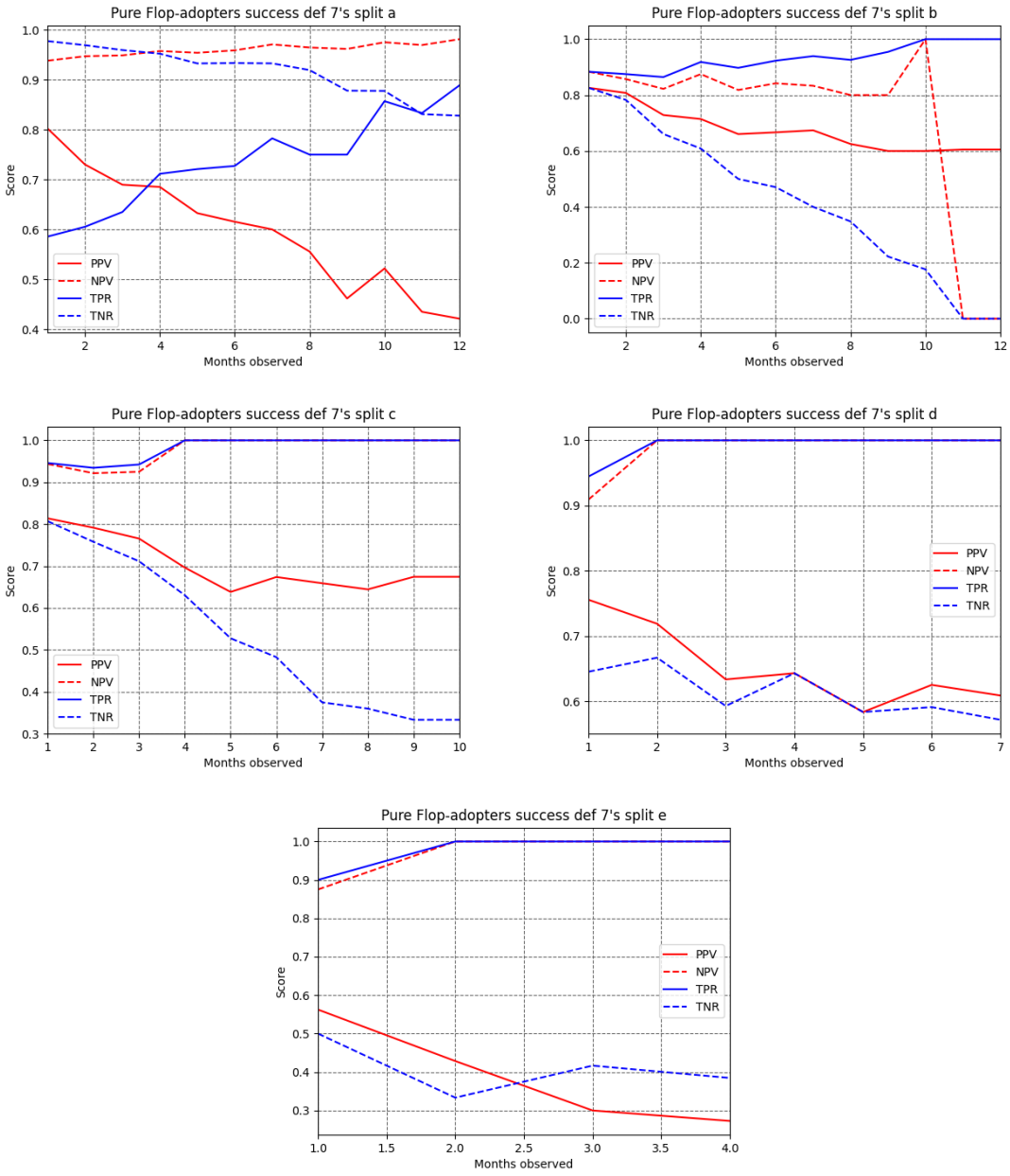

Figure A.21: Predictive accuracy varying the split and the observation period for the pure Flop-adopters experiment's success definition 7.

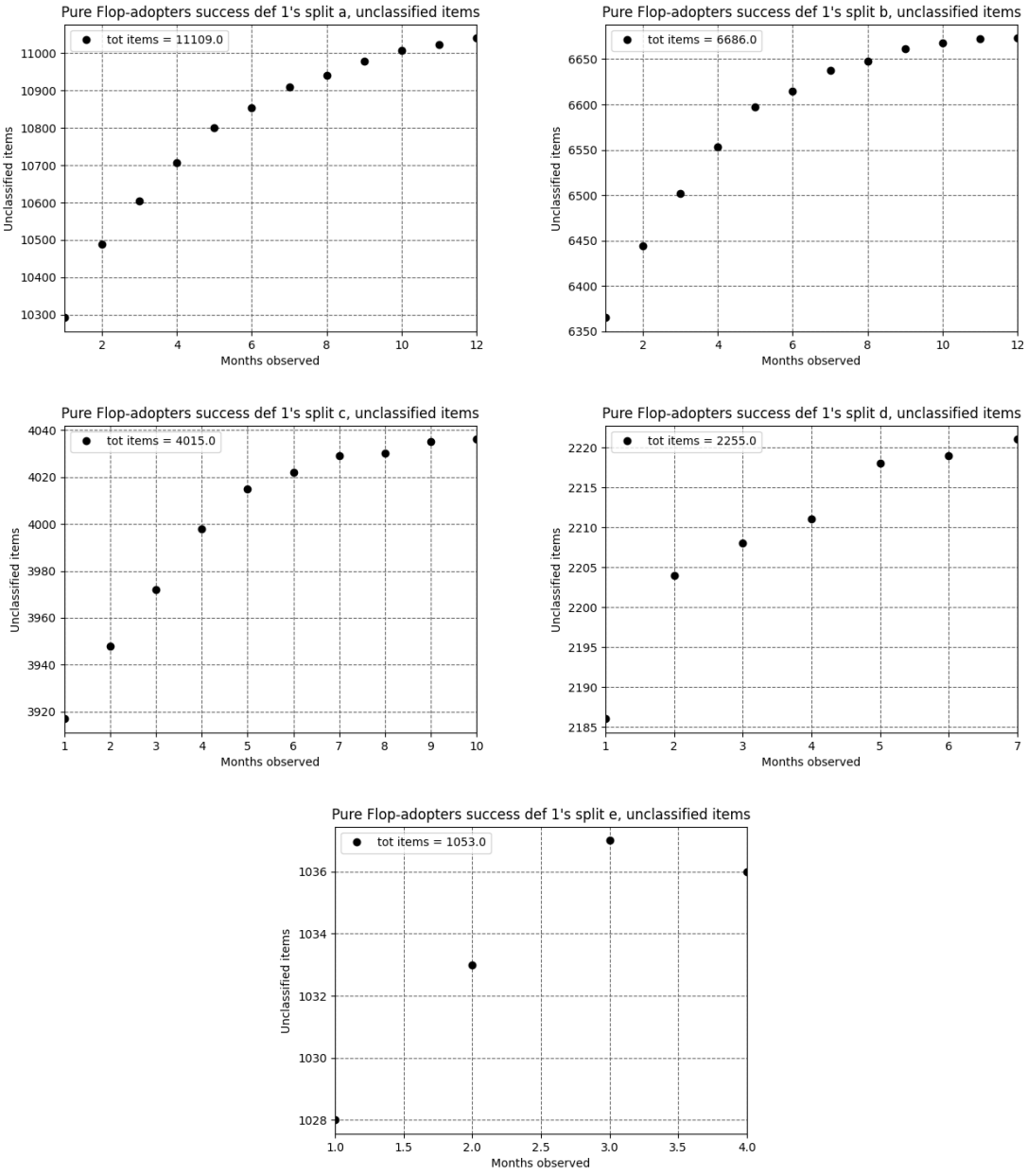

Figure A.22: Unclassified items for the pure Flop-adopters experiment's success definition 1.

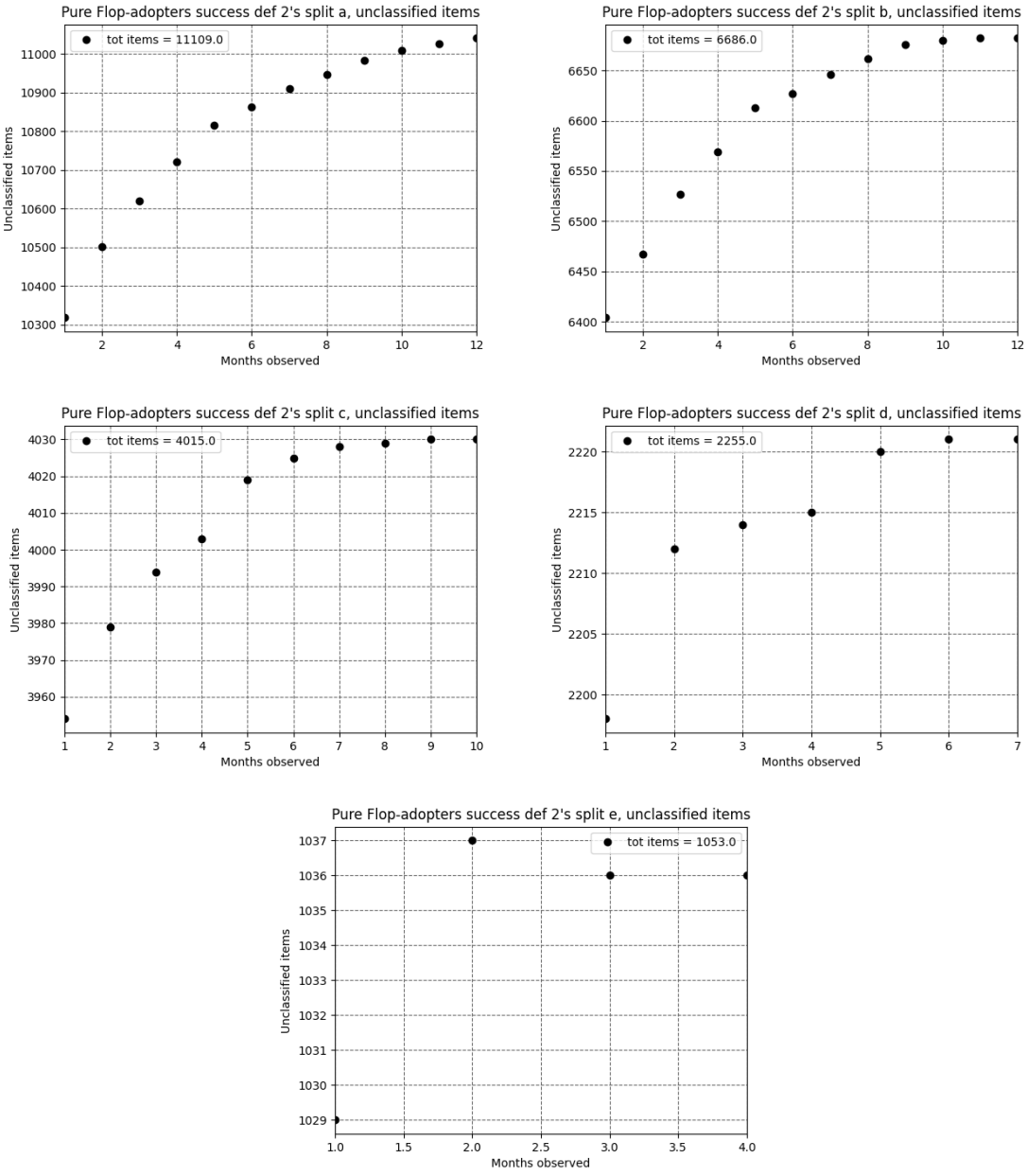

Figure A.23: Unclassified items for the pure Flop-adopters experiment's success definition 2.

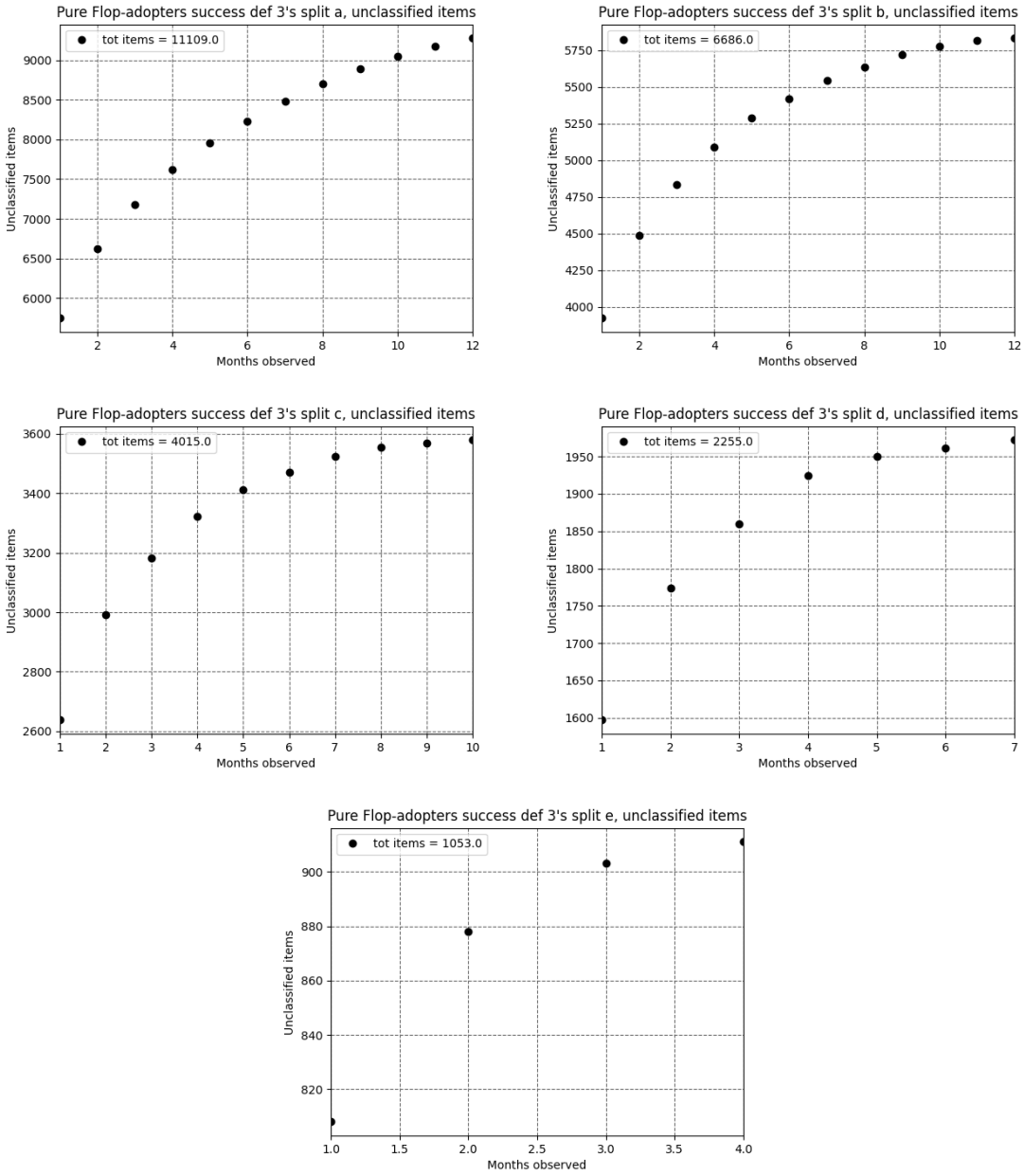

Figure A.24: Unclassified items for the pure Flop-adopters experiment's success definition 3.

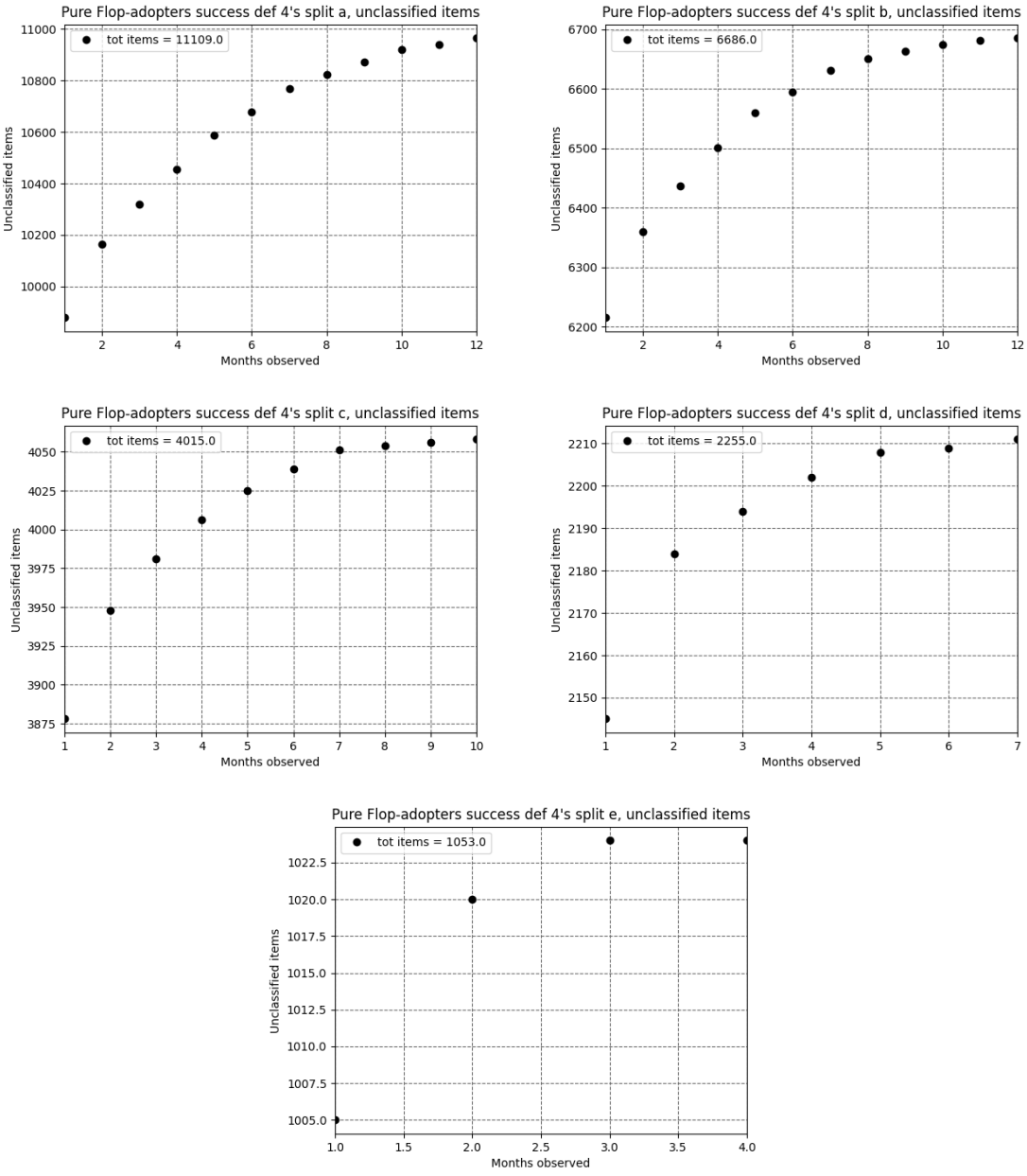

Figure A.25: Unclassified items for the pure Flop-adopters experiment's success definition 4.

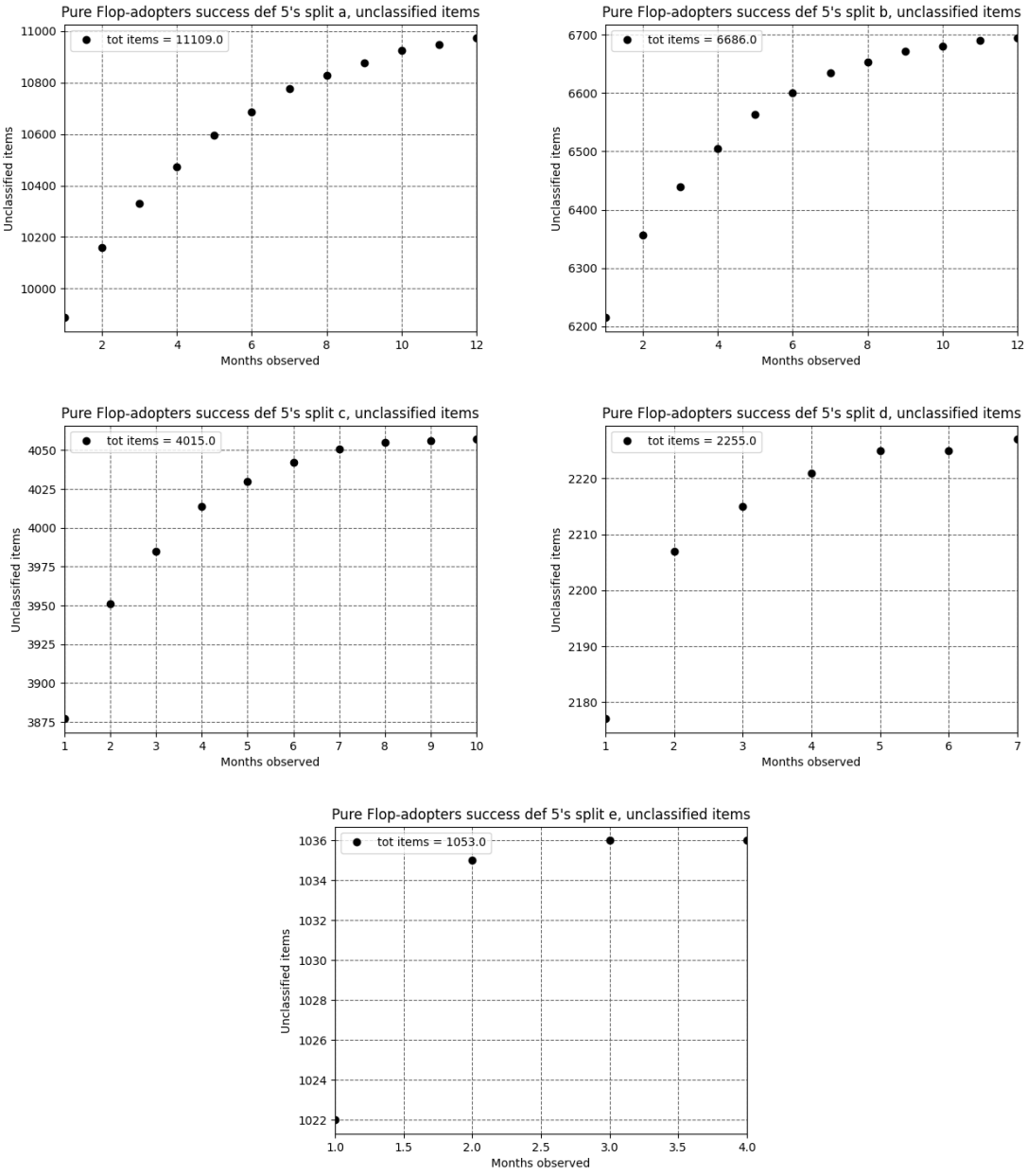

Figure A.26: Unclassified items for the pure Flop-adopters experiment's success definition 5.

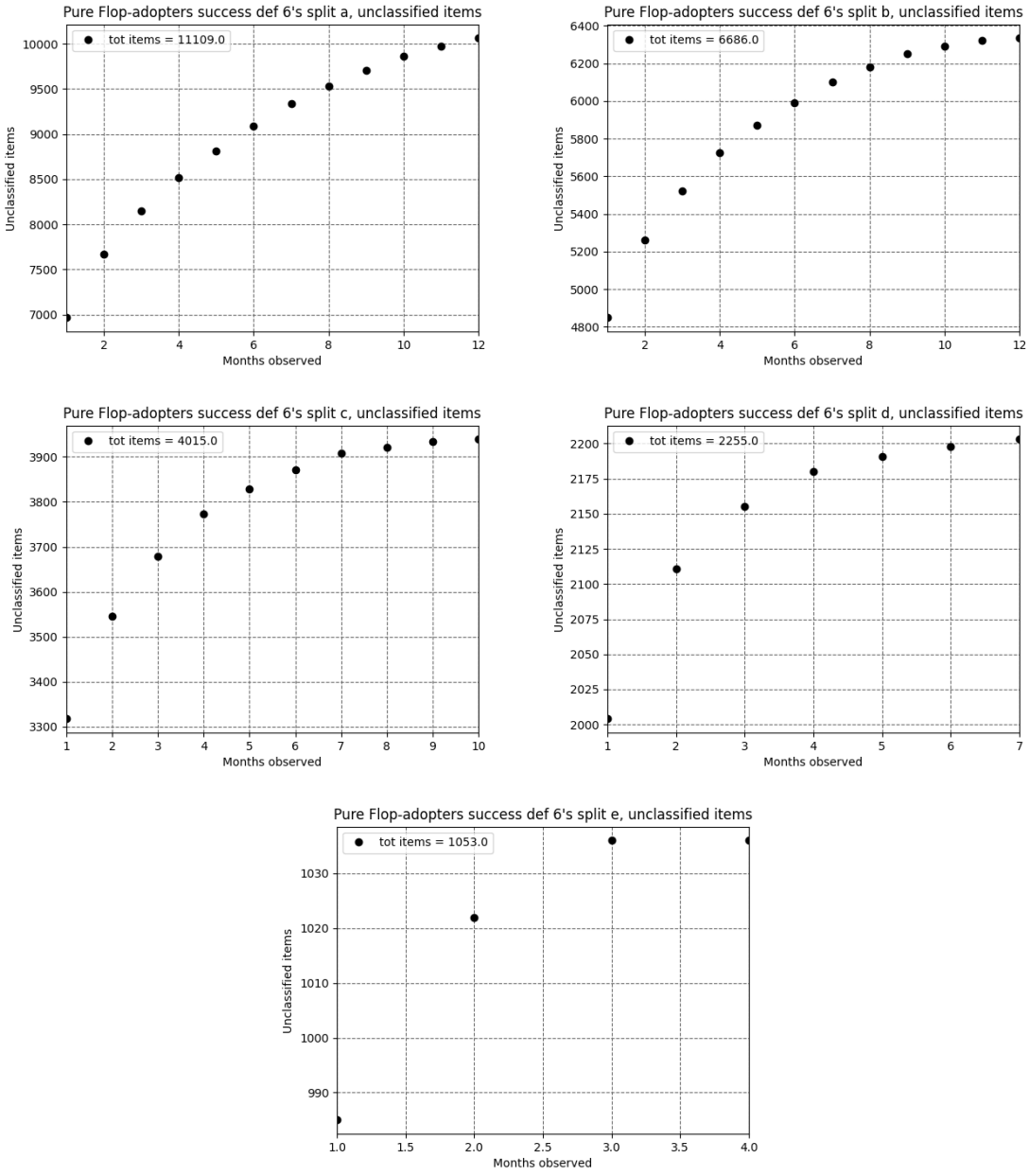

Figure A.27: Unclassified items for the pure Flop-adopters experiment's success definition 6.

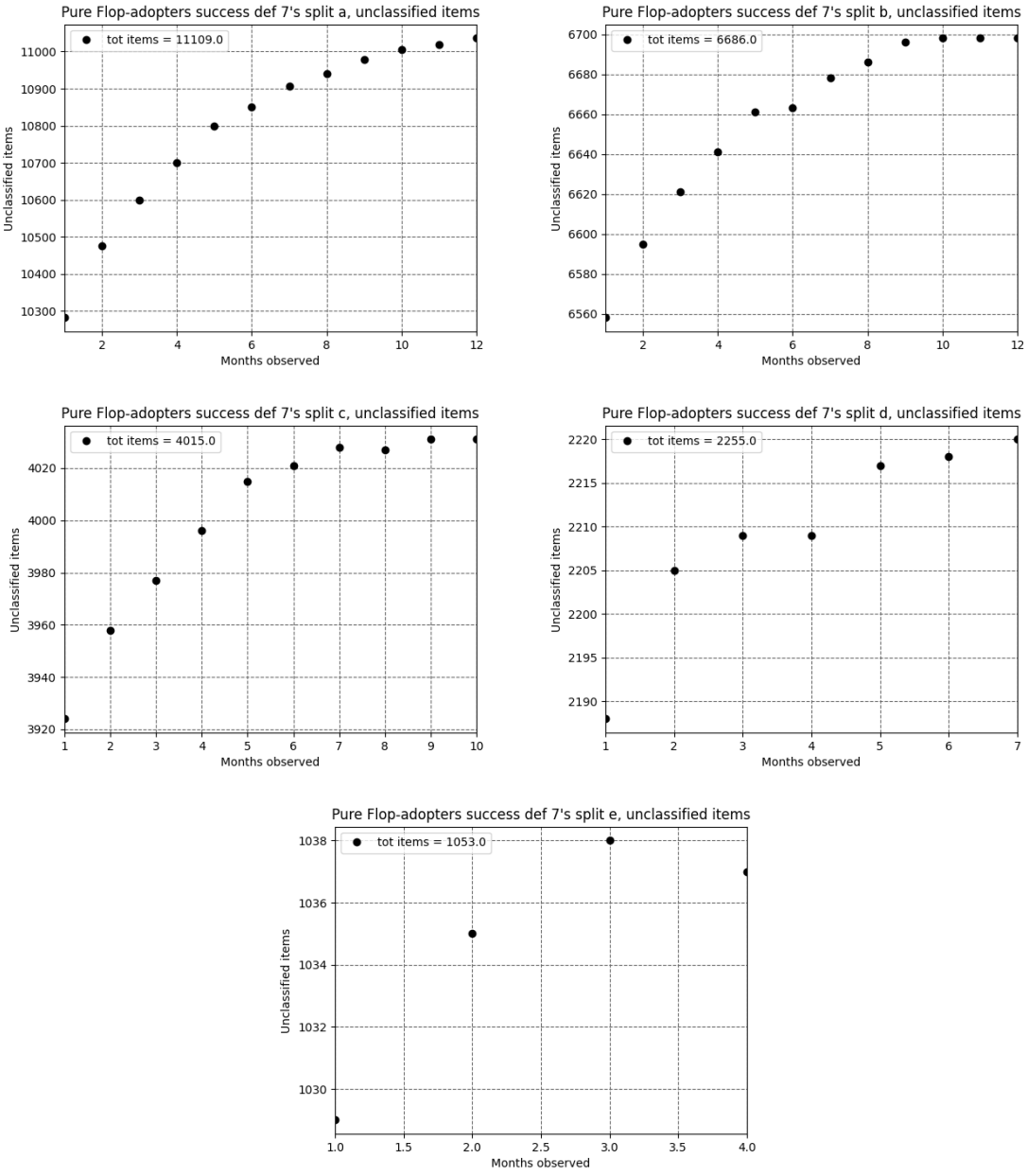

Figure A.28: Unclassified items for the pure Flop-adopters experiment's success definition 7.

Bibliography

[1] Paula Fitzgerald Bone. "Word-of-mouth effects on short-term and long-term product judgments. (English)". In: *Journal of Business Research* (2000). URL: https://www.sciencedirect.com/science/article/pii/014829639400047I?via%3Dihub.

[2] D. Pennacchioli et al. "The Three Dimensions of Social Prominence. (English)". In: *SocInfo: International Conference on Social Informatics"* (2013), pp. 319–332. URL: https://link.springer.com/chapter/10.1007/978-3-319-03260-3_28.

[3] Giulio Rossetti et al. "Forecasting success via early adoptions analysis: A data-driven study. (English)". In: (2017). URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0189096.

[4] Leonard Richardson. *BeautifulSoup*. 2019. URL: https://www.crummy.com/software/BeautifulSoup/.

[5] NetworkX developers. *NetworkX*. 2019. URL: https://networkx.github.io/.

[6] Gábor Csárdi and Tamás Nepusz. *IGraph*. 2019. URL: https://igraph.org/.

[7] G. Rossetti et al. *CDlib*. 2019. URL: https://cdlib.readthedocs.io/en/latest/index.html.

[8] Community library project. *SciPy*. 2019. URL: https://www.scipy.org/.

[9] Community library project. *NumPy*. 2019. URL: https://www.numpy.org/.

[10] Community library project. *Pandas*. 2019. URL: https://www.pandas.org/.

[11] Richard Hipp. *SqlLite*. 2019. URL: https://www.sqlite.org/index.html.

[12] Community library project. *Matplotlib*. 2019. URL: https://www.matplotlib.org/.

[13] Michael Waskom. *Seaborn*. 2019. URL: https://seaborn.pydata.org/.

[14] Gephi Consortium. *Gephi*. 2019. URL: https://gephi.org/.

[15] Institute for System Biology. *Cytoscape*. 2019. URL: https://cytoscape.org/.

[16] Last.fm Ltd. *Last.fm Web Services*. 2019. URL: https://www.last.fm/api/.

[17] The Scipy community. *NumPy 1.13.0*. 2017. URL: https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.histogram.html.

[18] Laura Pollacci, Riccardo Guidotti, and Giulio Rossetti. "Are we playing like MusicStars?" Placing Emerging Artists on the Italian Music Scene. (English)". In: (2016). URL: https://www.researchgate.net/publication/307878345_Are_we_playing_like_Music-Stars_Plac%20ing_Emerging_Artists_on_the_Italian_Music_Scene.

[19] Laura Pollacci et al. "The italian music superdiversity. (English)". In: *Multimed Tools Appl 78* (2018), pp. 3297–3319. URL: https://doi.org/10.1007/s11042-018-6511-6.

[20] Google. *Google trends*. 2006. URL: https://trends.google.coml.

[21] Riccardo Guidotti, Giulio Rossetti, and Dino Pedreschi. "Audio Ergo Sum A Personal Data Model For Musical Preferences. (English)". In: *Conference: 5th International Symposium "From Data to Models and Back" (DataMod)* (2016). URL: https://www.researchgate.net/publication/304396275_Audio_Ergo_Sum_A_Personal_Data_Model_For_Musical_Preferences.

[22] Riccardo Guidotti and Giulio Rossetti. ""Know Thyself" How Personal Music Tastes Shape the Last.FmOnline Social Network. (English)". In: *International Symposium "From Data to Models and Back (DataMod)* (2019). URL: http://www.giuliorossetti.net/about/ongoing-works/publications/.

[23] Giulio Rossetti. "Exorcising the Demon: Angel, Efficient Node-Centric Community Discovery. (English)". In: *Complex Networks and Their Applications VIII* (2019). URL: https://www.researchgate.net/publication/337955436_Exorcising_the_Demon_Angel_Efficient_Node-Centric_Community_Discovery.

[24] Michele Coscia et al. "DEMON: a Local-First Discovery Method for Overlapping Communities. (English)". In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Beijing, China* (2012). URL: https://arxiv.org/abs/1206.0629.

[25] Fosca Giannotti Michele Coscia and Dino Pedreschi. "A classification for community discovery methods in complex networks. (English)". In: *Statistical Analysis and Data Mining* (2011). URL: https://arxiv.org/abs/1206.3552.

[26] L. P. Cordella et al. "A (sub)graph isomorphism algorithm for matching large graphs. Pattern Analysis and Machine Intelligence. (English)". In: *IEEE Transactions* (2004). URL: https://ieeexplore.ieee.org/document/1323804.

[27] Jinsoo Lee et al. "An in-depth comparison of subgraph isomorphism algorithms in graph databases. (English)". In: *Proceedings of the 39th international conference on Very Large Data Bases* (2012), pp. 133–144. URL: https://ieeexplore.ieee.org/document/1323804.

[28] U. et al. Brandes. "On Modularity Clustering. (English)". In: *IEEE Trans. Knowl. Data Eng* (2008). URL: https://ieeexplore.ieee.org/document/4358966.