Constructing Applicants from Loan-Level Data: A Case Study of Mortgage Applications

Hadi Elzayn
Stanford University

Simon Freyaldenhoven
FRB Philadelphia

Minchul Shin
*FRB Philadelphia**

December 3, 2024

Abstract

We develop a clustering-based algorithm to detect loan applicants who submit multiple applications ("cross-applicants") in a loan-level dataset without personal identifiers. A key innovation of our approach is a novel evaluation method that does not require labeled training data, allowing us to optimize the tuning parameters of our machine learning algorithm. By applying this methodology to Home Mortgage Disclosure Act (HMDA) data, we create a unique dataset that consolidates mortgage applications to the individual applicant level across the United States. Our preferred specification identifies cross-applicants with 93% precision.

JEL-Classification: C38, C63, C81, G21, R21
Keywords: clustering, mortgage applications, HMDA

*We thank Andrew Gross, Ryan Kobler, Kellen O'Connor, and Eliana Sena Sarmiento for excellent research assistance. Emails: hselzayn@law.stanford.edu, simon.freyaldenhoven@phil.frb.org, minchul.shin@phil.frb.org.

1 Introduction

The Home Mortgage Disclosure Act (HMDA) data is a leading example of a consumer finance dataset at the account level that does not include person-level identifiers: it contains the near-universe of all mortgage applications in the US, but does not contain an applicant identifier that allows linking multiple applications to the same applicant. We present a novel approach to detect "cross-applicants" – individuals who submit multiple mortgage applications – using a clustering-based algorithm applied to loan-level data.

In a nutshell, our approach works as follows. Using application level mortgage data from a confidential version of the widely known HMDA dataset[1], we first split the data into *partitions*, characterized by the distinct outcomes of nine categorical variables, such as census tract of the property or the race of an applicant. We then apply a clustering algorithm to further break down these partitions into *clusters*. These clusters have the property that all applications within one cluster are "close" in terms of a number of continuous attributes such as application date or reported income. This is motivated by the fact that multiple applications submitted by the same individual for the same property may differ only slightly. For example, an applicant may submit two otherwise identical applications to different lenders on subsequent days.

Figure 1 illustrates our approach. Here, we have created a partition of size four using categorical variables in a first step. In the second step, we then use a single continuous variable, for instance application date, to further split the partition into three clusters. The first two clusters consist of single applications (reflecting two applicants), while the third cluster consists of two applications (reflecting a third applicant). Here, the last two applications are assigned to the same cluster because they are filed within a short time span. At this point, all applications in a given cluster are "near-identical." In both simulations

[1] We discuss our data in more detail in Section 4.

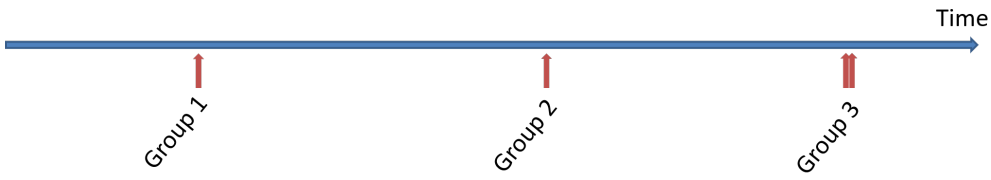


Figure 1: Illustration of clustering algorithm in a hypothetical univariate example for a partition with four applications, where the only clustering variable is "Time." The last two applications are filed within a short time span, and thus assigned to the same cluster.

and our empirical application, we provide evidence that clusters constructed this way indeed mostly represent single individuals: The vast majority of clusters include only applications from the same applicant, and thus represent individuals submitting multiple (near-identical) mortgage applications for the same property.

The contributions of this paper are twofold. First, we develop a methodological framework that allows us to identify and group loan applications submitted by the same individual in large-scale, anonymized datasets. This framework is flexible and can be applied to a variety of datasets where individual-level identifiers are absent. We implement our method using a state-of-the-art agglomerative clustering algorithm, making it feasible to apply this method to large datasets with millions of observations. Crucially, we develop a novel method to measure the quality of the proposed algorithm that does not require labeled training data. To do this, we exploit the fact that consumers can take out only a single first-lien mortgage for a given property. This insight further allows us to select the tuning parameters of our algorithm to optimize its performance. We validate our approach through simulation, demonstrating its effectiveness in identifying cross-applicants in simulated data.

Second, we then apply our method to the Home Mortgage Disclosure Act (HMDA) data, creating a unique dataset that consolidates mortgage applications at the individual applicant level across the country. Our preferred specification of the algorithm demonstrates a high level of precision, successfully identifying cross-applicants with an estimated 93% precision. This level of performance underscores the potential of our approach.

The paper proceeds as follows. Section 2 describes our clustering approach in more detail. We then demonstrate the empirical performance of our approach in simulated data in Section 3. In Section 4, we apply our approach to confidential HMDA and validate its performance. Section 5 concludes.

2 Identifying Cross-Applicants

2.1 Setup

We assume access to a loan-level dataset without personal identifiers. Specifically, we will frame our following discussion around a dataset of mortgage applications, in line with our empirical application. There are N potential borrower-property pairs, which we refer to as "individuals." If the same person applies for a mortgage for two distinct properties, we would count this as two distinct individuals. Each individual is indexed by i where $i \in 1, 2, ..., N$. Individuals submit loan applications. They may submit multiple applications for the same property, $m \in 1, 2, ..., n_i$. The number of applications, n_i, for each individual may be dependent on the individual's characteristics and can be random.

The covariate vector $Z_{im} = [X_{im}, C_i]$ includes variables observable by the econometrician. X_{im} is a vector of variables that may vary across applications if an individual submits multiple applications (e.g., the application date), while C_i is a vector of variables that are constant across an individual's applications (e.g., the census tract for applications involving the same property). The binary indicator L_{im} is set to 1 if individual i's loan application m is accepted, and 0 otherwise. For each approved loan, individuals decide whether to originate or not. Let $O_{im} = 1$ if a loan is originated, and $O_{im} = 0$ otherwise. Further, since we are considering first-lien mortgages throughout, an individual can originate at most one loan.

The econometrician does not observe the individual index i. Instead, an application $j \in 1, 2, ..., J$, where $J \geq N$representing a row in the data, consists of X_j, C_j, L_j, and the

element-wise product $L_j O_j$.

2.2 Methodology

Our cross-applicants are constructed as follows. We first split the data into partitions based on the realizations of the variables in C_j that are assumed to be constant across applications submitted by the individual i (e.g., census tract). We then further break down these partitions into groups based on the variables in X_j that are potentially different across applications submitted by the individual i (e.g., date) but expected to be "close." In particular, we group applications such that, for all applications x_j and $x_{j'}$ in the same group, $d(X_j, x_{j'}) \leq \varepsilon$. Here, $d(\cdot)$ denotes distance, x_j is vector of observed variables for application j of dimension r, and ε is a tuning parameter that determines the maximum distance between two applications in the same group.[2]

Definition 1. *We call a cluster of applications $\mathcal{S}$ ε-identical if $d(x_j, x_{j'}) \leq \varepsilon$ and $c_j = c_{j'}$ for any applications $j, j' \in \mathcal{S}$, where $z_j = [x_j, c_j]$ is a vector of observed variables for application j.*

We then treat applications in the same cluster as if they were submitted by the same individual, i. Our hope is that these clusters indeed represent individual applicants who submitted multiple ε-identical applications.

We first note that finding all ε-identical applications in the data is computationally challenging. One simple strategy is to begin with single applications as their own clusters. Then, we iteratively merge the two closest clusters until only one cluster remains. This process

[2]In our empirical implementation, we use a distance function $d(\cdot)$ of the following form:

$$d(x_j, x_{j'}) = \left(\sum_{s=1}^{r} d_s(x_{sj}, x_{sj'})^2 \right)^{1/2}.$$

Note that this corresponds to a weighted ℓ_2-norm if $d_s(x_{sj}, x_{sj'}) = w_s(x_{sj} - x_{sj'})$, although we also consider more general distances (see Appendix B).

results in an inverse tree structure where applications progressively merge into larger clusters until a single, giant cluster is formed. We then select the clusters where all applications within a given cluster are ε-identical by truncating the inverse tree structure at a specific ε value. All clusters constructed in this manner contain applications j, j' that are identical in terms of their categorical variables ($c_j = c_{j'}$) and near-identical in terms of their continuous variables ($d(x_j, x_{j'}) \leq \varepsilon$). Figure 2 illustrates this process for three applications with identical c_j values (e.g., the same location) but differing loan amounts, denoted as x_j. In this example, as we increase ε, the algorithm will first cluster the first two applications together into one ε-identical cluster. Once ε is larger than 10K, all three applications are clustered into a single cluster.

It is also important to note that once this inverse tree is created, there is no need to recompute clusters (e.g., distances) for different choices of ε, which facilitates the development of a method for selecting the ε value without incurring additional computational costs, which is the topic of the next subsection. The algorithm described above is a version of what is known as agglomerative clustering. Unfortunately, this algorithm has a worst-case time complexity of $\mathcal{O}(\ell^3)$, where ℓ is the size of the largest partition. Instead, we apply a state-of-the-art hierarchical agglomerative clustering algorithm for complete-linkage clustering that is based on the nearest-neighbor chain method. This algorithm has a worst-case complexity of $\mathcal{O}(\ell^2)$ while converging to the same clusters that the original, slower algorithm would produce [Müllner, 2011]. In our simulation and empirical applications, we implement agglomerative clustering algorithm using the `fastcluster` package in Python [Müllner, 2013].

2.3 Useful bounds for tuning parameter selection

In practice, our clustering algorithm may pick up some pairs of applications that are near identical in terms of their observable characteristics, but in fact correspond to multiple individuals. In that case, their unobservable characteristics may differ substantially. We

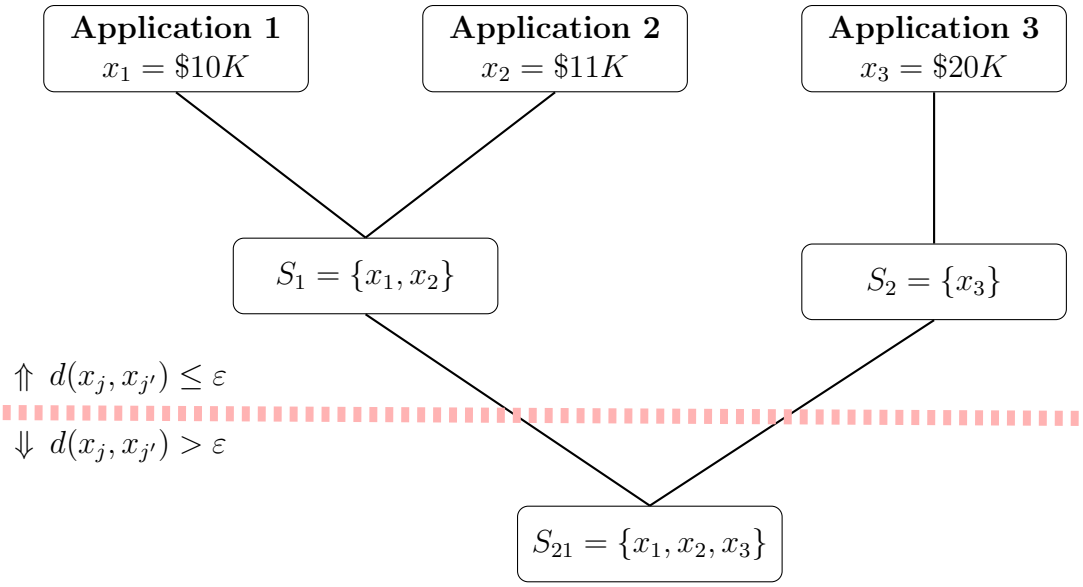


Figure 2: The figure illustrates a hierarchical clustering process for three applications with identical c_j values (e.g., the same location) but differing loan amounts, denoted as x_j. The red dashed line represents the cutoff defined by ε for a hypothetical value of ε between \$1K and \$10K. The clusters $S_1 = \{x_1, x_2\}$ and $S_2 = \{x_3\}$ are ε-identical clusters at this cutoff. If ε is increased beyond \$10K (i.e., the red line is lowered), the applications merge into a single ε-identical cluster $S_{21} = \{x_1, x_2, x_3\}$. This example demonstrates how hierarchical clustering relies on forming an inverse tree structure, after which identifying ε-identical clusters becomes straightforward by simply adjusting the position of the cutoff line.

therefore next propose a way to lower-bound the performance of our algorithm in identifying clusters of applications belonging to the same applicant. In addition, we will later use these bounds to choose our tuning parameters ($d(\cdot)$,ε).

We make the following assumptions:

Assumption 1 (Applications are i.i.d. across borrowers)**.** *For any l, m and $i \neq j$:*

$$\Pr[O_{im} = 1 | O_{jl} = 1] = \Pr[O_{im} = 1] = p.$$

Assumption 2 (Weakly Increasing Origination Probability)**.** *An applicant's origination*

probability is weakly larger if they submit more than one application:

$$Pr[O_{im} = 1 | n_i > 1] \geq \Pr[O_{im} = 1]$$

Let $\texttt{False}$ denote the event that a cluster is a false positive (i.e. contains applications from more than one applicant), and let $\texttt{Mult}$ denote the event that there are multiple originations in a cluster $\mathcal{S}$ (i.e. $\sum_{i,m \in \mathcal{S}} O_{im} > 1$).

Theorem 1. *Under Assumptions 1-2, the false positive rate can be bounded above as follows:*

$$\Pr[\texttt{False}] \leq \frac{\Pr[\texttt{Mult}]}{p^2}.$$

Equivalently, this implies that the precision of our algorithm is at least $1 - \frac{\Pr[\texttt{Mult}]}{p^2}$.

The proof can be found in the Appendix. For intuition, consider a simplified version of the problem where all clusters are at most of size two and applicants submit at most two applications. Recall that no individual can take out two first lien mortgages for the same property: for all i, $\sum_{m=1}^{M} L_{im} O_{im} \leq 1$. If our algorithm works perfectly, and each cluster contains only applications from a single applicant, the probability of seeing two originations in the same cluster is equal to zero, since for all clusters $\mathcal{S}$: $\sum_{i,m \in \mathcal{S}} L_{im} O_{im} \leq 1$. On the other extreme, suppose our clusters contain random pairs of applications. In that case, the probability of seeing two originations in a given cluster can be approximated by $P(O_{im})P(O_{jk}) = P(O_{im})^2$.

Thus, the rate at which we find multiple originations in the same cluster in the data is informative about the quality of our algorithm. In fact, this allows us to not only assess the quality of our clustering algorithm for a given value of tuning parameters ($d(\cdot)$,ε) but also to choose our tuning parameters.

Theorem 1 means that, in order to bound the precision of our algorithm, we need only an

estimate of the unconditional probability of origination p, and an estimate of the probability that our estimated clusters contain multiple originations $\Pr[\texttt{Mult}]$. To estimate the unconditional probability of origination for a single mortgage application p, we simply use the empirical probability of origination in our dataset, $\hat{p}$. Likewise, we simply use the fraction of clusters that have multiple originations, $\hat{p}_m$, as an estimate for $\Pr[\texttt{Mult}]$.

Finally, we note that we can exclude any clusters that indeed contain multiple origination (which we know are false positives) to improve the precision of our algorithm. Our initial clusters contained three types of clusters: 1) True cross-applicants $\hat{\mathcal{S}}^T$, 2) False positives with zero or one originations $\hat{\mathcal{S}}^F_{0-1}$, 3) False positives with multiple originations $\hat{\mathcal{S}}^F_{2+}$. Since we can easily identify the clusters in the third category, we can improve the performance of our algorithm by simply dropping estimated clusters with multiple originations. This yields a new lower bound on the precision of our algorithm equal to

$$\frac{1 - \frac{\Pr[\texttt{Mult}]}{p^2}}{1 - \Pr[\texttt{Mult}]}. \tag{1}$$

Its empirical counterpart is then given by

$$\frac{1 - \frac{\hat{p}_m}{\hat{p}^2}}{1 - \hat{p}_m}. \tag{2}$$

where $\hat{p}$ is the empirical probability of origination, and $\hat{p}_m$ is the empirical fraction of clusters with multiple originations.

3 Simulation

We next generate a hypothetical dataset to illustrate our algorithm and demonstrate its performance in a stylized setting. We first create one million "census tracts." For each census tract c, the number of applicants belonging to this census tract N_c is equal to $1 + \psi_c$, where

ψ_c drawn from a Poisson distribution with parameter $\lambda = 1$ to approximate the distribution of partitions we observe in our empirical application. Next, an applicant i submits a loan application. After each application, she continues to submit another application with probability 0.2 such that the expected number of applications per applicant n_i is 1.25.

We then create features associated with each application (in addition to the census tract C_i) as follows. First, to create a second variable that is constant across an individual's applications we randomly assign a group membership $G_i \in \{0, 1\}$ to applicant i with $Pr(G_i = 1) = \gamma_0 \in (0, 1)$, where G_i is independent across i. The variable G_i may represent characteristics such as race or gender. Next, each application m by applicant i is associated with three further covariates X_{im}, T_{im} and η_i. We assume that both X_{im} and T_{im} are observed by the econometrician, while η_i is not. We stress that X_{im} and T_{im} may differ (slightly) across applications m to reflect the observed data. We create realizations of these random variables as follows. T_{im} may reflect the time of the application, and is equal to $T_{im} = \tilde{T}_i + \nu_{im}$, where $\tilde{T}_i \sim Unif[0, 1]$ and $\nu_{im} \sim N(0, \sigma_T)$. X_{im} may reflect the loan amount, and is equal to $X_{im} = \tilde{X}_i + \xi_{im}$, where $\xi_{im} \sim N(0, \sigma_X)$. The conditional distribution of $(\tilde{X}_i, \eta_i)$ conditional on G_i is given by

$$[\tilde{X}_i, \eta_i]' \sim Lognormal(\mu_g, \Sigma_g), \tag{3}$$

where μ_g and Σ_g are mean and covariance matrix of bi-variate normal for $G_i = g$. Specifically, we use $\gamma_0 = 0.5$, $\mu_0 = [-3, -3]$, $\mu_1 = [-2.5, -2.5]$, $\Sigma_0 = \Sigma_1 = [0.25\ 0.1, 0.1\ 0.25]$.

Finally, each applicant has a default behavior associated with her. We assume that the default probability of an applicant, $Pr(D_i = 1)$, depends on both $\tilde{X}_i$ and η_i as follows:

$$Pr(D_i = 1|\tilde{X}_i, \eta_i, G_i, \tilde{T}_i, C_i) = \min(1, \delta_0 + \delta_1 \tilde{X}_i + \delta_2 \eta_i). \tag{4}$$

Potential lenders observe (or are able to estimate) an individual's default probability $P(D_i)$,

and their decision whether to extend the loan takes the form:

$$P(L_{im}=1) = \begin{cases} 1 & \text{if } P(D_i=1) < 0.27 \\ 0.5 & \text{if } P(D_i=1) \in [0.27, 0.29] \\ 0 & \text{if } P(D_i=1) > 0.29. \end{cases} \tag{5}$$

An applicant hears back sequentially from her applications. As long as she has not originated a loan, each time an application is approved the applicant originates the corresponding loan with probability 0.9. Once she originates her first loan, the applicant does not originate any additional loans.

We reemphasize that the econometrician observes application level data without knowing the index i.[3] That is, she does not know whether two applications j and j' are submitted by the same individual i. For each application j, the variables G_j, C_j, X_j, T_j, L_j as well as the element-wise products D_jL_j and O_jL_j are observable by the econometrician. On the other hand, $\tilde{X}_i$, $\tilde{T}_i$, and η_i are unobserved to the econometrician.

3.1 Results

We first depict a histogram of the number of observed applications per census tract c in Figure 3.

Since within each census tract, applications coming from different applicants may further have different values of G_j, the majority of partitions we create based on C_j and G_j are small, in line with our empirical application.

We then run our algorithm to identify cross-applicants - applicants that submit multiple applications, using the clustering approach outlined above. For the simulation exercise, we use simple Euclidean distance, $d(x_j, x_k) = \|x_j - x_k\|_2$, as our distance function between two

[3]However, while this is infeasible in practice, knowing the index i in our simulated dataset will allow us to evaluate the performance of our algorithm.

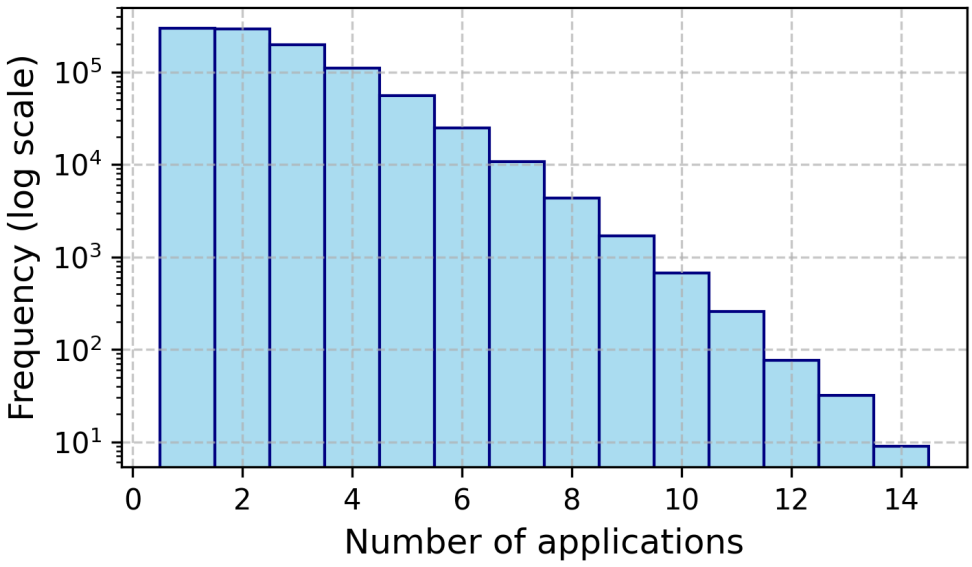


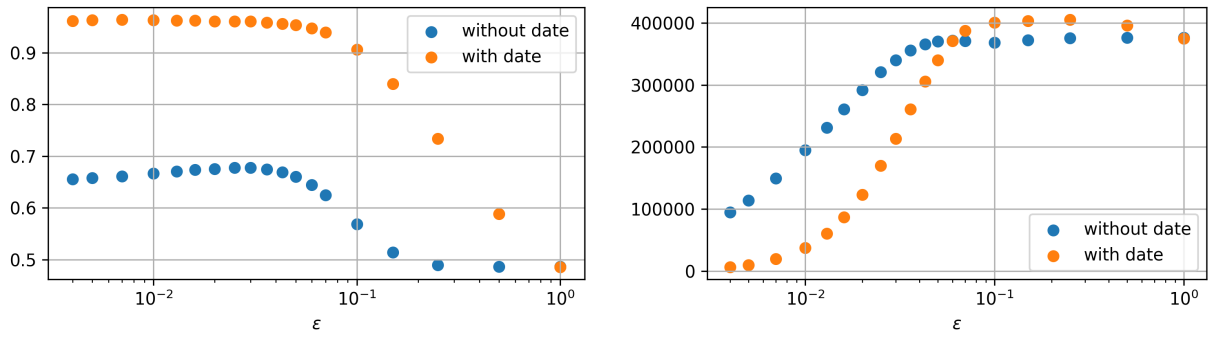
Figure 3: Number of observed applications per census tract c in our simulated data.

applications x_j, x_k when computing our clusters. To demonstrate how additional observed covariates impact our results, we run our algorithm two times. In the first specification ("without date"), we use only a single continuous variable in X_j during the clustering step, withholding the second observed covariate T_j. In the second specification ("with date"), we use both X_j and T_j during the clustering step.[4]

We first illustrate the performance of our algorithm as a function of the tuning parameter ε, where $d(x_j, x_k) < \varepsilon$ if applications j and k are grouped together in the same cluster $\mathcal{S}$. Figure 4a depicts the fraction of clusters that contain only applications from a single applicant. This represents the precision (= True Positives/(True Positives + False Positives)) of our algorithm, where a positive instance corresponds to a true cross-applicant.[5]

[4]Alternatively, we note that the two specifications can be viewed as using two different weight vectors on the covariate vector. That is, with $x_j = [X_j, T_j]$, and using a weighted ℓ_2-norm of the form $d_w(x_i, x_j) = \left(\sum_{k=1}^{r} w_k (x_{ki} - x_{kj})^2\right)^{1/2}$ as the distance metric between applications, our two specifications ("with date" and "without date") correspond to weight vectors of $w = (1, 1)$ and $w = (1, 0)$, respectively. We will revisit this interpretation in Section 4.

[5]To keep things as simple as possible, we drop all clusters with more than two applications in both our simulation results and our application that follows, such that all results are based on clusters with two applications.



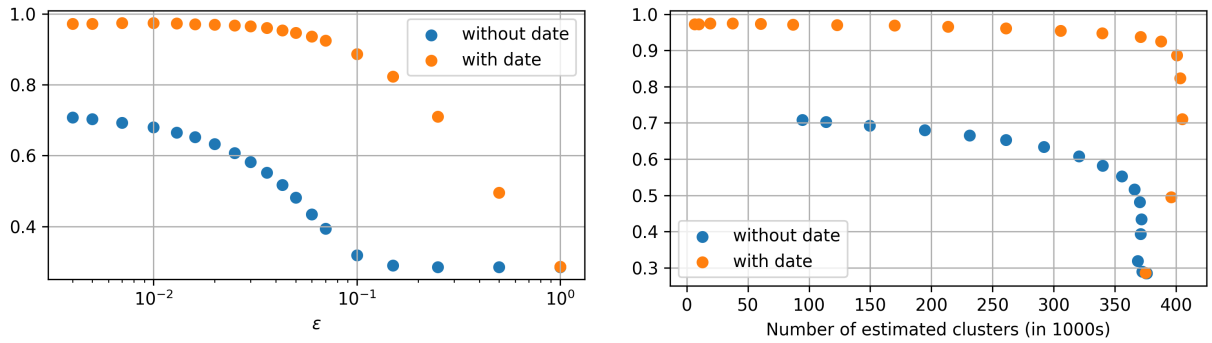
(a) Fraction of clusters that consist of applications from a single applicant.

(b) Number of estimated clusters.

Figure 4: Estimated cross-applicants as function of tuning parameter ε. Raw cluster estimates are adjusted by dropping clusters with multiple originations.

We first note that the availability of an additional covariate greatly improves the performance of the algorithm: Without the date variable T_j, the precision of our algorithm is below 70% for all values of ε, while including the date variable can lead to a precision above 95%. Next, we note that the quality of our algorithm increases as we reduce the size of ε. Intuitively, as we require applications within a cluster to be closer to identical, we reduce the number of "false positives" - applications that look similar but are submitted by distinct applicants. On the other hand, Figure 4b illustrates how the number of estimated cross-applicants increases with ε. We thus face a trade-off - while small values of ε tend to lead to fewer false positives, we might want to keep ε large enough to obtain a substantial sample size. In both our specifications here, we observe a sweet spot slightly below $\varepsilon = 0.1$. For example, with $\varepsilon = 0.06$ our algorithm finds more than 370,000 clusters that contain multiple applications in both specifications, and around 95% ("with date") and 64% ("without date") of these clusters contain only applications from a single applicant.

Since the construction of Figure 4a requires knowledge of an individual's identifier i, it is infeasible in practice and cannot be used to assess the quality of the estimates or to choose the value of the tuning parameter ε. However, as we argued in the previous section, the rate



(a) As a function of ε

(b) As a function of sample size

Figure 5: Implied fraction of clusters that consists of true cross-applicants (precision).

at which we find multiple originations in the same cluster is informative about the quality of our algorithm. The probability of origination for a given approved application is $\hat{p} = 0.7917$, and the probability of a random pair of applications having both loans originated is thus equal to $\hat{p}^2 = 0.6269$. We therefore use 0.6269 as an estimate for the probability that a cluster containing applications from multiple applicants has two originations. Using Equation (2), we can then translate $\hat{p}$ and $\hat{p}_m$, the empirical fraction of clusters with multiple originations for a given choice of tuning parameters, into an estimate for the fraction of clusters that consists of applications from the same applicant.

This is depicted in Figure 5a, which shows the estimated fraction of clusters that consists of applications from the same applicant using Equation (2). We first note the close resemblance between Figures 4a and 5a. We take this as an encouraging sign that our proposed method to assess the performance of our algorithm works well. In our main specification ("with date"), we obtain a feasible estimate for a lower bound of how many clusters contain applicants from a single applicant equal to 93.7% at $\varepsilon = 0.06$.

Finally, we directly depict the trade-off between the estimated precision of our algorithm and the number of estimated cross-applicants in Figure 5b. This allows for an easy illustration of the relevant trade-off across the different tuning parameters (i.e. the tolerance ε and distance function $d(\cdot)$). For example, when considering two sets of tuning parameters

$(\varepsilon_1, d_1(\cdot))$ and $(\varepsilon_2, d_2(\cdot))$, we strictly prefer $(\varepsilon_1, d_1(\cdot))$ to $(\varepsilon_2, d_2(\cdot))$ if it results in both higher implied precision and a larger sample size.

4 Application to the US Mortgage Market

4.1 Data

We obtain data on mortgage applications from the Home Mortgage Disclosure Act (HMDA). The vast majority of all mortgage applications filed in the US are subject to HMDA reporting and thus included in this dataset. While a publicly available version of this dataset exists, we work directly with a confidential version (cHMDA) that is available to users within the Federal Reserve System and includes more detailed information for each loan application (e.g. the exact date an application was filed).

We restrict our analysis to mortgage applications filed between 2018 and 2023. We exclude applications from earlier years because a number of important borrower and loan characteristics, such as the credit score and the loan-to-value ratio (LTV) are available only starting in 2018. We further retain only first-lien mortgages and applications that are either approved or denied, dropping applications that are withdrawn by the applicant before a decision was made, applications closed for incompleteness, loan purchases, and applications that went through only the pre-approval process. Finally, we drop applications filed outside the 50 states and Washington D.C.

4.2 Identifying cross-applicants

Around 22% of Americans apply for more than one mortgage during the mortgage application process [Consumer Financial Protection Bureau and Federal Housing Finance Agency,

2024][6]. Since the HMDA dataset is at the application level, and not the individual level, it is generally not possible to identify applicants that submit multiple applications. We apply our proposed method to identify these "cross-applicants" who applied for several loans during the mortgage application process.

Following our earlier discussion, we first split the data into partitions based on categorical characteristics we expect to be constant at the individual level. These are: census tract, property type, occupancy, loan purpose, applicant race, applicant sex, applicant age, loan type and a flag for whether or not there is a co-applicant.[7] Figure 6 shows the distribution of partition sizes for our sample. We note that 99% of the partitions contain five or less applicants in our data. This is intuitive: Each partition includes only applications for mortgages in a small geographic region (census tract), and among those we further separate applications by eight additional discrete attributes of applicant and property.

Thus, all applications in a given partition are similar in a number of dimensions, including the location of the property. However, they may include applications that are filed far apart in time, and they may include applications that differ substantially in their reported income and loan amounts. We therefore next apply the above-mentioned hierarchical agglomerative clustering algorithm to further break down the partitions into clusters. We define our clusters such that, for all applications $x_j, x_{j'}$ in the same cluster the following holds:

$$d(x_j, x_{j'}) \leq \left(\sum_{s=1}^{r} d_s(x_{sk}, x_{sj})^2 \right)^{1/2} \leq \varepsilon, \tag{6}$$

[6]This number is based on the National Survey of Mortgage Originations, conducted by the Consumer Financial Protection Bureau. This is a quarterly representative survey of individuals with newly originated mortgages that asks questions about the entire process of obtaining a mortgage.

[7]Note that this implies that applicants who apply for different types of loans cannot end up in the same cluster. This approach makes sense when we want to investigate whether and how a lender distinguishes between two (almost) identical applications submitted by the same individual. In a different context, it may be more appropriate to track an applicant who applies for different types of loans for the same property. Hence, this is an application-specific modeling choice, and we suggest carefully selecting the variables in the distance function to ensure that the identified cross-applicants are consistent with the research question.

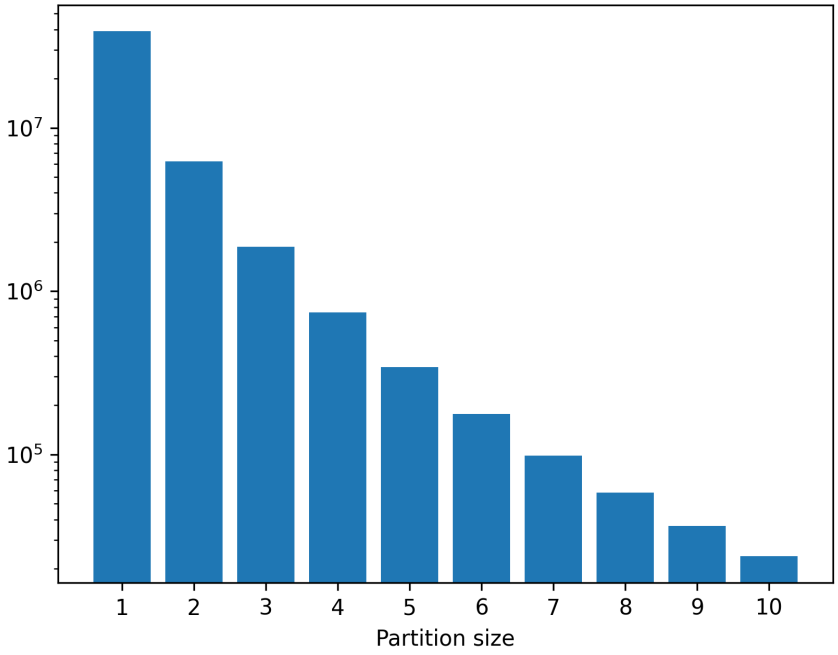


Figure 6: Distribution of partition sizes (number of applications per partition). Partitions larger than ten applications, which constitute 0.13% of the total, are omitted for clarity. The largest partition in the dataset contains 153 applications.

where x_j is a vector of observed variables for application j. Specifically, we use $x_j = (date_j, inc_j, size_j, fico_j, ltv_j)$, where $date_j$ represents the date an application is filed, inc_j is the reported income in thousands of dollars, $size_j$ is the requested loan amount in thousands of dollars, $fico_j$ denotes the reported credit score at the time of application, and ltv_j is the loan-to-value ratio of the loan. Note that this corresponds to a weighted ℓ_2-norm if $d_s(x_{sj}, x_{sj'}) = w_s(x_{sj} - x_{sj'})$, although we also consider more general distances (see Appendix B).

We consider a total of 96 combinations of distance functions $d(\cdot)$ and tolerance parameters ε, and select the best combination based on the accuracy-size trade-off discussed in the previous section and illustrated in Figure 7. In particular, Figure 7 depicts the precision of our algorithm relative to the sample size for the points $(d(\cdot), \varepsilon)$ on the "frontier": the

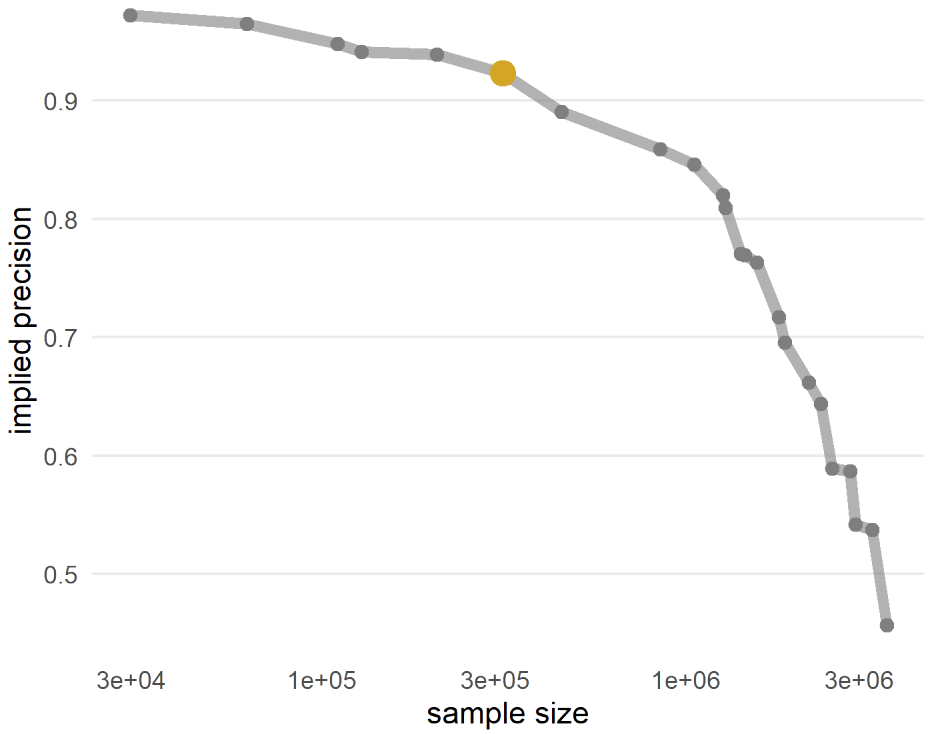


Figure 7: The precision-sample size frontier for identifying cross-applicants. Each point on the curve represents a specific combination of distance function and tolerance parameter. The large orange dot corresponds to our preferred specification, balancing precision and sample size.

combinations of $d(\cdot)$ and ε that are not dominated by an alternative combination yielding both higher precision and a larger sample size. Each point on the curve represents a specific combination of distance function and tolerance parameter. We highlight our preferred specification as the larger orange dot. At this specification, we obtain $314,344$ clusters, and estimate that 92.3% of our estimated clusters are true cross-applicants.

To validate our estimated cross-applicants, Figure 8 depicts the distribution of the difference in dates between applications within a cluster. We depict this distribution separately for those cross-applicants that have their first application denied and those that have their first application approved. We see that most applications within the same cluster are submitted within 3 weeks, and observe a bunching of the differences at multiples of seven, corresponding

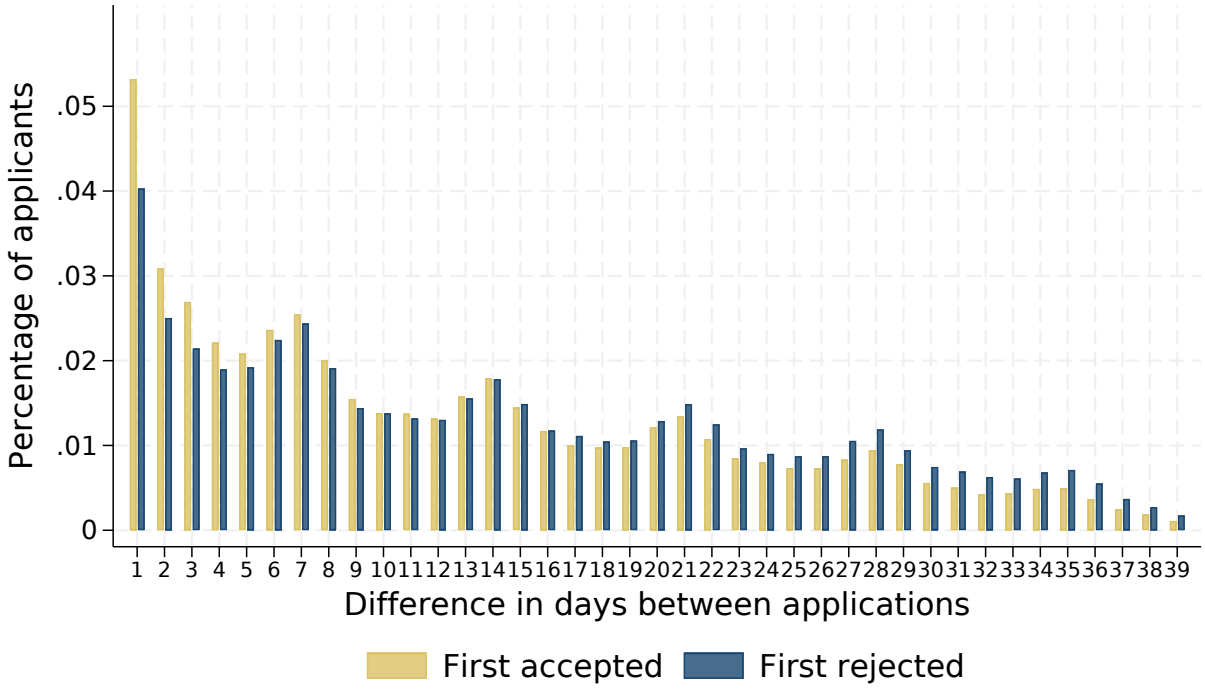
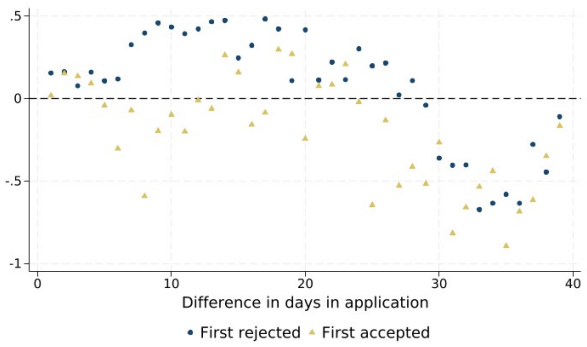


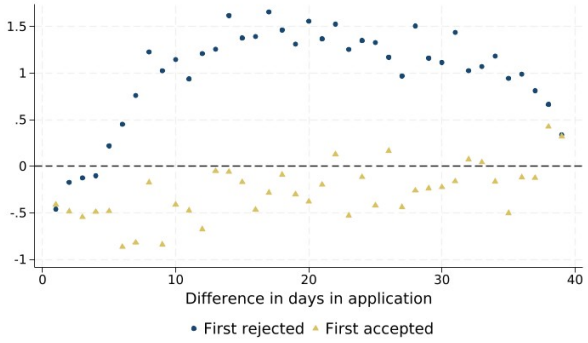
Figure 8: Histogram of date differences between the first and second application submitted by the same applicant. All estimated cross-applicants submitted both applications within 39 days.

to applications being submitted on the same weekday. Further, cross-applicants who first get denied take slightly longer to submit their second application, compared to cross-applicants who have their first application approved.

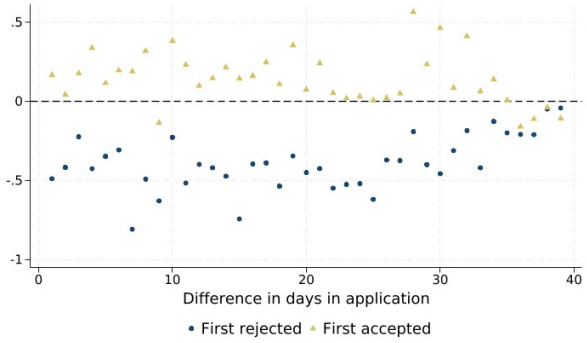
This suggests that applicants who had their first application denied may try to improve their profile before applying again. In Figure 9, we therefore explore how some of the key variables that determine a loan decision vary between the two applications an applicant submits. Figure 9a shows that, applicants whose first application is rejected tend to have a slightly higher credit score on their second application, compared with applicants whose first application is approved, when they submit their second application within four weeks of their first application. After four weeks, all cross-applicants experience a drop in their credit score. This is consistent with a hard credit inquiry as a result of their first application negatively impacting their credit score.



(a) Credit score



(b) Income (yearly, in 1,000s)



(c) Loan amount (in 1,000s)

Figure 9: Average change in the reported variable in an applicant's second application relative to her first application, as a function of the date differences between the first and second application submitted. Yellow triangles represent cross-applicants whose first application was approved. Blue dots represent cross-applicants whose first application was denied.

Figure 9b shows that applicants whose first application is rejected tend to report a higher income on their second application, compared with applicants whose first application is approved. This is consistent with denied applicants taking steps to (marginally) improve their profile if their first application is rejected. In particular, we note that income is generally self-reported (though subject to verification), making it potentially easier to alter compared to, for instance, one's credit score.

Figure 9c shows that applicants whose first application is rejected tend to request a slightly lower loan amount on their second application, compared with applicants whose first application is approved. This is again consistent with denied applicants taking steps to

(marginally) improve their profile if their first application is rejected.

5 Conclusion

In this paper, we presented a novel clustering-based algorithm designed to detect cross-applicants in large anonymized datasets, such as loan-level mortgage data. By applying this methodology to the Home Mortgage Disclosure Act (HMDA) dataset, we successfully identified individuals submitting multiple mortgage applications, achieving an estimated precision of 93%. Our approach introduces a new evaluation method that optimizes the trade-off between precision and efficiency without the need for labeled training data, making our proposed method highly practical. Our results open several promising directions for future research. We conclude by highlighting three potential applications of our work:

1. Measuring Fairness: Cross-applicants may be useful for measuring fairness across demographic groups in the mortgage market. In a companion paper, Elzayn et al. [2024] argue that cross-applicants who had one application approved and a second (near-identical) application rejected can be thought of as "marginal applicants." Comparing the subsequent default probabilities of these marginal applicants may then reveal discrimination in the lenders' loan granting decisions.

2. Monitoring Credit Conditions and Comparing Banks/Lenders: Identifying marginal borrowers may provide a way to monitor current credit conditions. The marginal borrowers as constructed above, in essence, are those borrowers who are on the lender's decision boundary. By examining the characteristics of these borrowers' applications, we can quantify how strict or lenient a lender is. By aggregating lenders' strictness measures, we can further monitor how lending conditions vary over time or across different regions. Additionally, this measure allows for comparisons across banks and lenders, allowing us to explore the factors that determine a bank's level of strictness.

3. Exploring Mortgage Shopping Behavior: Access to a dataset of cross-applicants also provides an opportunity to study the shopping behavior of mortgage applicants. By analyzing the patterns and timing of multiple applications, we can gain a better understanding of how borrowers compare lenders, and how these behaviors differ across various demographic or economic groups.

References

Consumer Financial Protection Bureau and Federal Housing Finance Agency. National survey of mortgage originations, 2024. URL `https://www.consumerfinance.gov/data-research/national-survey-mortgage-originations/`. Accessed: September 11, 2024.

Hadi Elzayn, Simon Freyaldenhoven, Ryan C. Kobler, and Minchul Shin. Measuring fairness in the U.S. mortgage market. *Working Paper*, 2024.

Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

Daniel Müllner. fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python. *Journal of Statistical Software*, 53:1–18, 2013.

A Mathematical Appendix

In this section, we derive how we can use the observed rate of multiple originations per cluster to bound the rate at which applications from distinct individuals are incorrectly clustered together in more detail. Let K be the number of distinct applicants in a cluster. Before the main result, we start with the following useful lemma.

Lemma 1. *The probability of multiple originations in a false cluster is bounded below by p^2. That is:*

$$\Pr[\texttt{Mult}|\texttt{False}] \geq \Pr[O_{im} = 1]^2 = p^2$$

Proof. By definition, $K \geq 2$ for false clusters and thus:

$$\begin{aligned}\Pr[\texttt{Mult}|\texttt{False}] &= \Pr[K = 2|\texttt{False}] \Pr[\texttt{Mult}|K = 2] + \Pr[K > 2|\texttt{False}] \Pr[\texttt{Mult}|K > 2] \\ &= \Pr[K = 2|\texttt{False}] \Pr[\texttt{Mult}|K = 2] + (1 - \Pr[K = 2|\texttt{False}]) \Pr[\texttt{Mult}|K > 2].\end{aligned}$$

We can further separate the clusters with two distinct applicants into those that contain exactly two applications, and those that contain more than two applications, and thus write $\Pr[\texttt{Mult}|K = 2] = w_2 \Pr[\texttt{Mult}|K = 2, |\mathcal{S}| = 2] + (1 - w_2) \Pr[\texttt{Mult}|K = 2, |\mathcal{S}| > 2]$ where $w_2 = \Pr[|\mathcal{S}| = 2|K = 2]$. By Assumption 1, $\Pr[\texttt{Mult}|K = 2, |\mathcal{S}| = 2] = p^2$. Since further, by Assumption 2, $\Pr[\texttt{Mult}|K = 2, |\mathcal{S}| > 2] \geq \Pr[\texttt{Mult}|K = 2, |\mathcal{S}| = 2]$, it follows that $\Pr[\texttt{Mult}|K = 2] \geq p^2$.

In fact, we can use the same argument to separate the clusters with $K > 2$ distinct applicants into those that contain exactly K applications, and those that contain more than K applications. Then, $\Pr[\texttt{Mult}|K = k] = w_k \Pr[\texttt{Mult}|K = k, |\mathcal{S}| = k] + (1 - w_k) \Pr[\texttt{Mult}|K = k, |\mathcal{S}| > k]$. By Assumption 1, $\Pr[\texttt{Mult}|K = k, |\mathcal{S}| = k] = 1 - \Pr[\neg\texttt{Mult}|K = k, |\mathcal{S}| = k]$.

This probability is given by

$$\begin{aligned}\Pr[\neg\texttt{Mult}|K=k,|\mathcal{S}|=k] :&= g(p,k) = (1-p)^k + \binom{k}{1}p(1-p)^{k-1} \\ &= (1-p)^k + kp(1-p)^{k-1} \\ &= (1-p)^{k-1}(1+p(k-1)) \\ &= (1-p)^{k-1}(1-p+kp).\end{aligned}$$

Note that for any fixed $p \in [0,1]$,

$$\frac{\partial g}{\partial k} \leq 0$$

for all k; to see this, we calculate that:

$$\frac{\partial g}{\partial k} = (1-p)^{k-1}\left[((k-1)\,p+1)\ln(1-p)+p\right].$$

Then notice that the inner term is 0 at $p=0$ and decreases with p, because

$$\frac{\partial}{\partial p}\left[((k-1)p+1)\ln(1-p)+p\right] = (k-1)\ln(1-p) - \frac{pk}{1-p},$$

which is non-positive for any $k \geq 1$, $p \in [0,1]$. Hence $\frac{\partial g}{\partial k}$ is product of a non-negative and non-positive term, i.e. is non-positive overall. In other words, $\Pr[\neg\texttt{Mult}|K=k,|\mathcal{S}|=k]$ is decreasing in k. Since $\Pr[\texttt{Mult}|K=k,|\mathcal{S}|=k] = 1-\Pr[\neg\texttt{Mult}|K=k,|\mathcal{S}|=k]$, we must have that $\Pr[\texttt{Mult}|K=k,|\mathcal{S}|=k]$ is increasing in k, and $\Pr[\texttt{Mult}|K=k,|\mathcal{S}|=k] \geq \Pr[\texttt{Mult}|K=2,|\mathcal{S}|=2] = p^2$.

Since, by Assumption 2, $\Pr[\texttt{Mult}|K=k,|\mathcal{S}|>k] \geq \Pr[\texttt{Mult}|K=k,|\mathcal{S}|=k]$, it also

follows that $\Pr[\texttt{Mult}|K = k] \geq p^2$. Putting it together, we obtain that

$$\begin{aligned}\Pr[\texttt{Mult}|\texttt{False}] &= \Pr[K = 2|\texttt{False}] \Pr[\texttt{Mult}|K = 2] + (1 - \Pr[K = 2|\texttt{False}]) \Pr[\texttt{Mult}|K > 2] \\ &\geq \Pr[K = 2|\texttt{False}]p^2 + (1 - \Pr[K = 2|\texttt{False}])p^2 = p^2.\end{aligned}$$

□

The proof of Theorem 1 then follows:

Proof of Theorem 1. We can write:

$$\Pr[\texttt{Mult}] = \Pr[\texttt{False}] \Pr[\texttt{Mult}|\texttt{False}] + \Pr[\neg\texttt{False}] \Pr[\texttt{Mult}|\neg\texttt{False}]$$

But $\Pr[\texttt{Mult}|\neg\texttt{False}] = 0$ because we consider first-lien mortgages and therefore an individual can originate at most one loan. We can thus write that:

$$\Pr[\texttt{False}] = \frac{\Pr[\texttt{Mult}]}{\Pr[\texttt{Mult}|\texttt{False}]} \leq \frac{\Pr[\texttt{Mult}]}{p^2},$$

where the inequality follows from Lemma 1. It also immediately follows that the precision of our algorithm is given by

$$\text{precision} = \Pr[\neg\texttt{False}] = 1 - \Pr[\texttt{False}] \geq 1 - \frac{\Pr[\texttt{Mult}]}{p^2}.$$

□

B Implementation Details

Recall that we use our agglomerative clustering algorithm to break down the partitions of the data into groups such that for all applications x_j and $x_{j'}$ in the same group, $d(x_j, x_{j'}) \leq \varepsilon$. We use a distance function of the following form:

$$d(x_j, x_{j'}) = \left(\sum_{s=1}^{r} d_s(x_{sj}, x_{sj'})^2 \right)^{1/2}.$$

Table 1 summarizes the different ways we compute the distance between applications. Each row corresponds to a specific definition of the distances $d_s(\cdot)$ for $s = 1, \ldots r$. If d_s is numeric, $d_s(x_{sj}, x_{sj'}) = w_s(x_{sj} - x_{sj'})$ with the number in Table 1 indicating the value of w_s. Thus, if the entire weight vector is numeric, the corresponding row represents a weighted ℓ_2-norm. If w_s is equal to "Penalize exact",

$$d_s(x_{sj}, x_{sj'}) = \begin{cases} (x_{sj} - x_{sj'}) & \text{if } x_{sj} \neq x_{sj'} \\ 55 & \text{if } x_{sj} = x_{sj'}. \end{cases}$$

If w_s is equal to "Reward exact",

$$d_s(x_{sj}, x_{sj'}) = \begin{cases} 0 & \text{if } |x_{sj} - x_{sj'}| < 7 \\ 2(|x_{sj} - x_{sj'}| - 7) & \text{otherwise.} \end{cases}$$

For each row, we then use $\varepsilon \in \{15, 22, 30, 40, 52, 70, 90, 110\}$ for a total of 96 combinations of $(d(\cdot), \varepsilon)$. Figure 10 shows the precision and sample size for each of these combinations.

	Application Date	Income	Loan Amount	Credit Score	Property Value
1.	1	0	0	0	1
2.	1	0	1	0	0
3.	1	1	0	0	0
4.	1	1	1	0	1
5.	1	1	1	1	1
6.	1	1	1	2	1
7.	1	1	1	3	2
8.	Penalize Exact	Penalize Exact	1	1	1
9.	Penalize Exact	Penalize Exact	Penalize Exact	1	1
10.	Reward Exact	1	1	1	1
11.	Reward Exact	1	1	2	1
12.	Reward Exact	1	1	3	2

Table 1: Universe of distance functions considered. Each line corresponds to one definition of distance between applications.

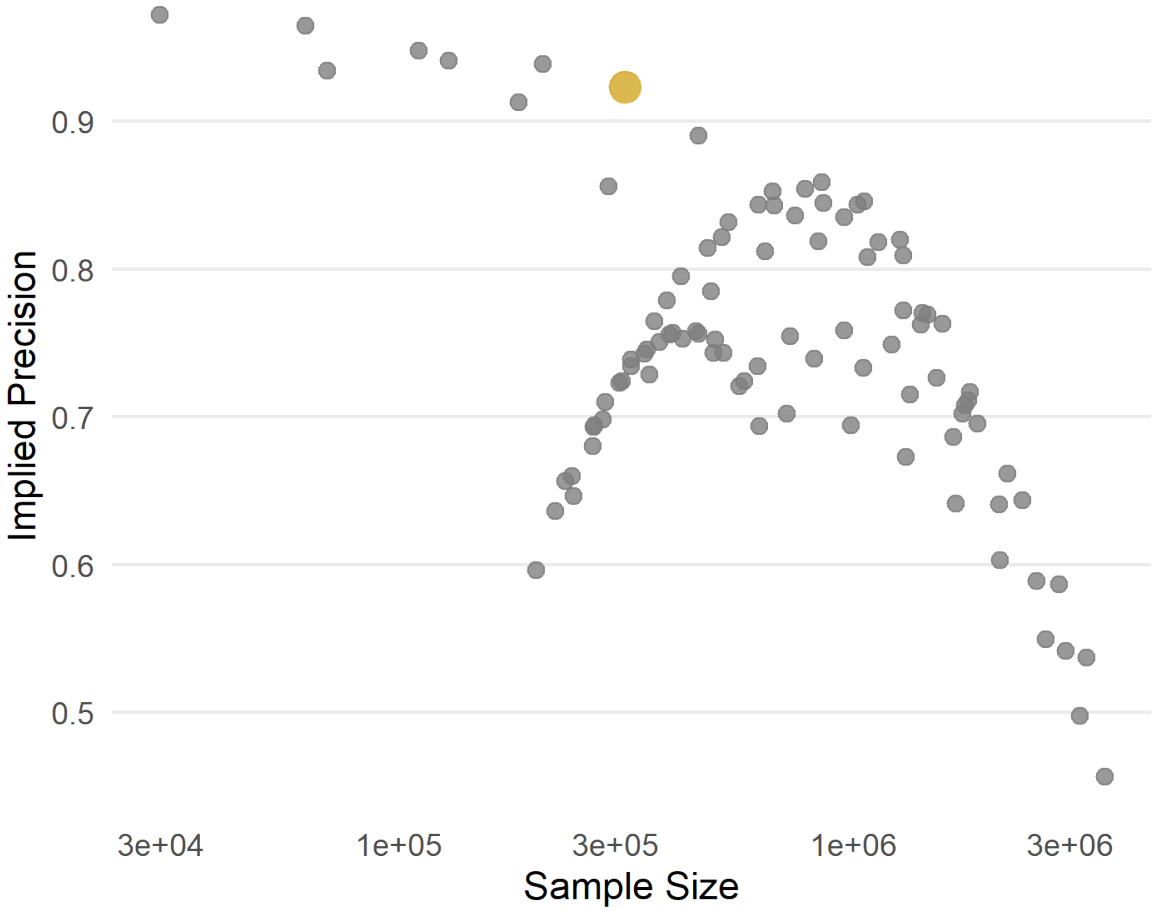


Figure 10: Precision and sample size of our algorithm as a function of different distance definitions and tolerances. Each point on the curve represents a specific combination of distance function and tolerance parameter. The large orange dot corresponds to our preferred specification, balancing precision and sample size.