Limitations and Opportunities of Bandit Algorithms for Feature Selection

Eddie (Yidi) Wu

Abstract

Feature selection is an important problem in statistical learning. This paper studies the performance of bandit-based feature selection and compares it with stability selection. Theories and simulation results show that bandit selection does not generally outperform stability selection for signal recovery and may suffer from high false positive rates, particularly in settings with correlated features or omitted variables. Nevertheless, bandit methods enjoy advantages in high-dimensional settings by allocating computational effort to promising regions of feature space and by adaptively handling combinatorial subset selection when exhaustive search is infeasible. We then discuss the strong identifiability condition under which bandit-based methods consistently select the set of true features, and propose a top-two Thompson sampling variant designed for pure exploration settings. Finally, we apply our method to the empirical asset pricing study of Gu et al. [2020] and obtain similar conclusions regarding signal importance.

1 Introduction

Feature selection has long been a central problem in statistics and machine learning, with a rich literature of methods tackling it. Suppose the true data generating process (DGP) has the form

$$y = f(\mathbf{z}) + \epsilon$$

where $f : \mathbb{R}^d \mapsto \mathbb{R}$ is measurable and ϵ is an error term satisfying $E[\epsilon|\mathbf{z}] = 0$. Let $S^* := \{1, ..., d\}$ denote the set of variables that enter the DGP. We define feature selection as the problem of recovering S^* from a random sample of $\{y_i, \mathbf{z}_i, \tilde{\mathbf{x}}_i\}_{i=1}^n$ with high probability, where $\tilde{\mathbf{x}}_i$ are features that can be correlated with $\mathbf{z}_i$ but do not directly enter the DGP. This objective focuses on support recovery of the true features, rather than minimizing generalization risk or maximizing out-of-sample prediction accuracy, though these objectives can be highly related in many settings.

Existing methods for feature selection can be divided into two categories. The first consists of one-shot methods such as information criterion, model-specific importance measures and model agnostic metrics like Shapley values [Lundberg and Lee, 2017] [Breiman, 2001]. They assess feature importance base based on a single model fitted on the full set of features in the data. The second includes iterative procedures, such as stepwise regression, stability selection and bandit-based algorithms [Meinshausen and Bühlmann, 2010] [Durand and Gagné, 2014]. These methods repeatedly sample training data or adaptively explore subsets of features to infer which features should be included in the final model.

The idea of stability selection is essentially subsampling from the entire training data multiple times, fitting the candidate model (a.k.a a base learner) on each subsample, and keeping track of whether a given feature is selected over the iterations. This produces a sequence of binary selection indicator for each feature which is then aggregated into selection probability. On the other hand, bandit-based methods use only a subset of features in each iteration, where the subset is chosen adaptively by a bandit algorithm. The intuition of doing this is two fold: first, moving towards the important regions of the feature space and dropping the less important noise variables, and second, improving computational efficiency by avoiding full model estimation using all features.

However, bandit-based methods face challenges in realistic settings. For instance, it is more often than not that covariates are highly correlated and interact with one another. Evaluating a subset of them might lead to distorted inference. In the basic case of a linear model, omitting true covariates induces bias in both the magnitude and direction of estimated coefficients through omitted variable bias:

$$\tilde{\beta}_1 = \beta_1 + (E[X_1 X_1^T]^{-1} E[X_1 X_2^T])\beta_2$$

When X_2 is omitted and the features in X_1 and X_2 are correlated, the parameter vector estimate $\tilde{\beta}_1$ of X_1 is biased. Therefore, excluding covariates can alter the importance of relevant covariates in the model and complicate support recovery.

Additionally, popular bandit algorithms such as Thompson sampling reply on assumptions such as independence of arm rewards and stationarity of reward distributions. They are often violated in the feature selection setting because reward in iteration t of a feature depends on the subset of features selected in the iteration and hence evolves over time, and

rewards can be correlated across features. These issues undermine the direct applicability of standard bandit algorithms without imposing additional structure on the problem.

In this paper, we study the category of iterative algorithms for feature selection by particularly focusing on stability selection and bandit-based approach. We first compare the accuracy of support recovery of stability selection with bandit-based methods. Our results show that even in simple settings, bandit-based approach can suffer from relatively high false positive rate and fail to consistently select the true feature set. We then provide theoretical explanations specific to the algorithms and DGP in our simulation setting.

Building on this analysis, we discuss sufficient condition for the consistency of bandit-based feature selection, which is the strong identifiability assumption of true signals in the DGP, and propose a pure exploration combinatorial bandit framework motivated by Liu and Rockova [2021] and Russo [2016] for feature selection. Then, we derive guarantees for its consistency and concentration of posterior probability mass on the true feature set. Next, we present details on practical implementation of the algorithm such as the choice of reward functions, and provide simulation evidence under high-dimensional settings. Our results demonstrate that the proposed approach achieves higher selection accuracy than running one-shot feature importance methods, and attains consistency of variable selection and more efficient posterior convergence than without exploration. Finally, we apply our method to the empirical asset pricing dataset of Gu et al. [2020] and show that it identifies economically meaningful predictors that are very similar to the authors' findings.

In terms of real-world relevance to practitioners, first, by explicitly promoting exploration and targeting support recovery, the method is able to distinguish signal from noise in high-dimensional settings with limited training data. Second, by adaptively exploring subsets of features, it reduces computations compared to methods that fit models on the full feature set. Lastly, the sequential nature of the algorithm allows it to natural adapt to online environment where new data arrives over time. In the modern data-rich environment where large number of features can be collected at the individual level, feature selection may be especially relevant for interpretability, policy design, or the construction of parsimonious and explanable models, in domains such as biology, healthcare, finance and digital platforms. In such applications, accurately identifying the key drivers of outcome is critical for decision making and scientific discovery. Our approach, while not without limitations, provides a flexible tool for achieving this goal.

1.1 Related work

The stability selection framework was first introduced by Meinshausen and Bühlmann [2010]. It combines sub-sampling with a general selection algorithm to compute selection probability of a feature over iterations. More specifically, they use a variant of LASSO as the learner and define selected variables in an iteration to be those with nonzero coefficients. Then, they provide theoretical bound on the expected number of false positives and prove consistency of this method using LASSO. Building on their work, Shah and Samworth [2012] propose complementary pairs stability selection, the false discovery error bound of which is tighter and requires weaker assumption on the underlying model. Moving forward, Beinrucker et al. [2016] extend stability selection by running the learner on disjoint subsets of covariates for each subsample in each iteration. This is very close in spirit to bandit feature selection,

except that they take random covariate subsets. We use stability selection with LASSO as a benchmark to assess the performance of bandit-based selection.

Regarding bandit-based frameworks, several papers have forged a connection between feature selection and reinforcement learning. Gaudel and Sebag [2010] set up feature selection as a Markov Decision Process (MDP) whose state space is given by the power set of the feature set and choose features to minimize the generalization error of the learned model. They solve the MDP with Monte Carlo tree search. Rasoul et al. [2021] tackle a similar MDP but employ a temporal difference algorithm to find the best subset of features. Ashtiani et al. [2014] partition the sample space into sub-regions and select features within each locality using Upper Confidence Tree.

Durand and Gagné [2014] formulate feature selection as a combinatorial multi-armed bandit problem. Their algorithm represents each feature as a bandit arm and iteratively selects subsets of features with constant subset size via Thompson sampling and computes rewards based on the accuracy of predictions by a neural network. Liu and Rockova [2021]'s framework combines Thompson sampling with a median probability model (MPM) computational oracle to choose the set of features at every time step and obtains rewards from BART. More importantly, they provide theoretical results on the regret bounds and consistency of their algorithm. Thompson sampling is known to attain the optimal logarithmic in-sample expected regret for the stochastic multi-armed bandit problem, [Agrawal and Goyal, 2012] as well as a logarithmic in-sample expected regret for the combinatorial bandit problem [Wang and Chen, 2018]. However, it only achieves a polynomial rate of convergence of posterior distributions of the arm parameters [Russo, 2016], and bandit algorithms with logarithmic expected regret bound is shown to be far from optimal for the best arm identification problem [Bubeck et al., 2009]. We seek to improve on this by shifting from exploitation to exploration.

In many settings where researchers are more concerned with finding the best subset of variables as quickly as possible rather than minimizing the cumulative error of selecting wrong variables during feature selection, algorithms biased towards exploration might be preferred. A number of papers have proposed stochastic (non-combinatorial) bandit algorithms where the probability of selecting a sub-optimal arm decays at an exponential rate [Bubeck et al., 2013] [Garivier and Kaufmann, 2016] [Chen et al., 2017] [Glynn and Juneja, 2018]. Notably, Russo [2016] proposes a class of top-two algorithms that converges at an exponential rate with the best possible exponent among all allocation rules. Kasy and Sautmann [2021] develop an exploration sampling algorithm which transforms the Thompson sampling probabilities and attains exponential rate of convergence. Caria et al. [2023] introduce Tempered Thompson Sampling which alternates between uniformly random assignment and Thompson sampling with a fixed probability at every iteration. The combinatorial bandit implementation for feature selection by Liu et al. [2021] takes a similar approach for exploration: their sampling algorithm has probability ϵ of uniformly choosing a random subset of features, and probability $(1-\epsilon)$ of choosing K features via Thompson sampling.

Several papers also address the question of best subset identification in combinatorial bandit. Wang and Zhu [2022] develop a pure exploration sampling algorithm which enhances exploration by playing subsets of arms with the largest reward gaps and the least number of observations. Nakamura and Sugiyama [2024] modify the algorithm of Wang and Zhu [2022] to attain a tighter upper bound on the sample complexity in a fixed confidence setting. However, these algorithms pull base arms one by one in a superarm, which generates valid

reward under their assumptions but cannot be applied mechanically to the feature selection setting. Therefore, we base our algorithm in this paper on Russo [2016] and propose a top-two Thompson sampling process for combinatorial bandit.

While previous work on adaptive feature selection such as Durand and Gagné [2014] and Liu and Rockova [2021] wed combinatorial bandit to a specific statistical model e.g. using tree-based models to compute bandit rewards, this paper aims to develop a model-agnostic framework by implementing model-agnostic reward functions. A common way of evaluating feature importance is the partial dependence plot first proposed by Friedman [2001], which computes the expected outcome for each value in the support of the feature of interest, where the expectation is taken over the joint distribution of all other features. Then, the differential in expected outcome at different values of the feature of interest conveys the importance of the feature. Another method is the Shapley value which originates from cooperative game theory studied by Shapley [1953]. It evaluates the contribution of a feature to outcome prediction by examining all feature subsets including and excluding the feature of interest. Several subsequent papers have overcome the computational intractability of Shapley value via efficient approximations Lundberg and Lee [2017] Rozemberczki et al. [2022].

Another prevalent method is permutation importance, proposed by Breiman [2001], which randomly permutes the values of a feature and examines the change in predictive performance of the model. As variable importance assessment based on tree splits in tree-based models is biased, permutation importance serves as a corrected measure Altmann et al. [2010]. Molnar et al. [2023] study the theoretical link behind permutation importance and the true DGP and formalize permutation importance as a statistical estimator of the ground truth estimand. Other commonly used model-agnostic feature importance method include Locally Interpretable Model-agnostic Explanations (LIME) proposed by Ribeiro et al. [2016]. The idea is to learn an interpretable model locally around the prediction to inspect variable importance. We define reward based on permutation importance and will explain this in more details in the subsequent sections. The potential limitations of this method and possible ways to overcome those are also discussed in Section 6.3.

1.2 Outline

In Section 2, we introduce the details of stability selection and existing bandit algorithms based on Thompson sampling.

In Section 3, using a linear DGP, we compare the accuracy of support recovery of bandit algorithms versus stability selection, with LASSO as the base learner. Then, we provide theoretical reasons to why the bandit algorithm might underperform stability selection even in simple linear settings.

In Section 4, we discuss the strong identifiability assumption which is sufficient for the consistency of bandit-based algorithms. Based on that, we introduce our top-two Thompson sampling for combinatorial multi-armed bandit, and present results on its consistency and posterior rate of convergence.

In Section 5, we expound on the implementation details of our method and demonstrate its performance in numerical simulations in comparison to existing methods.

Next, in Section 6, we apply bandit-based selection to the empirical asset pricing study of Gu et al. [2020].

Finally, in Section 7, we discuss the implications of our findings and suggest potential directions for future research.

2 Combinatorial Bandits for Feature Selection

2.1 Set-up and notations

The feature selection problem is formulated as follows. Let $(\mathbf{z}, y) \sim P$, where

$$y = f(\mathbf{z}) + \epsilon$$

$\mathbf{z} \in \mathbb{R}^d$, $y \in \mathbb{R}$, and $E[\epsilon|\mathbf{z}] = 0$. Let $S^* := \{1, ...d\}$ represent the set of features that are included in the DGP. Then, given an i.i.d sample $\mathcal{D}_n = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where the vector $\mathbf{x}_i = [\mathbf{z}_i, \tilde{\mathbf{x}}_i] \in \mathbb{R}^K$ with $K >> d$ and both d and S^* are unknown, the objective is to learn the true feature set S^* with high probability. To clarify, feature selection (or support recovery) in this context is not concerned with maximizing predictive accuracy or conducting accurate causal inference, although these objectives might be related.

One example of feature selection using a one-shot method is by running LASSO. Given a training sample of $(y_i, x_{1,i}, ...x_{K,i}), i = 1, ...n$, LASSO predicts y_i by $x_i^T\beta$, minimizing the following objective:

$$\hat{\beta} = argmin_\beta(\frac{1}{n}||Y - X\beta||_2^2 + \lambda||\beta||_1)$$

L1 penalty encourages sparsity in the coefficients. This can be seen by supposing the features are isotropic i.e. $\frac{1}{n}X^TX = I_K$, then $\hat{\beta}_j = sign(u_j)(|u_j| - \frac{\lambda}{2})_+$ where u_j is the j-th row of $u = \frac{1}{n}X^TY$. A larger λ suppresses more coefficients to be closer to 0. Feature selection can be based on the non-zero coefficients in $\hat{\beta}$.

2.2 Stability selection

Meinshausen and Bühlmann [2010] first introduce the stability selection algorithm. Instead of selecting feature in one-shot, it draws random subsamples from training data and aggregates binary indicator of whether a feature is selected in an iteration over all iterations. This ensures finite control over false positives and outperforms one-shot method. Algorithm 1 describes the algorithm in detail when using LASSO as the base learner.

2.3 Combinatorial multi-armed bandits

While stability selection uses the full set of features in every iteration, bandit-based method selects a subset of features using bandit algorithm and fits the base learner with features in the subset.

Before presenting combinatorial multi-armed bandits, we begin by introducing the basic bandit set-up. The canonical stochastic multi-armed bandit (MAB) is formulated as follows: suppose there are K actions indexed as $\{a_k, k = 1, ...K\}$. Each action yields a stochastic reward $r_t^k \sim \mathcal{P}(\theta_k)$ with a fixed distribution over time, parameterized by θ_k. At each time step, the agent follows a policy in which chooses one action to play to maximize some

Algorithm 1 Stability selection

Input: penalty parameter λ; number of iterations T; data D; selection threshold π.
Output: stable feature set $\hat{S} \subseteq 2^{[K]}$.

Initialize $Count_k \leftarrow 0$ for $k = 1, \ldots, K$
for $t = 1, \ldots, T$ **do**
    Draw a subsample D' from D without replacement.
    Run LASSO on D' with penalty λ.
    Let $\hat{S}_t$ be the selected feature set.
    $Count_k \leftarrow Count_k + 1$ for all $k \in \hat{S}_t$.
end for
$\Pi_k \leftarrow Count_k/T$ for $k = 1, \ldots, K$.
$\hat{S} \leftarrow \{k : \Pi_k \geq \pi\}$.

objective. The action choice in time t depends on all the actions and rewards observed in history: $\{a_1, r_1^{a_1}, ...a_{t-1}, r_{t-1}^{a_{t-1}}\}$.

The nature of the stochastic MAB problem depends on the objective under consideration. There are two types of objectives. The first type is to find a policy that maximizes expected cumulative reward (or equivalently minimizes expected cumulative regret). Algorithms for solving such objective need to balance exploration and exploitation. Another type of objective is to device a policy that either finds the optimal arm with as high confidence as possible in a given number of iterations (fixed budget setting), or find the optimal arm with a pre-specified confidence level in as few iterations as possible (fixed confidence setting). Algorithms for solving such objectives allocate more resources to exploration at the expense of larger regret.

Building on this, the stochastic combinatorial multi-armed bandit (CMAB) is formulated as: suppose there are K base arms: $\{a_k : k = 1, ...K\}$. The set of actions is now $\mathcal{S} \subseteq 2^{[K]}$. At each time step, the agent chooses a superarm $S \in \mathcal{S}$ to play. The agent either observes reward $R_t(S)$ for the superarm (full bandit feedback), or reward $r_t^k(S)$ for each base arm in the superarm (semi-bandit feedback), or both. Superarm reward can be a simple summation of base arm rewards, or a more sophisticated nonlinear function. Again, the objective can be to find a policy to either maximize cumulative reward or finding the best superarm with high confidence. Examples of stochastic CMAB can be finding the shortest path from point A to B in a graph, where each edge is a base arm, and a superarm is a set of base arms which can connect A to B, and the stochastic reward can be the negative of the time it takes to travel the edge. An objective could be to find the path that takes the shortest amount of time on average to go from A to B.

2.4 Bandit for feature selection

A straightforward but often intractable way to formulate feature selection as a bandit problem is to define every combination of features as separate bandit arms. Then, having defined an objective for feature selection, the usual stochastic MAB algorithms such as Upper Confidence Bound (UCB) and Thompson sampling (TS) can be implemented. However, this is computationally intractable in the sense that the number of arms grows exponentially in the

number of features.

Instead, existing literature suggests defining each feature to be a base arm and implementing appropriate stochastic CMAB algorithms. In each iteration, a superarm i.e. a subset of features is selected to fit a base learner which gives off reward. The reward can be full bandit Liu et al. [2021] or semi-bandit feedback Durand and Gagné [2014] Liu and Rockova [2021]. We adopt semi-bandit feedback which is more relevant to our setting since in many learners it is possible to observe individual feature importance, and updating features individually enables faster rate of convergence to the true feature set than updating a set of features altogether using full bandit feedback.

The following combinatorial Thompson sampling algorithm with semi-bandit feedback was first proposed by Durand and Gagné [2014], and later refined and theoretically justified by Liu and Rockova [2021]. We present the algorithm here, as described in Algorithm 2, and compare it with stability selection.

Algorithm 2 Thompson sampling for variable selection

Input: Beta prior α_0^k, β_0^k for $k = 1, ..., K$; number of iterations T; data D.
Output: Beta posterior α_T^k, β_T^k for $k = 1, ..., K$.

for $t = 1, \ldots, T$ **do**
  Draw $\theta_t^k \sim \text{Beta}(\alpha_{t-1}^k, \beta_{t-1}^k)$ for $k = 1, ..., P$.
  $S_t \leftarrow \{k : \theta_t^k \geq 0.5\}$.
  Observe $r_t^k \sim Q_{s_t}$, from the reward distribution of S_t.
  Update Beta parameters:
  for all $k \in S_t$ **do**
    $\alpha_t^k \leftarrow \alpha_{t-1}^k + r_t^k$
    $\beta_t^k \leftarrow \beta_{t-1}^k + (1 - r_t^k)$
  end for
end for

To begin with, the agent places Beta distribution priors with parameters α_0^k, β_0^k on each feature. These parameters govern the player's initial knowledge of feature inclusion probability. Having $\alpha_{k,0} = 1$ and $\beta_{k,0} = 1$ is equivalent to uniform prior on $[0, 1]$ while adjusting the parameter values can encode prior knowledge.

At the first time step $t = 1$, the agent samples a θ_1^k for every feature, and define a super-arm S_t to include features with $\theta_1^k \geq 0.5$. The inclusion threshold at 0.5 is justified by Liu and Rockova [2021] by deriving it from the median probability model (MPM) as they argue that it is standard practice to report the MPM in model selection, and thresholding the selection probability of each feature at 0.5 is the maximizer of posterior probability of a feature set in Bayesian model selection with spike-and-slab prior. They also provide worst case regret bound on the order of $O(\log T)$ when the superarm inclusion probability is 0.5.

Next, the agent fits a base learner using the chosen features S_t, and observe the binary 0-1 base-arm reward r_t^k for each feature. We note that contrary to the usual stochastic CMAB set-up, Q_{S_t} is explicit dependent on the feature set S_t chosen at time t, which is a more realistic setting than stationary reward for feature selection.

Finally, we update the Beta distribution of each feature k using the binary r_t^k, to obtain the posterior distributions at the end of this time step. This is repeated for a pre-specified

number of iterations, or till convergence. An example of convergence criterion in implementation could be that the ranking of posterior means of the Beta distributions stays stable for a pre-specified number of iterations. In simulation set-up where we know the true feature set which say has size d, convergence could be defined as posterior means of features in the true feature set being in top d among all posterior means.

3 Stability Selection versus Bandit Algorithms

In this section, we compare the accuracy of selecting the true features of stability selection versus Thompson sampling. We consider two DGPs, a simple one with isotropic features and a more challenging one with correlated features.

For the isotropic feature model, the DGP is:

$$y = X_{p,K}\beta_{p,K} + \epsilon \tag{1}$$

where $X_{p,K} = [X(p), X(K)]$, $X(p) \in \mathbb{R}^{n\times p}$ and $X(K) \in \mathbb{R}^{n\times(K-p)}$, and

$$\beta_{p,K} = \begin{pmatrix} \beta^* \\ 0 \end{pmatrix}, \qquad \beta^* \in \mathbb{R}^p \tag{2}$$

and $x_i \sim N(0, I_K)$, $\epsilon_i \sim N(0, \sigma_\epsilon^2)$, $\beta^* = [1, ..., 1]^T$. The first p features form the true feature set while the remaining $K - p$ features are noise.

For the correlated feature model, it is a highly challenging DGP from Meinshausen and Bühlmann [2010] where x has a factor structure:

$$X_{p,K} = F\phi + \eta \tag{3}$$

where we set $F \in \mathbb{R}^{n\times 2}, F_i \sim N(0, I_2)$ and $\phi \sim N(0, I_{2\times K})$, $\eta_i \sim N(0, \sigma_\eta^2)$. y is generated in the same way as equation 1. Correlations of features in the range of $[-0.81, 0.80]$, with an average absolute correlation of 0.2 across all pairs of features.

We fix training sample size to be $n = 200$, the total number of features to be $K = 1000$, and the number of true signals $p = 10$, noise signals $K - p = 990$. We run stability selection and Thompson sampling for several specifications of maximum iterations ranging from 100 to 500. In each iteration, we either bootstrap a sample of size 200 or sub-sample with size 100. The maximum number of iterations and how sample is generated in each iteration does not change the conclusions of our findings. In bandit selection, reward function is defined such that, in each iteration, features with non-zero coefficients earn a reward of 1, while other features in the superarm in that iteration earns a reward of 0.

3.1 Simulation results

Suppose we run stability selection for t iterations, the selection probability of a feature k at the end of time t is $\pi_t^k = \frac{c_t^k}{t}$ where c_t^k is the cumulative number of times that feature k has been selected over t iterations. For bandit selection, the selection probability of feature k is the posterior mean $\frac{\alpha_t^k}{\alpha_t^k+\beta_t^k}$, which is almost identical to π_t^k in definition since α_t^k and β_t^k also

keep track of the number of selections and no-selections, except that Beta posterior mean, by specifying $\alpha_0^k = \beta_0^k = 1$, starts from $1/2$ at $t = 0$.

First, using a fixed λ value for LASSO penalty, we study the evolution of selection probabilities over iteration. Figure 1 plots the results for the two methods.

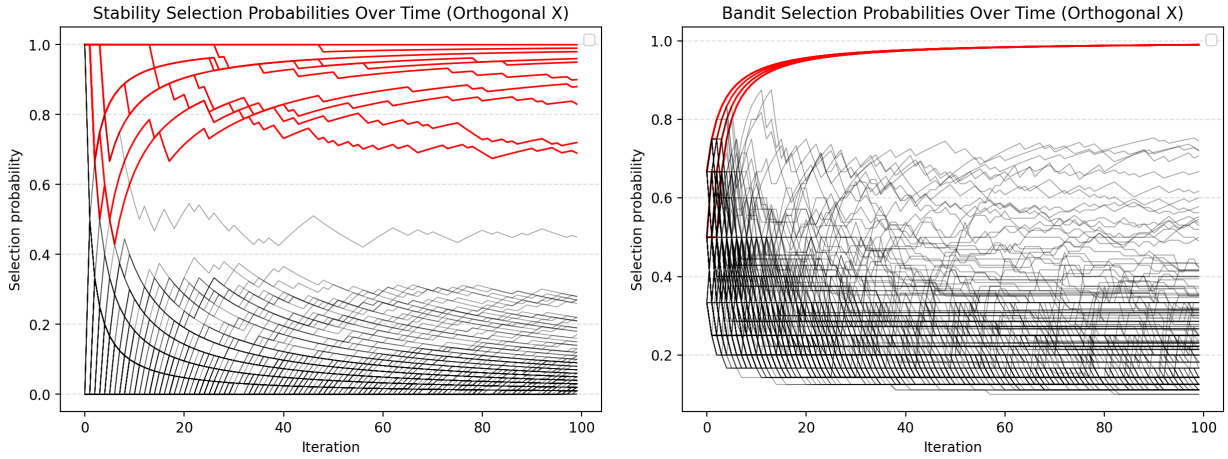

Figure 1: Selection probability against iteration for stability selection (left) and bandit selection (right). Red is true feature while black is noise. In iteration t, selection stability is the number of times a feature has been selected divide by the number of times a feature has been played over the past t iterations.

The red curves are the true features while the black curves are the noise variables. Although stability selection does not select the true features with confidence close to 1, suppose we set the eventual selection threshold at 0.5, the false positive rate is 0 for stability selection, but for bandit selection, although the probabilities of true features are close to 1, we also have the posterior probabilities of many noise features hovering between 0.6 to 0.8. In actual application, it would be difficult to distinguish from the plot alone whether those features should be classified as signal. With a selection threshold of 0.5, there would be many false positives. Increasing the maximum number of iterations does not reduce the false positives of bandit selection.

To illustrate this more comprehensively, next we look at the selection probabilities at the end of max iterations of each feature plotted against λ for the two methods, as shown in Figure 2.

At each value of λ, we plot the final selection probability at the end of maximum iteration for each feature. Stability selection attains a much better separation of noise and signal across a wide range of λ values, while bandit again suffers from high false positive rate.

Figure 3 is the counterpart of Figure 2 for the more challenging factor X model. Separation deteriorates for both models but stability selection maintains its edge in having better separation than bandit selection. This suggests that even for linear DGP with correct model specification, bandit algorithms may suffer from high false positive rate.

Additionally, one might also consider a variant of the LASSO bandit in which the regularization parameter λ is allowed to vary over iterations rather than being fixed. However, this approach does not improve performance. In fact, it leads to a substantially higher false

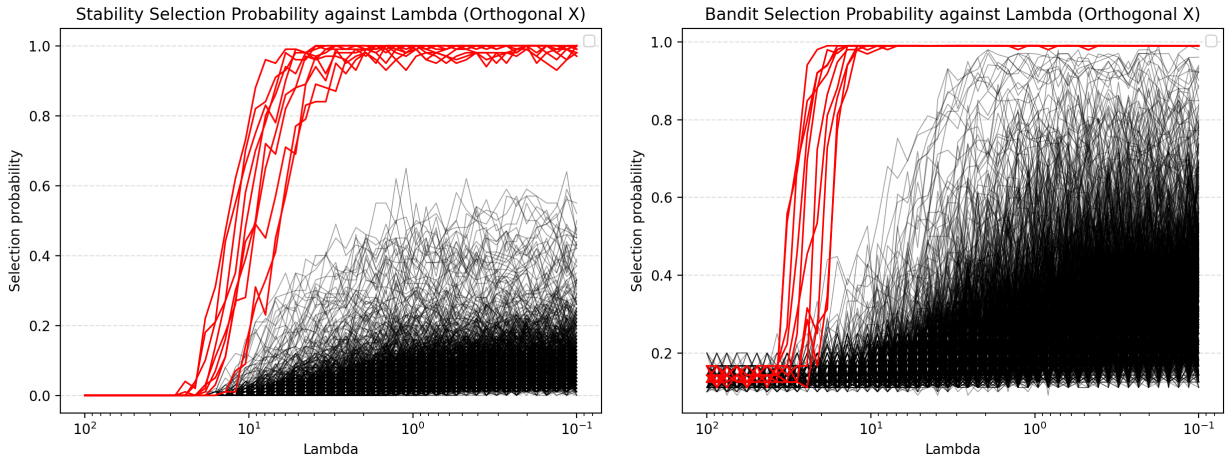

Figure 2: Selection probability against LASSO penalty λ for the isotropic feature model. Selection probabilities for each λ are the final probabilities at the end of maximum iteration.

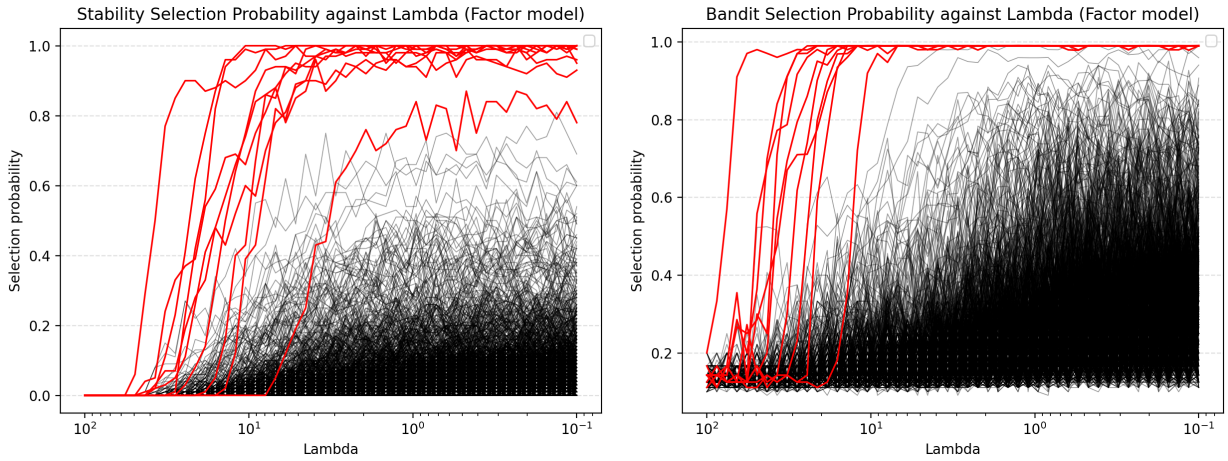

Figure 3: Selection probability against LASSO penalty λ for the factor structure feature model. Selection probabilities for each λ are the final probabilities at the end of maximum iteration. Separation between signal and noise deteriorates for both methods due to feature correlations.

positive rate than fixing λ in Figure 1. The underlying issue is that in this adaptive setting, standard data-driven methods for selecting λ, such as k-fold cross-validation, are aimed at optimizing out-of-sample prediction accuracy instead of true feature set recovery, hence λ tends to be small, resulting in the inclusion of many noise variables in finite sample. More details and evidence are provided in Appendix A.

3.2 Potential limitations of bandit selection

In this subsection, we offer some explanations on why bandit selection might underperform stability selection when using LASSO as the base learner. We build our analysis on Proposition 1 in Zhao and Yu [2006], who derive the condition for LASSO feature selection consistency and lower bound on the probability of correct sign identification. First, we define the $sign(.)$ function as:

$$sign(z) = \begin{cases} 1 \quad if \quad z > 0 \\ 0 \quad if \quad z = 0 \\ -1 \quad if \quad z < 0 \end{cases}$$

Next, consider the linear model:

$$y = X_{p,K}\beta_{p,K} + \epsilon$$

where $X_{p,K} = [X(p), X(K)]$, $X(p) \in \mathbb{R}^{n\times p}$ and $X(K) \in \mathbb{R}^{n\times(K-p)}$, and

$$\beta_{p,K} = \begin{pmatrix} \beta^* \\ 0 \end{pmatrix}, \qquad \beta^* \in \mathbb{R}^p \tag{4}$$

Given the design matrix $X_{p,K}$, let

$$\Sigma^n_{p,K} := \begin{pmatrix} C^n_{11} & C^n_{12} \\ C^n_{21} & C^n_{22} \end{pmatrix} = \frac{1}{n} X^T_{p,K} X_{p,K} \tag{5}$$

denote the sample covariance matrix of $X_{p,K}$ partitioned according to true and noise features, i.e. the C^n_{11} block is the covariance matrix of the true features while C^n_{22} is the covariance matrix of the noise features. Also, let $\hat{\beta}(\lambda)_{p,K}$ denote the LASSO estimate of y on $[X(p), X(K)]$ for a fixed λ. We seek to understand how restricting LASSO to a subset of true and noise features affects the probability of recovering the coefficients. This is motivated by the difference in the choice of feature set between stability selection and bandit feature selection. The former uses the full feature set in every iteration while the latter uses only a subset of features. We begin by considering the case where in some iterations, bandit algorithm retains all true features and only drop noise features in the chosen feature set.

Corollary 3.1 (Effect of omitting noise on LASSO sign identification) *Assuming that the Irrepresentable Condition (IC) holds for the full design matrix $X_{p,K}$ i.e.*

$$|C^n_{21}(C^n_{11})^{-1} sign(\beta^*)| \leq \mathbf{1} - \eta$$

with a positive constant vector η, then the IC also holds for $X_{p,k}$ where $p \leq k \leq K$ i.e. using all the true features but a subset of noise features. Moreover, the lower bound of the probability of correct sign identification i.e. $\inf P(sign(\hat{\beta}(\lambda)_{p,k}) = sign(\beta_{p,k}))$ *is non-increasing in the number of noise features* $(k - p)$.

Proof: Please refer to Appendix B.1. □

To interpret this corollary, first, IC is the sufficient and almost necessary condition for consistent LASSO sign identification Zhao and Yu [2006]. It is almost necessary due to minor mathematical technicality related to having $\leq$ or $<$ in the IC. They show that when IC holds, probability of correct sign identification by LASSO converges to 1 at a near exponential rate in n under the setting with a fixed number of features.

Corollary 3.1 says that when running LASSO with a subset of features, if we retain all true features and drop some noise features, the IC still holds for the smaller design matrix and the lower bound of the finite-sample probability of correct sign identification can only increase in the presence of fewer noise features. This is rather intuitive and implies that if the bandit algorithm is able to choose all the true features together with some noise features in the superarm, this is an improvement over using the full set of features.

Next, we consider the case where some true features are dropped from the feature set and either some or no noise features are dropped. This is when using only a subset of features instead of the full set becomes detrimental to consistent feature selection. Note that this leads to the omitted variables setting in linear regression, so the consistency of LASSO estimator we are referring to here is with respect to the pseudo-truth parameter which corresponds to the subset of the true features retained, instead of the original full DGP.

Corollary 3.2 (IC violation from omitting signals) *Consider the same linear model set-up, let $X_{q,k}$ be the design matrix which contains a subset $q < p$ of true features and a subset $(k - q)$ of noise features. There exists a distribution $(x_i, \epsilon_i) \sim \mathcal{P}$ and a coefficient vector $\beta_{p,K}$ such that, given a sample of size n drawn from $\mathcal{P}$, the sample covariance matrix of the full design matrix $\Sigma^n_{p,K} = \frac{1}{n} X^T_{p,K} X_{p,K}$ satisfies the IC but the sample covariance matrix of the design matrix with omitted true features $X_{q,k}$, where $q < p$, violates the IC.*

Proof: Please refer to Appendix 3.2. □

Corollary 3.2 points out that there exists settings where dropping some true features from the design matrix violates the IC even though the full design matrix satisfies IC. Since the IC is sufficient and almost necessary for sign consistency, dropping some true features might results in the situation where LASSO cannot consistently identify the true features with correct sign. We note that sign identification is stronger than mere support recovery in feature selection.

The effect of dropping true features has more harm than just violating IC. In the case where IC is satisfied for both the full design matrix and the smaller design matrix obtained after dropping some true features, one might think that probability of correct sign identification (to the pseudo-truth parameter) is larger in the smaller model than in the full model. the following proposition shows that this is not necessarily the case.

Proposition 3.1 (Effect of omitting signals on LASSO sign identification) *Consider the same linear model set-up, let $X_{q,k}$ be the design matrix which contains a subset $q < p$ of true features and a subset $(k - q)$ of noise features. Let*

$$\beta_{q,k} = \begin{pmatrix} \tilde{\beta} \\ 0 \end{pmatrix}, \qquad \tilde{\beta} \in \mathbb{R}^q$$

be the pseudo-truth coefficient vector from omitting some true features. Suppose that both the sample covariance matrix of the full design matrix $X_{p,K}$ and the sample covariance matrix with omitted features $X_{q,k}$ satisfy the IC, there exists a distribution $(x_i, \epsilon_i) \sim \mathcal{P}$ and a coefficient vector $\beta_{p,K}$ such that, for a sample of size n drawn from $\mathcal{P}$,

$$\inf P(sign(\hat{\beta}_{q,k}) = sign(\beta_{q,k})) < \inf P(sign(\hat{\beta}_{p,K}) = sign(\beta_{p,K}))$$

Proof: Refer to the Appendix 3.1. □

Proposition 3.1 suggests that although it might be tempting to think that using a smaller subset of features has a higher probability lower bound of correct sign identification for the pseudo truth coefficient vector, there exists cases where using a smaller subset actually reduces the probability lower bound of correct sign identification versus using the full set of features. While the analysis based on the previous three theoretical results is not comprehensive, it points out some limitations of using a subset of features instead of the full feature set, and partially explains why bandit underperforms stability selection even in the linear DGP setting and using a learner with correct functional form specification.

4 Bandit Algorithms Consistency

Having highlighted the potential shortcomings of using a subset of features for feature selection, a natural question is, under what conditions can we guarantee consistency for bandit-based methods? For the class of Thompson sampling algorithms, a sufficient condition for the consistency of bandit feature selection is proposed by Liu and Rockova [2021], which characterizes the signal strength of the true features versus noise in the DGP:

Assumption 1 (Strong identifiability) *Define $S^* := argmax_S E[R_t(S)]$ as the optimal superarm. S^* is strongly identifiable if $\exists \alpha$ where $0 < \alpha < 0.5$ such that:*

- *$\forall k \in S^*$, $E[r_t^k | S^*] \geq E[r_t^k | S_t] > 0.5 + \alpha$ for all S_t which contain k, and for all t.*
- *$\forall k \notin S^*$, $E[r_t^k | S_t] < 0.5 - \alpha$ for all S_t which contain k, and for all t.*

Since the semi-bandit reward r_t^k is binary in Thompson sampling algorithms, the assumption essentially requires that a true feature receives a reward of 1 more often than not, regardless of what subset is played at time t, while a noise feature receives a reward of 0 more often than not, uniformly over all S_t. This suggests that the signal strength of the true features are so strong that it does not get tarnished by the inclusion or exclusion of the other features, while the signal strength of all noise features is weak. Hence, there is a

robust separation of signal strength between feature and noise, resulting in the true features being strongly identifiable.

We think this is a rather strong assumption since in many practical settings, the contribution of a feature may depend crucially on its interaction with other features due to redundancy, complementarity or model misspecification. That said, since many bandit algorithms learns from feature-level semi-bandit rewards, some form of stable feature-wise signal is necessary. If feature signal strength can vary arbitrarily based on the subset played, then it is highly challenging for algorithm based on semi-bandit feedback to reliably distinguish signal from noise.

This assumption provides an identifiable regime in which semi-bandit feedback is informative enough for consistent feature recovery, and it establishes a benchmark setting on which one can consider more general dependency structures.

Under this assumption, while existing Thompson sampling algorithms for feature selection focuses on attaining a regret guarantee on the order of $O(\log T)$ by balancing exploration and exploitation, we consider the setting where the regret incurred from selecting a sub-optimal set of features during experimentation is small versus the regret incurring from implementing the optimal policy after the experimentation phase.

Very often, learning happens before implementation and greater exploration might be preferred to exploitation when experimental regret is negligible relative to simple regret which includes future deployment costs. As such, our objective is to find the optimal superarm, with fixed confidence level, in as few iterations as possible. This tilts the balance more towards exploration from exploitation.

We are motivated by Russo [2016] who shows the exponential rate of posterior concentration using top-two Thompson sampling for classical stochastic MAB. In his canonical bandit set-up, in each iteration, instead of playing the arm with the highest sampled value, he runs the sampling process a second time to obtain a competing arm, and play either the first arm or the competing arm with non-zero probability. This encourages more exploration at the expense of higher regret. Based on this idea, we propose the top-two Thompson sampling for CMAB:

The algorithm requires an additional input hyperparameter $\delta \in [0, 1]$ which controls the degree of exploration of this algorithm. In each iteration, the algorithm samples two sets of superarms that are not identical. It plays the first superarm with probability $1 - \delta$, and plays the union minus the intersection of the two superams with probability δ.

There are several details worth highlighting. Contrary to Thompson sampling which is optimal for minimizing cumulative regret, top-two Thompson sampling limits the exploitation of features with high known importance to promote greater exploration. Taking the union minus the intersection of the two candidate superarms is essentially shifting away from regions of feature space in which we have high confidence and focusing on the under-explored regions at every time step. This allocates more measurement effort to explore uncertain regions of the feature space, leading to faster concentration of posterior mass on the optimal set.

Under the strong identifiability assumption, we established the consistency and convergence rate results for top-two Thompson sampling (TTTS) for CMAB in the following proposition:

Algorithm 3 Top-two Thompson sampling for CMAB

Input: Beta prior α_0^k, β_0^k for $k = 1, ..., K$; max iterations T; data D; exploration probability δ.

Output: Beta posterior α_T^k, β_T^k for $k = 1, ..., K$.

for $t = 1, \ldots, T$ **do**
  Draw $\theta_t^k \sim \text{Beta}(\alpha_{t-1}^k, \beta_{t-1}^k)$ $\forall k$. Draw $u_t \sim U[0, 1]$.
  $S_t \leftarrow \{k : \theta_t^k \geq 0.5\}$.
  if $u_t \leq \delta$ **then**
    Draw S_t' in the same way as S_t until $S_t' \neq S_t$.
    $S_t \leftarrow (S_t \cup S_t') \setminus (S_t \cap S_t')$
  end if
  Observe $r_t^k \sim Q_{s_t}$ from the reward distribution of S_t.
  for all $k \in S_t$, **do**
    Update α_t^k, β_t^k
  end for
end for

Proposition 4.1 (Posterior concentration) *Under the strong identifiability assumption, in the top-two Thompson sampling for CMAB algorithm, the following holds for each base arm k:*

1. *The number of times it is played up till time t satisfies:*

$$N_k(t, \delta) \to \infty \ a.s.$$

2. *The posterior probability of selection converges to:*

$$\Pi_t(\{\theta_k \geq 0.5\}) \to \mathbf{1}\{i \in S^*\} \ a.s.$$

   where θ_k is posterior mean. This implies that $\Pi_t(S^*) \to 1$ a.s.

3. *Moreover, there exists $c_k > 0$ and $T_k > 0$ for each k s.t. for all $t > T_k$*

$$\Pi_t(\{k \ misclassified\}) \leq e^{-c_k N_k(t,\delta)}$$

   .

Proof: Please refer to Appendix C.

□

We do not need to assume that the reward function is stationary over time or independent across features due to the dependence on S_t at every time step. Under strong identifiability, posterior convergence can be established under the martingale strong law. The exponential rate of convergence depending on $N_k(t, \delta)$ comes from the Beta distribution exponential tail KL-divergence type bound. We note that while the rate of convergence is exponential in $N_k(t, \delta)$, it is not necessarily exponential in t. If $N_k(t, \delta)$ is logarithmic in t, then the rate

of posterior convergence is only polynomial in t. Therefore, it remains a question for future research to verify that the convergence rate of this algorithm is also exponential in t.

That said, we provide some intuitions on why TTTS has a faster convergence rate than Thompson sampling. From Lemma C.1 which is part of the proof for this proposition in Appendix C, we observe that in TTTS, the probability of a feature k being played in a round is maximized at the posterior mean of k equaling 0.5, and this probability, which has the shape of a parabola, decreases gradually to 0 as posterior mean approaches either 0 or 1. This means that the features for which we are more certain of its selection probability are less likely to be explored again, and sampling effort is allocated to those with posterior inclusion mean close to 0.5 i.e. we are less certain about its inclusion probability. On the contrary, in Thompson sampling, the higher the posterior mean, the more likely it is chosen in a subset to reduce the regret from choosing a sub-optimal subset.

Additionally, we note that in Russo [2016], the optimal exploration parameter is not a fixed constant but varying over time according to the posterior distributions in the canonical stochastic MAB setting. He shows that fixing the exploration parameter at 0.5, the exponent in the convergence rate is within a factor of 2 of the optimal exponent across all problem instances. As such, in our TTTS for CMAB, the optimal δ should also vary across iteration. In our actual implementation, we set it as 0.5, although future research should investigate how far the convergence rate with $\delta = 0.5$ deviates from the convergence rate when using optimal δ.

5 Algorithm Implementation Details and Simulations

One important consideration when implementing Thompson sampling type of bandit algorithm for feature selection is the design of the reward function. Due to the nature of the Beta distribution prior, reward is binary for each feature k and the likelihood is Bernoulli with mean depending on S_t, in order to update the Beta distributions by Bayes rule. Note that the strong identifiability assumption depends not only on the true DGP, but also the choice of base learner and reward function. Previous work including Durand and Gagné [2014], Wang et al. [2014] and Liu and Rockova [2021] combine Thompson sampling with a specific statistical model such as tree-based method and multi-layered perceptron. They define the reward function to be model specific, such as a binary indicator of whether decision tree splits on a feature. Another example is a binary indicator of whether a feature has non-zero LASSO coefficient, which we have done in Section 3.

As a generalization of this approach, we hope to increase the flexibility of implementing TTTS for feature selection by using model-agnostic methods of assessing feature importance to compute semi-bandit reward. Such reward functions can flexibly accommodate many different base learners, although each has its limitations so practitioner has to exercise discretion in the choice of reward function according to prior understanding of the problem at hand. In this section, we review three popular methods of assessing feature importance - partial dependence (PD), Shapley value (SV), and permutation importance (PI). We choose PI in our implementation and explain the advantages of PI over the other methods under our settings.

PD measures the marginal effect of a feature X_k on the predicted outcome $f(X_k)$ where

$f(\cdot)$ is the true functional form of the model. Given a point x_k in the support of X_k, the PD function $f_{PD}(\cdot)$ is defined as:

$$f_{PD}(x_k) = E_{X'} f(x_k, X') = \int f(x_k, X') dP(X')$$

i.e. the average predicted outcome when $X_k = x_k$. In terms of implementation, the sample PD function can be computed by selecting a set of D points in the support of X_k and using bootstrap for each of the points. Then, the PD feature importance for a feature X_k is given by:

$$\sqrt{\frac{1}{D-1} \sum_{d=1}^{D} \left(\hat{f}_{PD}(x_k^{(d)}) - \frac{1}{D} \sum_{d=1}^{D} \hat{f}_{PD}(x_k^{(d)}) \right)}$$

i.e. the variance in the predicted outcome over the set of D points. There are two major drawbacks to using this feature importance measure as the reward function. First, PD does not capture the interactions between features. Suppose a feature k affects the outcome only via interaction with other features, there is a possibility that PD show a flat relationship between the outcome and feature k. Second, since computing PD requires bootstrap, when data is high dimensional and the number of observations is small relative to the number of features, PD estimation tends to have large variance. Thus, PD is not suitable as a reward function in our setup.

SV assesses the amount of contribution of a feature to the predicted outcome. Let F be the set of all features, and $f_S(\cdot)$ be a model that includes only features in the subset S, SV of a feature k is defined as:

$$\phi_k = E_{X_{S \cup \{k\}}} \left[\sum_{S \subseteq F \setminus \{k\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left[f_{S \cup \{k\}}(X_{S \cup \{k\}}) - f_S(X_S) \right] \right]$$

i.e. the expectation of the weighted average difference in predicted outcome with feature k and predicted outcome without feature k. As the exact formula involves an exponential number of subsets of F, several computationally tractable methods such as SHAP have been proposed to approximate SV Lundberg and Lee [2017]. While SV appears to be a good measure of feature importance, it depends strongly on the estimated model $\hat{f}(S \cup \{k\})$. Consider a trained model which overfits to noise in training data and produces poor out-of-sample predictions, some noise variables will have large SV due to their contribution to the predicted outcome. SV will consider those variables important even though they are just noise. As such, SV requires careful model choice and hyperparameter tuning.

PI measures the drop in out-of-sample fit of a model if feature k is randomly permuted in the out-of-sample data. Let X_k be a feature whose importance we are interested in, $\tilde{X}_k$ be feature k after permutation, X' be all the other features, $P(\tilde{X}_k | X')$ be the conditional density of $\tilde{X}_k$, and $L(\cdot)$ a loss function, then PI is defined as: Molnar et al. [2023]

$$\pi_k = E_{X',Y} \left[E_{\tilde{X}_k | X'} \left(L(Y, f(\tilde{X}_k, X')) \right) \right] - E_{X,Y} \left[L(Y, f(X)) \right]$$

Permutation breaks the relationship between feature k and the outcome. If feature k is truly informative, the change in loss will be significant. For actual implementation, the dataset

is divided into a training and a test set. The training set is used to obtain the estimated model $\hat{f}(\cdot)$ while the test set is used to compute PI. To convert PI into a binary reward r_t^k, one can either use the PI value of feature k as the Bernoulli parameter to sample a binary reward, or binarize the PI value by thresholding at a pre-specified ν. This depends on the practitioner's belief and prior knowledge about the problem, e.g. suppose a researcher deems a feature to be in the true DGP if it leads to a 5% or more drop in test fit, then he can set $\nu = 0.05$, and $r_t^k = \mathbf{1}(PI_t^k \geq \nu)$.

Given a choice of model, PI does not require optimal hyperparameter tuning because it examines the relative change in loss on out-of-sample data. Even if the model overfits to noise, permuting noise variables does not influence out-of-sample loss considerably while permuting true important variables might relatively worsen the out-of-sample model performance. On the other hand, if the model underfits, true but weak signals might have little impact on test fit both before and after permutation. Therefore, if one believes that the true DGP is complex, it is recommended to use more complex models or conduct hyperparameter search in each iteration to avoid underfitting and missing true signals. On simulated data, Altmann et al. [2010] show that PI works very well in deciding the significance of variables and distillating the important variables. More details on computation are presented in algorithm 4. The drawbacks of PI, particularly when high feature correlation can diminish PI of these features, together with ways to address them, are discussed in section 6.3.

Algorithm 4 Permutation importance

Input : Fitted estimator M; test data D_{out}; feature k.

Output : Feature importance π_k

Initialize the number of times to repeat permutation B.

for $b \in [1, B]$ **do**
  $\hat{y} \leftarrow M(D_{out})$
  $R^2 \leftarrow L(y, \hat{y})$ where $L(\cdot)$ is the function for computing R^2.
  Permute feature k in the design matrix of D_{out} to obtain D'_{out}.
  $\hat{y} \leftarrow M(D'_{out})$
  $R'^2 \leftarrow L(y, \hat{y})$
  $\pi_{b,k} \leftarrow R^2 - R'^2$
end for
$\pi_k \leftarrow \frac{1}{B} \sum_{b=1}^{B} \pi_{b,k}$

5.1 Comparing Bandit with model-specific methods

Running simulations based on known DGPs, we demonstrate several merits of bandit feature selection. First, it attains a higher accuracy of identifying the true important variables from noise compared to one-shot methods. Second, using top-two Thompson sampling, which aids exploration, achieves a faster rate of convergence than using Thompson sampling, and is less likely to be trapped in local minima. Lastly, the iterative learning algorithm is highly versatile because it is compatible with a broad range of machine learning algorithms and can work in both offline and online settings. In this subsection, we show the performance

of TTTS for CMAB for two challenging nonlinear DGPs, using random forest regressor as the base estimator. Random forest is a ensemble method which predicts outcome as the averaged predicted outcome of a set of regression trees:

$$\hat{y} = \frac{1}{M} \sum_{m=1}^{M} T_m(x)$$

where M is the number of regression trees and $T_m(X)$ is the prediction of a tree m conditioning on x.

We consider the following DGPs, both based on examples from Friedman [1991]:

$$y_i = 10 \sin(\pi X_{i,1} X_{i,2}) + 20(X_{i,3} - 0.5)^2 + 10X_{i,4} + 5X_{i,5} + \epsilon_i \tag{6}$$

$$y_i = 0.1e^{4x_{i,1}} + \frac{4}{1 + e^{-20(x_{i,2} - 0.5)}} + 4x_{i,3} + 3x_{i,4} + 2x_{i,5} + \epsilon_i \tag{7}$$

where $X_i \sim Unif[0,1]^p$, $\epsilon \sim N(0, 0.5^2)$. We set n = 300 and p = 1000, so it is a high-dimensional setting with sparsity. The outcome is nonlinear in some features and linearly additive in the other features.

Figures 4 and 5 plot the results for the two DGPs respectively. The left most panel shows built-in impurity-based importance and the middle panel shows permutation importance computed from the optimally tuned random forest regressor. The model estimation is process is that we first conduct hyperparameter search of random forest via grid search and k-fold cross validation on the training data and then refit on the full training data using the optimal hyperparameters. With the fitted model, we obtain the built-in impurity based feature importance, i.e. the importance of a feature is the mean reduction in variance across all splits on that feature in the forest, and then compute the permutation importance of each feature by algorithm 4.

Then, using the same training data, we run bandit feature selection by algorithm 3 with an arbitrary set of random forest hyperparameters, uniform Beta prior, and 200 iterations. This is to simulate a real-world situation where one might not have sufficient data for k-fold cross validation and hyperparameter tuning. Hence, I do not use the optimal hyperparameters found previously, but specify the hyperparameters based on heuristics, e.g. `n_estimators`=100 and `max_depth`=10.

In Figure 4, both one-shot methods are unable to separate x_3 from noise and attribute a low importance to x_5, i.e. they miss one feature out of the five true features. The right panel (c) shows the selection probabilities of each feature over time using TTTS, where the true features are plotted in red. All noise features have selection probabilities suppressed below 0.5, while x_1, x_2, x_4, x_5 have selection probabilities rising to above 0.5 in mere 15 iterations. x_3 has inclusion probability fluctuating around 0.3 initially, but once the algorithm is highly confident about the other true features, it is also able to learn the importance of x_3 in later iterations. By the end of 200 iterations, we see a clear separation between noise and all true features.

In Figure 5, the most challenging variables to be identified are x_4 and x_5. Impurity-based importance fails to pick out x_5 and assigns a very small importance to x_4, while one-shot permutation importance computed directly on the fitted random forest with optimal

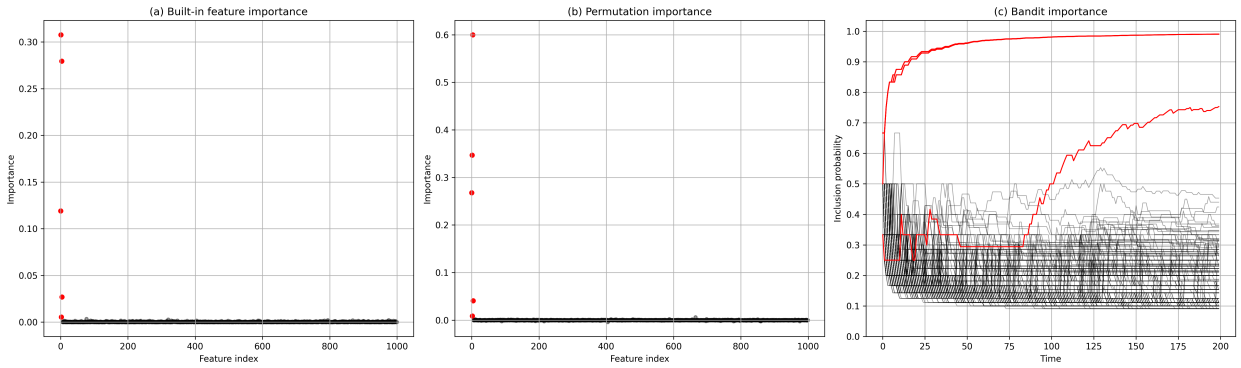

Figure 4: Feature importance or selection probabilities across three methods for equation 6. Left is built-in feature importance of random forest which tracks the average reduction in variance contributed by each feature. Middle is one-shot permutation importance ran on the fitted model. Right is selection probabilities of TTTS using permutation importance as reward.

hyperparameters misses out on both x_4 and x_5. On the other hand, bandit feature selection identifies x_1, x_2, x_3 and x_4 within 20 iterations, and manages to pick out x_5 after several more iterations. These results suggest that TTTS has the potential to attain higher accuracy than classical one-shot methods.

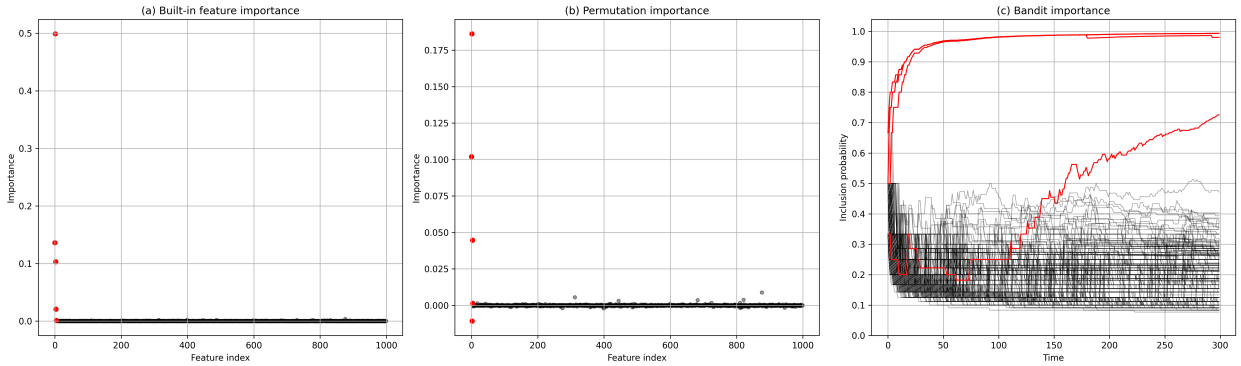

Figure 5: Selection probabilities across three methods for equation 7. Left is built-in feature importance of random forest which tracks the average reduction in variance contributed by each feature. Middle is one-shot permutation importance ran on the fitted model. Right is selection probabilities of TTTS using permutation importance as reward.

Additionally, under both settings, we compare the performance of TTTS to stability selection i.e. using the full set of features in every subsample of every iteration. The results for stability selection are similar to what we observe in one-shot methods with the same reward function definition, because it is essentially repeating one-shot methods over multiple runs, and in most iterations one-shot methods fail to pick out the weaker signals in the true DGP. TTTS outperforms under this setting because in the presence of fewer noise signals in latter iterations, the weak true features stand out and their influence become more conspicuous in the fitted model.

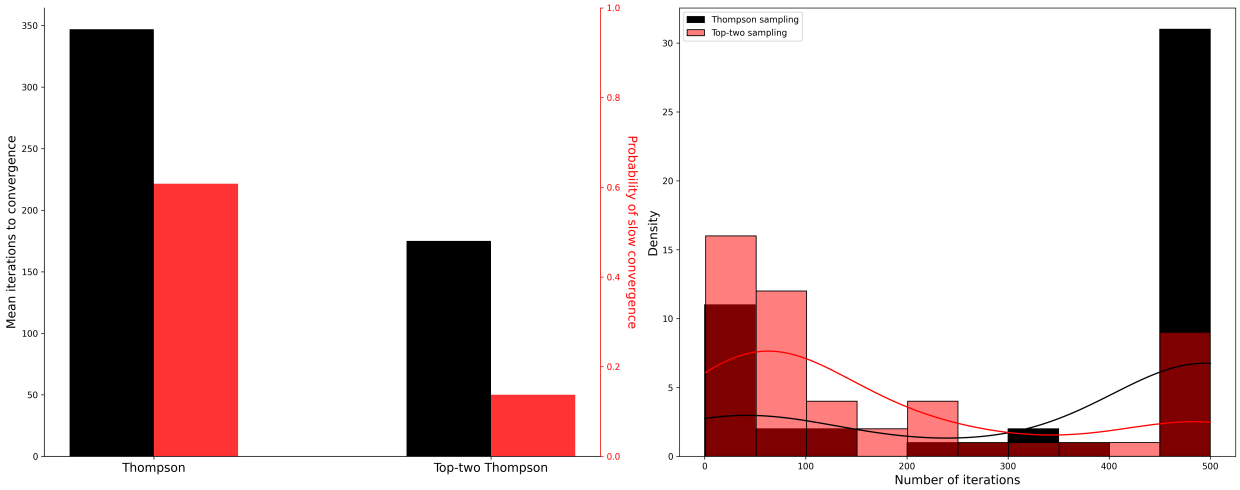

Figure 6: Comparing rate of convergence between TTTS and Thompson sampling using data generated from equation 6, with random forest as the base estimator. Left: left y-axis is the number of rounds to iteration while the right y-axis is the probability of not converging within 500 iterations. Right: the density plot of time taken to converge.

5.2 Comparing selection efficiency

In this subsection, we compare the number of iterations taken for TTTS to converge versus Thompson sampling. In the first experiment, we generate data from DGP 6 and use random forest as the base estimator. Repeating for 50 trials, we keep track of the number of iterations to convergence for each algorithm, and cap the maximum number of iterations at 500, and set $n = 300$ and $p = 500$. Convergence is defined as all true important variables having posterior inclusion probabilities above 0.5 and all noise variables having posterior inclusion probabilities below 0.5 for a pre-specified number of steps. We report the mean number of iterations taken for convergence and the probability of not converging within 500 iterations.

In Figure 6, focusing on the RHS panel which shows the density plots of the number of iterations to convergence, in 60% of the trials, Thompson sampling does not converge within 500 iterations, implying it is stuck in a local minimum that is difficult to escape. On the other hand, top-two Thompson sampling is much more robust to local minima and only less than 20% of the trials do not converge within 500 iterations. From both the bar plot and density plot, TTTS converges using many fewer iterations than Thompson sampling.

To demonstrate the robustness of this result i.e. the same conclusions hold when using other DGPs and base learners, in the second experiment, we generate data from equation 7 and use LASSO as the base estimator. The reason for picking LASSO is that the variables in 7 are additively separable. Despite being mis-specified, LASSO is still able to learn variable importance via bandit feature selection. Similarly, we observe lower probability of getting stuck in a local minimum and faster rate of convergence, as shown in Figure 7

The outperformance of TTTS originates from its tendency to explore variables with posterior mean of selection probability close to 0.5. By selecting the union minus the intersection of two super-arms, the variables that are more frequently selected are filtered out while the variables that are less likely to be selected are played.

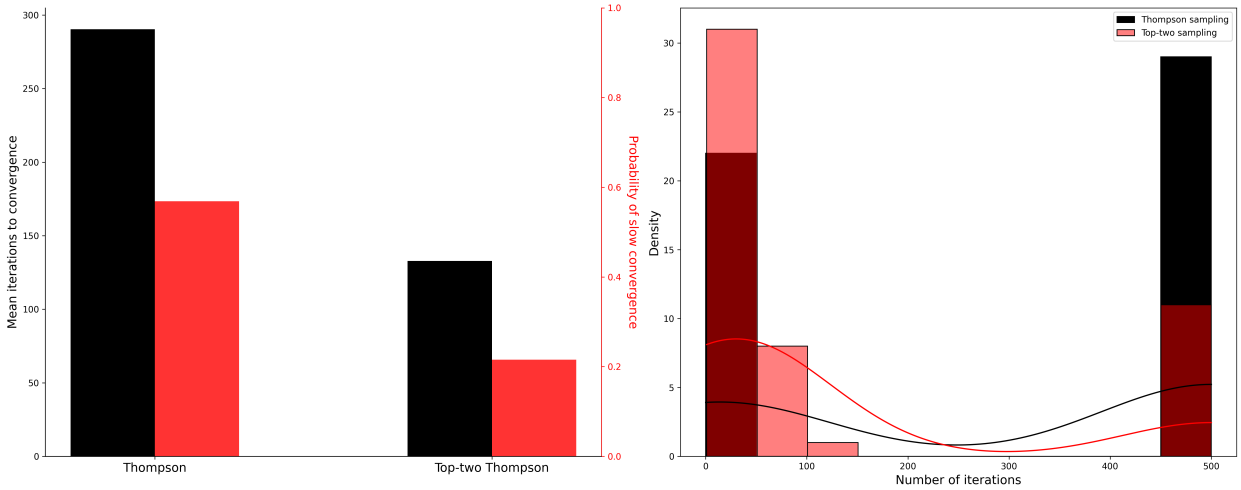

Figure 7: Comparing rate of convergence between TTTS and Thompson sampling using data generated from 7, with LASSO as the base estimator. Left: left y-axis is the number of rounds to iteration while the right y-axis is the probability of not converging within 500 iterations. Right: the density plot of time taken to converge.

5.3 Demonstrating online learning

So far, we have been focusing on offline learning, where a single dataset $\mathcal{D}$ is available for learning feature importance. Very often, practitioners are interested in online feature learning, where data comes in batches at a fixed time interval: $\mathcal{D}_1, \mathcal{D}_2 ... \mathcal{D}_n$. In every $\mathcal{D}_t$, either all features or a subset of features can be observed. In the case where all features are observed at every time step, online learning is analogous to offline learning with bootstrap, but the difference lies in that bootstrap produces the same data points as the original $\mathcal{D}$, while online learning receives new data points from the same DGP. Alternatively, the case where the observer only chooses to observe a subset of observations at every time step has crucial real-world relevance because observing all features might be expensive or impossible in some settings. Then, bandit feature selection offers a solution to learning feature importance under such scenarios.

In this section, we demonstrate the ability of bandit feature selection to learn feature importance from data arriving in batches. To study the robustness of TTTS for CMAB in adapting to different base learners, we consider three different DGPs, each using a different base estimator, to show the model-agnostic nature of the algorithm. The first two DGPs are from equation 6 and 7 from Friedman [1991], and the last DGP is a challenging case from Liang et al. [2018] as introduced in Liu and Rockova [2021].

The DGP in Liang et al. [2018] is as follows:

$$f(x_i) = \frac{10x_{i,2}}{1 + x_{i,1}^2} + 5\sin(x_{i,3}x_{i,4}) + 2x_{i,5} \tag{8}$$

$$y_i = f(x_i) + \epsilon_i \tag{9}$$

where $x_i = \frac{e_i + z_i}{2}$, and $\epsilon_i \sim N(0, 0.5)$, for $i = 1, ...n$. z_i is a $p \times 1$ vector whose every entry is i.i.d. $N(0, 1)$, and e_i is a $p \times 1$ vector constructed by first sampling a $\eta \sim N(0, 1)$

and then set every entry of e to be equal to η. This enables all features in x_i to be mutually correlated with a correlation coefficient of 0.5. The DGP is challenging not only because of its nonlinearity and feature interactions, but also the correlation between features which attenuates permutation importance.

We use gradient boosting regressor (GBR) for the first DGP, feedforward neural network (FFNN) for the second DGP, and random forest for the third DGP. GBR is an ensemble method like random forest (RF), but instead of building a set of trees in parallel, it builds a sequence of trees to minimize a loss function. Each next tree is fitted to the residuals of the previous tree:

$$F_m(x) = F_{m-1}(x) + \gamma h_m(x)$$

where x is the feature vector, $F_{m-1}(x)$ is the prediction based on all $m-1$ fitted trees so far, $h_m(x)$ is the fitted value of the m-th decision tree, and γ is the learning rate. It is a sequential process where at each step, a weak learner is trained on the residuals obtained from all previous weak learners.

FFNN is the most basic type of neural network where all the nodes are fully connected and information only flows in one direction. Suppose the first layer has J_1 nodes, the $J_1 \times 1$ vector $a^{(1)}$ is given by:

$$z^{(1)} = W_1^T x + b_1$$
$$a^{(1)} = \sigma(z^{(1)})$$

where W_1 is a $p \times J_1$ matrix of weights, b_1 is a $J_1 \times 1$ vector of biases (analogous to the intercept in linear regression), $\sigma(\cdot)$ is a nonlinear activation function applied element-wise to $z^{(1)}$. The vector of nodes in subsequent layers are given by the same iterative process, but with the vector from the previous layer as input. Essentially, FFNN takes the input, iteratively performs affine transformation and applies nonlinear transformation, to arrive at the output.

For GBR, we set `n_estimators` $= 100$, `max_depth` $= 3$, and `n_iter_no_change` $= 10$. For FFNN, we use two hidden layers with 64 and 32 neurons, `ReLU` activation, adaptive learning rate, and early stopping. For RF, we choose `n_estimators` $= 100$, `max_depth` $= 5$, and `min_samples_split` $= 2$. The choice of hyperparameters are arbitrary and based on heuristics.

Figure 8 presents the results. In both left and middle panels, as the DGPs are less challenging and the base estimators are powerful, we report only the first 50 iterations. We observe that selection probability of the true important variables quickly converge towards 1 while the noise variables are suppressed below 0.5. In the left panel, towards iteration 50, there is one noise variable that has selection probability rising above 0.5 temporarily due to inherent randomness in data, and it goes below 0.5 after several more iterations.

The right panel plots the selection probabilities for the third DGP which is highly challenging and we report 500 iterations. The rate of convergences of the five true important variables are ranked from fastest to slowest as: x_2, x_5, x_1, x_4, x_3. Since local reward is based on permutation importance which assesses the drop in out-of-sample predictability after permuting a feature, it is notable that the eventual selection probabilities represent the relative influence of the five true predictor on the outcome in terms of predictive performance. x_3 and x_4 interact with each other in a $\sin(\cdot)$ function, resulting in the smallest amount of

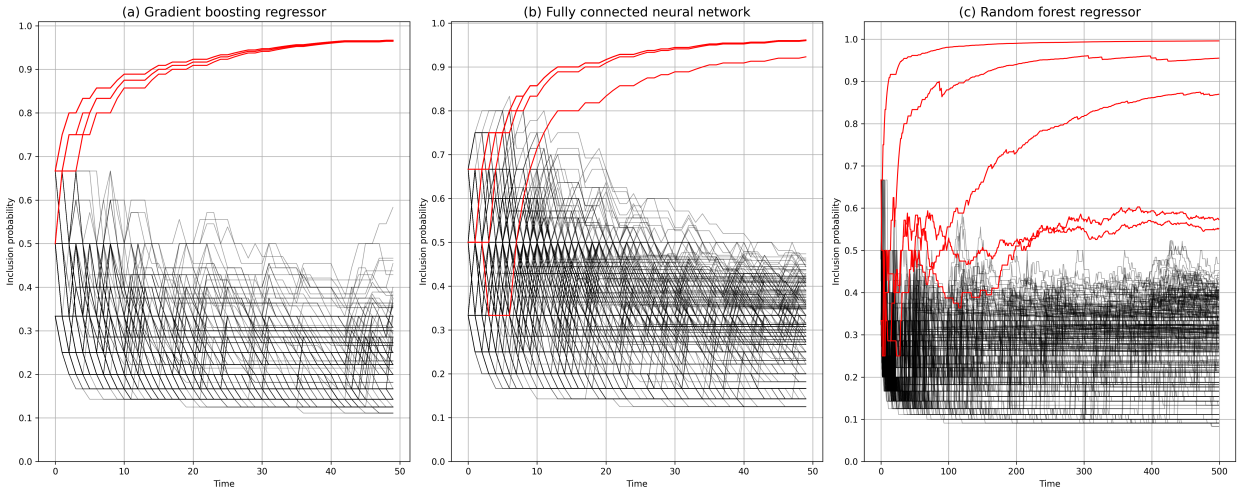

Figure 8: Selection probabilities over iterations for three different DGPs. Left and middle panel have uncorrelated features, while the right panel has features with correlation of about 0.5, hence the last DGP requires more iterations and the separation between signal and noise is not as distinct.

influence among all five predictors. Moreover, because x_3 is essentially indistinguishable from x_4 in the DPG, their selection probability trajectories are very similar. We observe that the selection probabilities of the noise features do not converge to 0 due to the feature correlation.

6 Empirical Studies

In this section, we apply bandit feature selection to the empirical asset pricing study conducted by Gu et al. [2020] to understand what variables are important for driving cross-sectional asset returns. Gu et al. [2020] compare the out-of-sample predictive performance of a wide range of mainstream machine learning algorithms used in both academia and industry today, including regularized regressions, tree-based methods, and neural networks, on the stock returns of over 30,000 publicly listed companies in the US between 1957 and 2016. They divide the data into training, validation and testing samples, and roll the split points forward in time iteratively to study the average predictability of each algorithm during the sample period. They find that the best performing algorithms are neural networks, gradient boosted trees and random forest, which attain an out-of-sample predictive R^2 of $0.34-0.40\%$ for monthly stock returns, and $3.09-3.60\%$ for annual stock returns.

Gu et al. [2020] also investigate the features that are crucial for returns predictability by setting all values of a feature of interest to 0 and calculating the reduction in predictive R^2. Then, they average this across all training samples to obtain an overall importance measure for each feature. We aim to compare the output of bandit feature selection against their results, using gradient boosted trees as the base estimator. The reason for choosing gradient boosted trees over random forest or neural networks is primarily because gradient boosted trees are computationally much more efficient than the other two methods given the volume of

data at hand. Moreover, for complicated datasets like stock returns, hyperparameter tuning plays a crucial role in the performance of neural networks which require the tuning of a myriad of parameters including the number of layers, number of nodes, learning rate, batch size, optimizer settings, and early stopping. In contrast, gradient boosted trees typically demand fewer tunable parameters, making them more practical and efficient in this application.

6.1 Data and method

As the number of observations is very large relative to the number of features in the full data sample, the high-dimensionality assumption that we have been assuming in the numerical experiments does not hold. As such, we run bandit feature selection in an online setting where data arrives at a monthly frequency.

In Gu et al. [2020], they compile a large collection of 94 stock level features covering characteristics related to momentum, liquidity, volatility, valuation, and fundamentals, covering every month from Mar 1957 to Dec 2016. Additionally, they construct eight macroeconomic predictors following the definitions given in Welch and Goyal [2008]. These are the variables they use for predicting cross-sectional stock returns. All explanatory variables are lagged by 1 month at least relative to stock returns.

We download the firm level features dataset which is updated till Jun 2021 from Xiu's personal website. Since the macroeconomic predictors are updated on a monthly basis and take on the same value across all observations within a cross section, we do not include the macroeconomic predictors in my online setting. Finally, we download stock returns data from CRSP for all firms in the Gu et al. [2020] dataset, and risk free rate from Kenneth French's Data Library to compute excess return.

To give an overview of our data preprocessing steps, first, following Gu et al. [2020], rather than using the raw stock feature values, we rank each stock feature and map the ranks into the interval $[-1, 1]$ to facilitate the optimization of the machine learning algorithms. Second, while Gu et al. [2020] generate dummy variables for the industry membership of each firm and feature interactions, we skip this step because the `HistGradientBoostingRegressor` class in `sklearn` is able to recognize the industry variable as a categorical variable and gradient boosting trees can account for the interactive effects of features by increasing model complexity without having to explicitly create interactions. Third, instead of running the full sample from 1957 to 2021, during which the significant drivers of stock returns might have changed drastically due to variations in the underlying economic dynamics and market environment, we take a subset of 10 years of the full sample from 2010 - 2019, the period after the Great Recession and before the COVID19 recession, during which drivers of cross sectional stock returns are likely to remain stable, and are more relevant to the present day markets. Finally, even when running the dataset with online setting, every monthly cross section still has over 5000 stocks on average, but only 94 stock level features and 74 industry memberships. Hence, in each cross section, we generate 5000 noise variables from $N[0, 1]$, most values of which fall in the interval $[-1, 1]$, to artificially enforce the dimensionality to be similar to the number of observations in each monthly dataset. This gives us a dataset with over 5000 stock returns and over 5000 features arriving each month.

As returns prediction is a challenging problem and predictability is very small as seen from miniscule out-of-sample R^2, using an arbitrary set of hyperparameters might yield

predictive R^2 very close to 0 or even negative R^2. Therefore, we use the first month i.e. Jan 2010 for hyperparameter tuning. The key parameters are learning rate, maximum number of trees, maximum number of leaf nodes per tree, maximum depth per tree, minimum number of samples required for split, and amount of L2 regularization. After obtaining the optimal hyperparameters by gridsearch and k-fold cross validation, we use the same set of hyperparameters for all subsequent months.

6.2 Empirical Results

We run bandit feature selection for 600 iterations, i.e. 5 iterations for each of the 120 months between 2010 and 2019, and set local reward threshold to be a 2% relative drop in out-of-sample fit measured by R^2 when a feature is permuted. Panel (a) in figure 9 shows the change in selection probabilities over time, while panel (b) shows the features corresponding to the top 25 inclusion probabilities. We observe that 6 features have selection probabilities above 0.6 most of the times: industry momentum (indmom), bid-ask spread (baspread), 12-month momentum (mom12m), sensitivity to market factor (beta), 1-month momentum (mom1m), and idiosyncratic return volatility (idiovol). All of these variables are highly crucial cross sectional return predictors in the finance literature, and coincide with the ranking of variable importance by Gu et al. [2020]. They identify all six aforementioned predictors to have strong predictability in all of their machine learning algorithms (with the only exception that baspread and beta are unimportant for partial least squares and principal component regression).

Next, we observe more than 20 features to have selection probabilities fluctuating between 0.6 and 0.5 over time. All of these features, with the exception of cash holdings (cash) and volatility of liquidity in terms of dollar trading volume (std_dolvol), coincide with the set of features identified to have strong predictability by Gu et al. [2020]. Note that the ranking of feature importance in Gu et al. [2020] is the average importance across all statistical models. In our application, although we only use GBR as the estimator, we manage to identify features that are deemed unimportant for GBR but important for other algorithms in Gu et al. [2020], suggesting that bandit feature selection may be able to learn the truly important variables in the actual DGP, rather than just the important variables in the context of the model.

Panel (a) also shows the inclusion probability trajectories of the artificial noise variables, all 5000 of which are hovering between 0.1-0.4 since around 50 iterations. This indicates that bandit feature selection is highly effective in identifying the noise variables from signals and suppressing the probability of picking these variables again in future iterations.

There are two comments worth highlighting. First, our ranking of variable importance does not align exactly with the ranking of variable importance in Gu et al. [2020]. This could be because they use the full sample which is much larger than my subset of samples and it is highly likely that cross sectional drivers of stock returns have changed considerably over the decades. Second, our results are similar to Gu et al. [2020] in the sense that while several predictors like indmom, mom1m, mom12m, mvel1 and retvol have posterior inclusion probabilities above 0.55, a lot more other predictors have inclusion probabilities hovering close to 0.5. This can be explained by the fact that in predicting cross sectional stock returns, there are a few strong predictors and many more mediocre but nevertheless

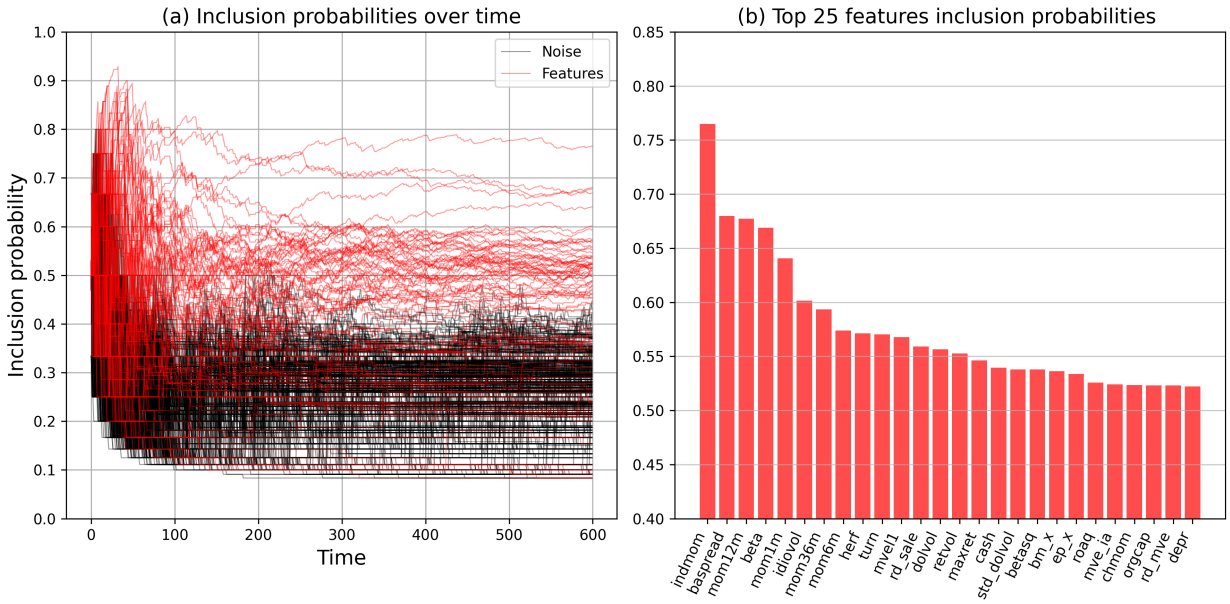

Figure 9: Left: selection probabilities over time. Right: Features corresponding to top 25 selection probabilities.

useful predictors, each of which explains a tiny bit of cross sectional returns, as implied by the variable importance results of Gu et al. [2020]. Re-running this set-up using a more lenient threshold, e.g. assigning local reward to a 1% relative drop in predictive R^2 when a feature is permuted, achieves a better separation of signal from noise. A lower threshold might be preferred if there is a large number of weak predictors.

6.3 Practical considerations

This section discusses the rationale behind certain design choices of bandit feature selection, as well as potential drawbacks and ways to overcome them. First, permutation importance is unable to identify true features that are highly correlated with each other or with other noise features. This is because for highly correlated features, permuting one feature does not reduce predictive performance of the model much given the presence of the other feature. From section 5.3, we see that bandit feature selection is able to handle nonlinear DGPs involving features with correlations of 0.5. If correlation goes to a larger value like 0.8 or above, permutation importance as a reward function might not work well.

In our application, Figure 11 shows the correlation matrix of all 94 firm level features in Gu et al. [2020] in our subset sample. In all $94C2 = \frac{94\times93}{2} = 4371$ pairs of variables, less than 15 pairs have absolute value of correlation coefficient greater than 0.75, and most pairs have absolute correlations well below 0.25, as such the conclusions in section 6.2 are unlikely to suffer from correlated variables. Additionally, in most of the highly correlated pairs, the features are identified to be unimportant by Gu et al. [2020]. The only four exceptions, i.e. highly correlated pairs of important variables, are (baspread, retvol), (baspread, idiovol), (maxret, retvol), (dolvol, mvel1). Nevertheless, they are identified to have high selection

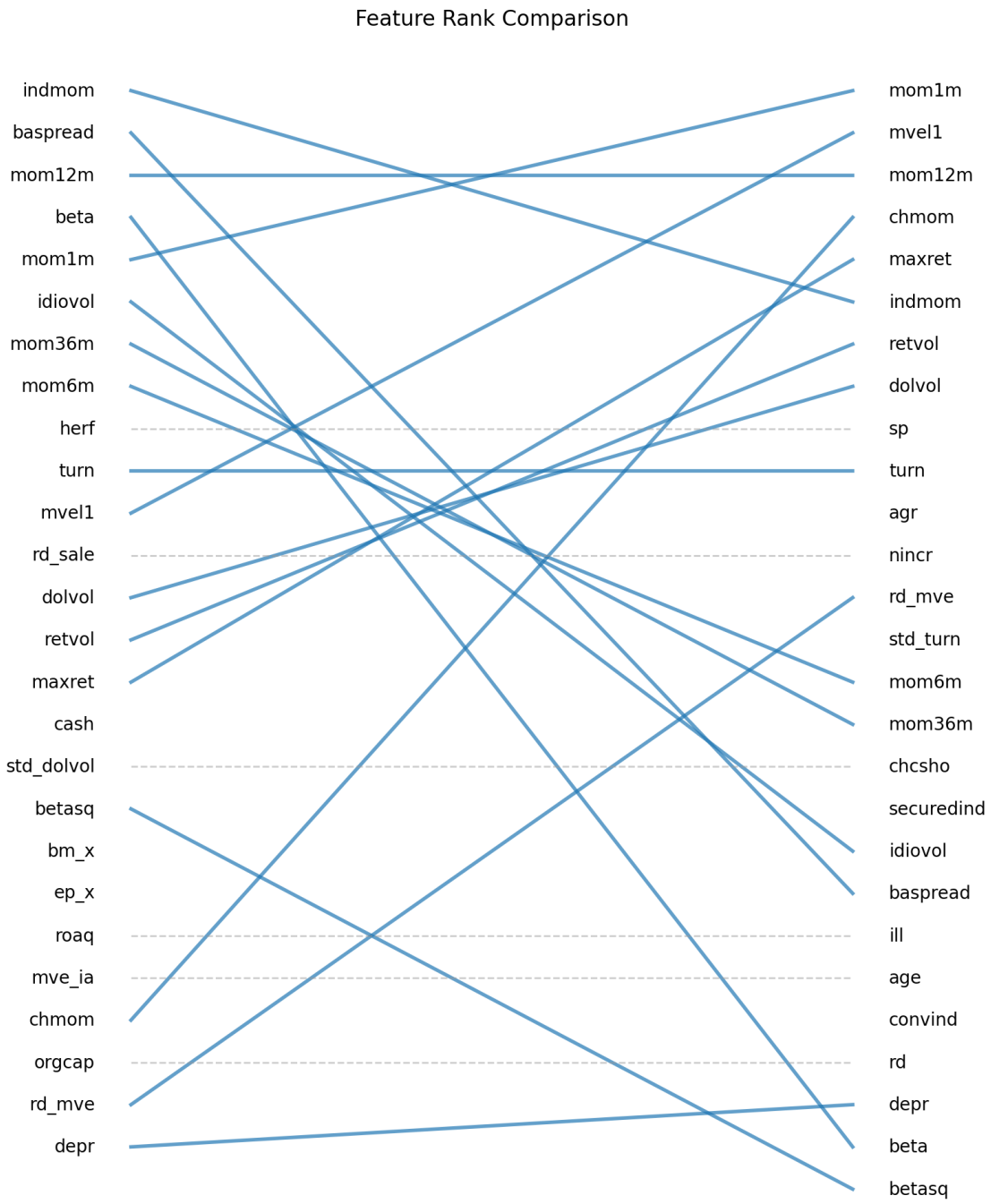

Figure 10: Left: selection probabilities in our model ranked in descending order. Right: feature importance in Gu et al. [2020] ranked in descending order. There is significant overlap between the two sets of features.

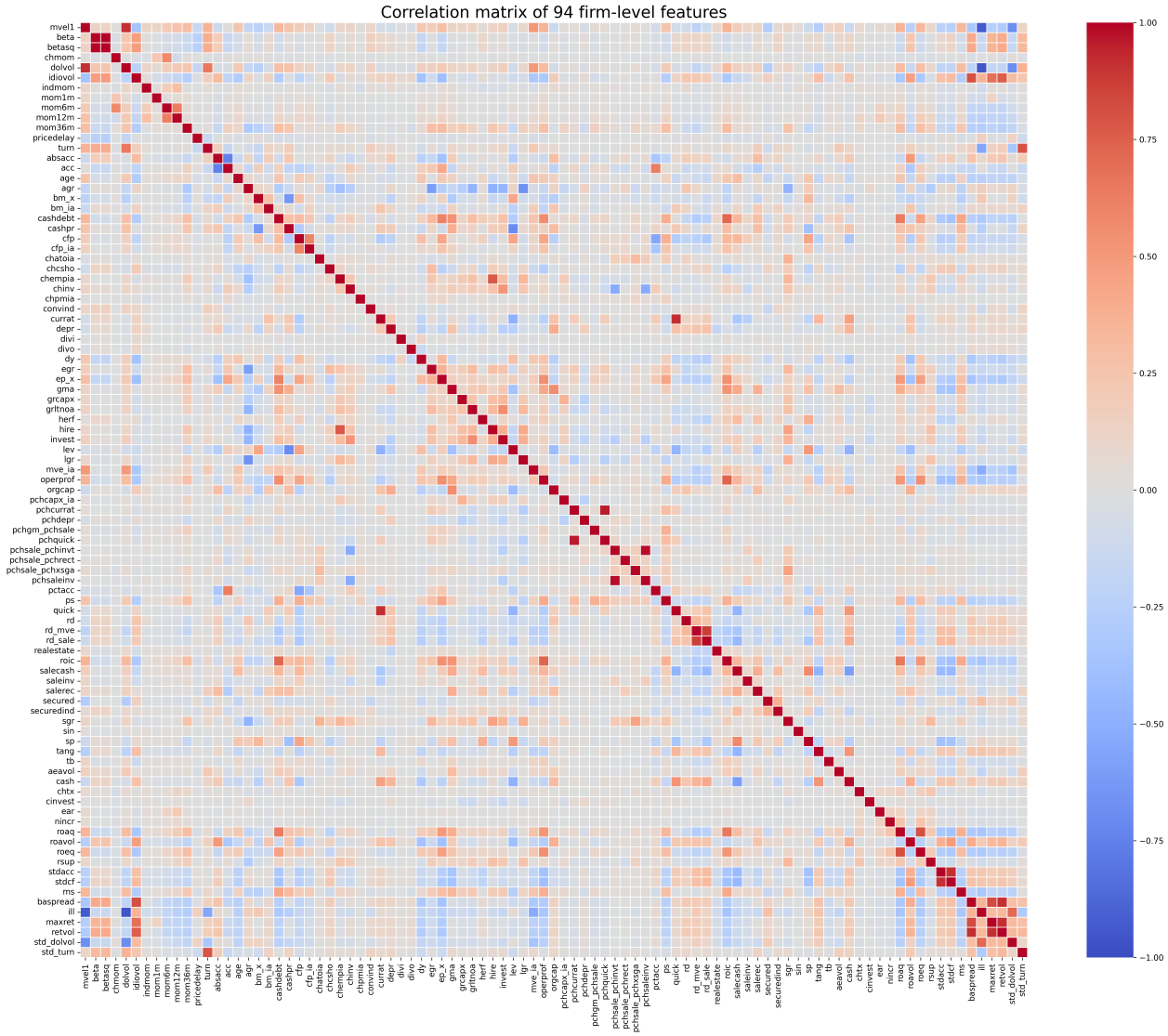

Figure 11: Correlation matrix of 94 firm level features in the empirical application.

probabilities by bandit feature selection, most likely because in some subsets at certain time steps, only one variable in the pair is selected, enabling permutation importance to assign it positive local reward.

One solution to overcome this limitation is to bundle the highly correlated features and permute the entire bundle when assigning local reward. This requires an additional step in the bandit algorithm which constructs the correlation matrix after having selected the super-arm at time t, and either bundles the correlated variables as one block or performs dimensionality reduction such as principal component analysis to end up with a single variable for computing permutation importance.

Next, Molnar et al. [2023] formulate the theory on permutation importance by drawing connection to the underlying DGP. Recall the PI formula in section 5, in actual computation, $f(\cdot)$ is approximated by $\hat{f}(\cdot)$ and expectation is approximated by Monte Carlo integration.

This implies that true PI based on the DGP is biased by the errors in model approximation and Monte Carlo approximation. Molnar et al. [2023] show that if the model estimator is unbiased i.e. $E(\hat{f}) = f$, then sample PI is an unbiased estimator of population PI. Moreover, they show that the gap between population PI and sample PI is a function of the expected variance of f and covariance of f and $\hat{f}$. In light of this, they propose a sample PI computed from averaging multiple model refits and corrected confidence intervals.

Bandit feature selection has a similar flavor as the refitting PI estimator of Molnar et al. [2023]. In the offline setting, it computes PI using bootstrapped data and different sets of features at every time step to iteratively correct the sample PI which is affected by bias and data randomness, to arrive at a more accurate importance estimate eventually.

The theorems in Molnar et al. [2023] might offer theoretical backing to why using a base estimator with a mis-specified functional form (e.g. using LASSO to learn nonlinear additive DGP in section 5.2) allows feature selection probabilities to be learned, and why hyperparameter tuning at every time step is not necessary for learning posterior selection probabilities. Since the gap between sample PI and population PI is bounded by the covariance between f ad $\hat{f}$, when this quantity is small, sample PI does not deviate too far from population PI. Moreover, for nonparametric machine learning algorithms, specifying a more complex functional form tends to increase variance and reduce bias As such, when specifying hyperparameters of a nonparametric base estimator, it is recommended to go for a moderately complex model to reduce the bias of sample PI, and the point estimate can be made more accurate via iterative learning.

Intuitively, when hyperparameters are not optimal and the estimator overfits to training data, because PI is evaluated on out-of-sample data, noise variables do not impact out-of-sample fit anyways while true important variables will have influence, albeit small. If computation is not a major constraint, it would also be possible to conduct hyperparameter tuning at every time step within the bandit feature selection algorithm.

Additionally, one drawback of bandit feature selection is that it does not have a good convergence criterion, i.e. when to stop the algorithm when running on real-world dataset. One potential solution is to monitor the selection probabilities graph at fixed time intervals, and stop when the practitioner observes stability or a clear separation in the selection probabilities. In our Python implementation, we define a stopping condition i.e. the ranking of all variables with selection probabilities above 0.5 remains unchanged for K iterations, where K is set to be 50 as default.

Finally, one should note that when implementing bandit feature selection on data with temporal ordering, because bandit feature selection involves bootstrap and row permutation, it is essential to preserve the temporal order of dataset when fitting estimators and computing PI to avoid look-ahead bias.

7 Discussion

In this paper, we study the limitations of bandit-based feature selection and under the strong identifiability assumption, we propose a top-two Thompson sampling algorithm for CMAB and investigate its effectiveness for recovering true feature set through both simulations and an empirical application. Bandit-based feature selection is suitable for settings where

full model evaluation is computationally expensive or infeasible, especially in many modern applications involving high-dimensional data. Bandit algorithms offer a natural framework for adaptive allocation of computational resources, shifting exploration to more promising regions of the full feature space.

This adaptive feature selection is also useful in cases where signal strength is heterogeneous. Weak but relevant features may be difficult to detect using non-adaptive methods under limited sample sizes. Bandit-based approaches address this issue by sequentially focusing less on the noise features and making it possible to identify weak informative features. Additionally, bandit-based approach fits naturally in the online learning environment where data arrive sequentially over time.

However, bandit-based method face several limitations. First, the performance of bandit selection is only as good as the quality of the base learner. Suppose the true DGP is complex but the base learner is simple and misspecified, it is not clear to what extent running iterative subset selection using bandit can recover the true feature set. The effect of the choice of the base learner and of model misspecification on consistency awaits investigation under different DGP settings.

Second, with regards to TTTS for CMAB, it remains an open question to check whether TTTS attains exponential rate of posterior convergence in t, how to compute the optimal exploration parameter δ conditioning on the posterior distribution in that iteration, to what extent fixing $\delta = 0.5$ deviates from the optimal exponent, and how to efficiently tune the exploration parameter in actual implementation.

Finally, the strong identifiability condition is a high-level blanket assumption and depends on not only signal strength in the DGP but also the choice of base learner and reward function. It will be valuable to establish theoretical guarantees on what combinations of DGP, learner, and reward function give rise to the strong identifiability condition, in order to understand bandit-based feature selection under the hood. Also, it would be interesting to explore other conditions that apply more directly to DGP or base learner that can guarantee the consistency of adaptive feature selection methods.

References

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. *JMLR: Workshop and Conference Proceedings*, 23:39.1–39.26, 2012.

Andre Altmann, Laura Tolosi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *BIOINFORMATICS*, 26(10):1340–1347, 2010.

Mohammad-Hassan Zokaei Ashtiani, Majid Nili Ahmadabadi, and Babak Nadjar Araabi. Bandit-based local feature subset selection. *Neurocomputing*, 138:371–382, 2014. doi: http://dx.doi.org/10.1016/j.neucom.2014.02.001.

Andre Beinrucker, Ürün Dogan, and Gilles Blanchard. Extensions of stability selection using subsamples of observations and covariates. *Stat Comput*, 26:1059–1077, 2016.

Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

Sebastien Bubeck, Remi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. *International Conference on Algorithmic Learning Theory*, pages 23–47, 2009.

Sebastien Bubeck, Tengyao Wang, and Nitin Viswanathan. Multiple identifications in multi-armed bandits. *Proceedings of the 30th International Conference on Ma- chine Learning, JMLR*, 28, 2013.

A. Stefano Caria, Grant Gordon, Maximilian Kasy, Simon Quinn, Soha Shami, and Alexander Teytelboym. An adaptive targeted field experiment: Job search assistance for refugees in jordan. *Journal of the European Economic Association*, 2023. doi: https://doi.org/10.1257/rct.3870-2.2.

Jiecao Chen, Xi Chen, Qin Zhang, and Yuan Zhou. Adaptive multiple-arm identification. *Proceedings of the 34th International Conference on Machine Learning, PMLR*, 70, 2017.

Audrey Durand and Christian Gagné. Thompson sampling for combinatorial bandits and its application to online feature selection. *AAAI Workshop: Sequential Decision-Making with Big Data*, 2014.

Jerome H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19 (1):1–67, 1991.

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine1. *The Annals of Statistics*, 29(5):1189–1232, 2001.

Aurelien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. *JMLR: Workshop and Conference Proceedings*, 49:1–30, 2016.

Romaric Gaudel and Michele Sebag. Feature selection as a one-player game. *International Conference on Machine Learning*, pages 359–366, 2010.

Peter Glynn and Sandeep Juneja. Selecting the best system and multi-armed bandits. *arXiv preprint arXiv:1507.04564*, 2018.

Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33:2223–2273, 2020. doi: 10.1093/rfs/hhaa009.

Maximilian Kasy and Anja Sautmann. Adaptive treatment assignment in experiments for policy choice. *Econometrica*, 89(1):113–132, 2021. doi: https://doi.org/10.3982/ECTA17527.

Faming Liang, Qizhai Li, and Lei Zhou. Bayesian neural networks for selection of drug sensitive genes. *Journal of the American Statistical Association*, pages 955–972, 2018. doi: 10.1080/01621459.2017.1409122.

Kunpeng Liu, Haibo Huang, Wei Zhang, Ahmad Hariri, Yanjie Fu, and Kien Hua. Multi-armed bandit based feature selection. *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 316–323, 2021. doi: https://doi.org/10.1137/1.9781611976700.36.

Yi Liu and Veronika Rockova. Variable selection via thompson sampling. *Journal of the American Statistical Association*, 00(0):1–18, 2021. doi: 10.1080/01621459.2021.1928514.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *31st Conference on Neural Information Processing Systems*, 2017.

Nicolai Meinshausen and Peter Bühlmann. Stability selection. *J. R. Statist. Soc. B*, 72: 417–473, 2010.

Christoph Molnar, Timo Freiesleben, Gunnar Konig, Julia Herbinger, Tim Reisinger, Guiseppe Casalicchio, Marvin N. Wright, and Bernd Bischl. Relating the partial dependence plot and permutation feature importance to the data generating process. *Explanable Artificial Intelligence CCIS 1901*, pages 456–479, 2023.

Shintaro Nakamura and Masashi Sugiyama. Thompson sampling for real-valued combinatorial pure exploration of multi-armed bandit. *The Thirty-Eighth AAAI Conference on Artificial Intelligence*, 2024.

Sali Rasoul, Sodiq Adewole, and Alphonse Akakpo. Feature selection using reinforcement learning. *arXiv preprint arXiv:2101.09460*, 2021.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016. doi: http://dx.doi.org/10.1145/2939672.2939778.

Benedek Rozemberczki, Lauren Watson, Peter Bayer, Hao-Tsung Yang, Oliver Kiss, Nilsson Sebatstian, and Sarkar Rik. The shapley value in machine learning. *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, 2022.

Daniel Russo. Simple bayesian algorithms for best arm identification. *29th Annual Conference on Learning Theory, PMLR*, 49, 2016.

Rajen D. Shah and Richard J. Samworth. Variable selection with error control: another look at stability selection. *J. R. Statist. Soc. B*, 75:55–80, 2012.

Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2.28: 307–317, 1953.

Jialei Wang, Peilin Zhao, Steven C.H. Hoi, Member IEEE, and Rong Jin. Online feature selection and its applications. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING,*, 26(3):698–710, 2014. doi: 10.1109/TKDE.2013.32.

Siwei Wang and Wei Chen. Thompson sampling for combinatorial semi-bandits. *Proceedings of the 35th International Conference on Machine Learning, PMLR*, 80, 2018.

Siwei Wang and Jun Zhu. Thompson sampling for (combinatorial) pure exploration. *Proceedings of the 39th International Conference on Machine Learning, PMLR*, 162, 2022.

Ivo Welch and Amit Goyal. A comprehensive look at the empirical performance of equity premium prediction. *The Review of Financial Studies*, pages 1455–1508, 2008. doi: https://doi.org/10.1093/rfs/hhm014.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

Appendix

A LASSO Bandit with Time-varying Penalty

In Section 3, we compare the support recovery performance of stability selection and bandit selection using a fixed regularization parameter λ across iterations. One might think this is not optimal since the size of the feature subset is not constant over time, and tends to shrink as noise features are excluded. This means the optimal level regularization may also change over time. In this appendix section, we study whether allowing λ to adapt over time improves the accuracy of support recovery of bandit selection.

We choose optimal λ by k-fold cross validation (CV), minimizing out of sample MSE on the test fold, which is the standard practice in real-world applications. We run the simulations under the simple isotropic X DGP here, with training sample size 200, and total number of features 1000, among which 10 are true signals and 990 are noise features.

In Figure 12, the left panel shows the evolution of selection probabilities over iterations using optimal cross validation λ in each iteration. Despite being adaptive, we see that many noise features have high selection probabilities. The false positive rate is actually higher than using a fixed λ in Figure 1. The reason for this is that cross validation chooses λ to minimize prediction error, and λ that is good for prediction is often too small for good variable selection since given limited training data, noise features might provide some predictability. As such, LASSO keeps too many variables and bandit selection rewards too many noise arms.

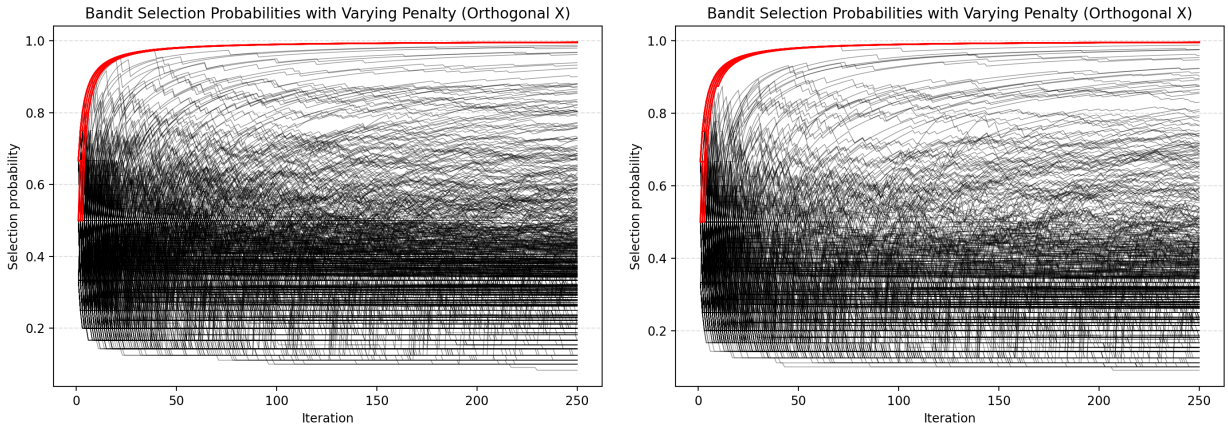

Figure 12: Selection probability against iteration for k-fold CV λ (left) and +1 s.e. k-fold CV λ (right). Red is true feature while black is noise. False positive rate is reduced slightly from using +1 s.e. λ but remains high.

With that knowledge, we consider a second simulation where we do not use the optimal k-fold CV λ, but the CV λ +1 s.e., to artificially make λ larger. It is computed by first calculating a threshold defined as the mean plus the standard error of test errors across the folds for the optimal CV λ. Then, we choose the largest λ in the grid with mean test error below the threshold.

Figure 13 shows the optimal CV λ chosen in each iteration in the left panel, and the optimal CV λ +1 s.e. in the right panel. There are two notable patterns. First, optimal

λ is on a decreasing trend over iterations, i.e. smaller penalty is chosen. This is because bandit selection excludes the noise features and selects a smaller feature subset over time, hence requiring a smaller penalty. Second, the λ's on the RHS are uniformly larger than those on the LHS, indicating larger penalty in each iteration by adding one standard error to the mean MSE of the optimal CV λ.

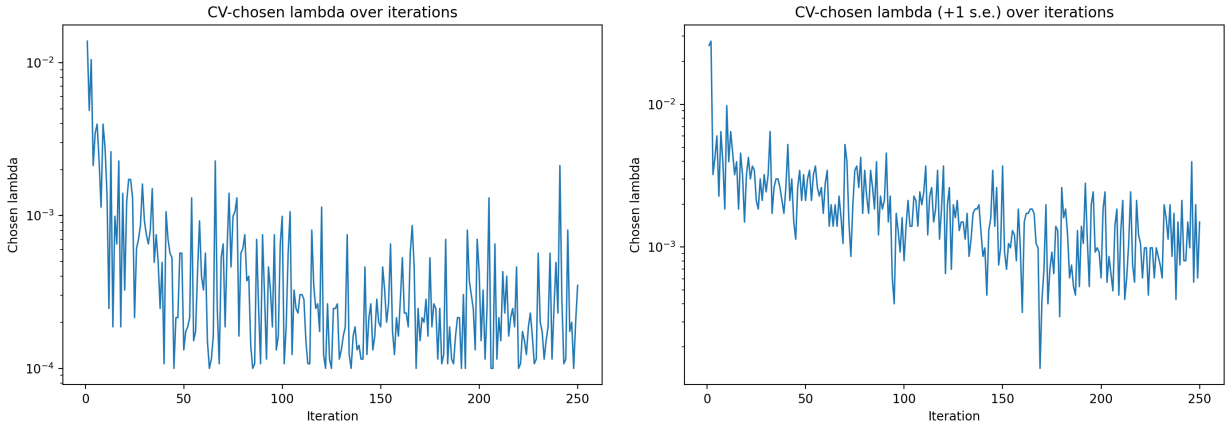

Figure 13: K-fold CV λ against iteration for without +1 s.e. (left) and with +1 s.e. (right). The chosen λ decreases over iterations as subset played becomes smaller. Plus 1 s.e. λ are notably larger than without plus 1 s.e. in every iteration.

The right panel of Figure 12 reports the resulting selection probabilities when using CV λ +1 s.e.. False positive rate improves over using optimal CV λ as expected due to larger penalty in each iteration, but remains very high. Noises would have been indistinguishable from features in real-world applications.

These findings undermine the applicability of LASSO bandit in practice. When λ is chosen by cross validation, the learner tends to include too many noise variables, leading to poor support recovery. Choosing λ adaptively becomes a challenging question. One fix is to consider using a fixed λ and run the algorithms in Section 3 over a grid of values, but this approach still underperforms stability selection. As such, the choice of penalty remains a critical problem in LASSO bandit selection in actual applications.

B Proofs for Results in Section 3

B.1 Proof for Corollary 3.1

This corollary extends from Proposition 1 of Zhao and Yu [2006]. In that proposition, under the setting we describe in subsection 3.2, they show that when IC holds with a constant vector $\eta > 0$, then the probability of correct sign identification is bounded below by:

$$P(sign(\hat{\beta}(\lambda)) = sign(\beta)) \geq P(A_n \cap B_n) \tag{10}$$

where

$$A_n = \{|(C_{11}^n)^{-1}W^n(1)| < \sqrt{n}(|\beta_{(1)}^n| - \frac{\lambda}{2n}|(C_{11}^n)^{-1}sign(\beta_{(1)}^n)|)\} \tag{11}$$

$$B_n = \{|C_{21}^n(C_{11}^n)^{-1}W^n(1) - W^n(2)| \leq \frac{\lambda}{2\sqrt{n}}\eta\} \tag{12}$$

and

$$W^n(1) = \frac{1}{\sqrt{n}}X_n(1)^T\epsilon_n, \quad \frac{1}{\sqrt{n}}W^n(2) = X_n(2)^T\epsilon_n \tag{13}$$

Note that the event A_n implies that the signs of the coefficient estimates of the true features are estimated correctly, while conditioning on A_n, B_n implies that the coefficient estimates of the noise features are shrunk to zero.The effect of dropping noise features on the IC $|C_{21}^n(C_{11}^n)^{-1}sign(\beta_{(1)}^n)|$ is removing rows from C_{21}^n. This results in fewer number of inequality constraints to be satisfied and hence η will have fewer elements. Next, η does not appear in the event A_n and only affects B_n. Dropping noise features i.e. removing rows from C_{22}^n and η, and columns from $X_n(2)$ reduce the number of inequalities in B_n and does not alter the remaining inequalities. Hence, B_n cannot become smaller and $P(A_n \cap B_n)$ can only increase.

B.2 Proof for Corollary 3.2

This corollary also directly follows from Proposition 1 of Zhao and Yu [2006]. Consider a design matrix with 2 signals and 1 noise feature, and the covariance of noise with the two features is σ and $-\sigma$ with $\sigma > 1$ while the two features are uncorrelated with a variance of 1 each, so $C_{21}^n = [\sigma, -\sigma]$ and $C_{11}^n = I_2$. Let $\beta_{(1)}^n = [1, 1]^T$. Then, with both true features, the IC is satisfied with $\eta = [1, 1]^T$. But dropping either one of the true features and retaining the other violates the IC.

B.3 Proof for Proposition 3.1

To show this proposition, we construct a counterexample where dropping some true features does not violate the IC but results in a smaller probability lower bound of correct sign identification. Suppose the full model is

$$y_i = x_{1,i} + x_{2,i} + \epsilon_i$$

and there is a single noise feature $x_{3,i}$ and $x_1 \perp x_2$. $x_1 = [2, 0, 2, 0]$, $x_2 = [2, 2, 0, 0]$ and $x_3 = [2, 0, 2, 2]$. We first compute the correct sign identification probability with all three features, and then using just x_1 and x_3.

With all three features,

$$\Sigma_{2,1} = \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & -0.5 \\ 0.5 & -0.5 & 0.75 \end{pmatrix}$$

and checking that IC holds with $\eta = 1$:

$$|(0.5, -0.5) \begin{pmatrix} 1 \\ 1 \end{pmatrix}| = 0$$

Then, the event

$$A_n = \left\{ | \begin{pmatrix} 2 & 0 & 2 & 0 \\ 2 & 2 & 0 & 0 \end{pmatrix} \varepsilon | \leq \begin{pmatrix} n - \lambda/2 \\ n - \lambda/2 \end{pmatrix} \right\}$$

and the event

$$\begin{aligned} B_n &= \left\{ | \left(\begin{pmatrix} 0.5 & -0.5 \end{pmatrix} \begin{pmatrix} 2 & 0 & 2 & 0 \\ 2 & 2 & 0 & 0 \end{pmatrix} - n \begin{pmatrix} 2 & 0 & 2 & 2 \end{pmatrix} \right) \varepsilon | \leq \frac{\lambda}{2} \right\} \\ &= \left\{ | (\begin{pmatrix} 0 & -1 & 1 & 0 \end{pmatrix} - \begin{pmatrix} 8 & 0 & 8 & 8 \end{pmatrix}) \varepsilon | \leq \frac{\lambda}{2} \right\} \\ &= \left\{ | \begin{pmatrix} -8 & -1 & -7 & -8 \end{pmatrix} \varepsilon | \leq \frac{\lambda}{2} \right\} \end{aligned}$$

And $P(A_n \cap B_n)$ can be written as

$$\begin{aligned} P(A_n \cap B_n) &= \left\{ | \begin{pmatrix} 2 & 0 & 2 & 0 \\ 2 & 2 & 0 & 0 \\ -8 & -1 & -7 & -8 \end{pmatrix} \varepsilon | \leq \begin{pmatrix} 4 - \lambda/2 \\ 4 - \lambda/2 \\ \frac{\lambda}{2} \end{pmatrix} \right\} \\ &:= \{ |A\varepsilon| \leq b \} \\ &:= \{ B \leq b \} \end{aligned}$$

Since features are fixed and ε is the only source of randomness here, assuming ε follows a multivariate standard normal distribution $N(0, I_4)$, then $B \sim N(0, AA^T)$ and the above probability can be evaluated numerically to be 0.04263 with λ set to be 2.

Next, dropping x_2, the same calculation can be performed. First, IC holds with $\eta = 0.5$:

$$|(0.5) \begin{pmatrix} 1 \end{pmatrix}| = 0.5$$

Then, the event

$$A_n = \left\{ | \begin{pmatrix} 2 & 0 & 2 & 0 \end{pmatrix} \varepsilon | \leq \begin{pmatrix} n - \lambda/2 \end{pmatrix} \right\}$$

and the event

$$B_n = \left\{ | \left((0.5) \begin{pmatrix} 2 & 0 & 2 & 0 \end{pmatrix} - n \begin{pmatrix} 2 & 0 & 2 & 2 \end{pmatrix} \right) \varepsilon | \leq \frac{\lambda}{2} \eta \right\}$$

$$= \left\{ |((1 \quad 0 \quad 1 \quad 0) - (8 \quad 0 \quad 8 \quad 8))\varepsilon| \leq \frac{\lambda}{4} \right\}$$
$$= \left\{ |\begin{pmatrix} -7 & 0 & -7 & -8 \end{pmatrix} \varepsilon| \leq \frac{\lambda}{4} \right\}$$

And $P(A_n \cap B_n)$ can be written as

$$\begin{aligned} P(A_n \cap B_n) &= \left\{ |\begin{pmatrix} 2 & 0 & 2 & 0 \\ -7 & 0 & -7 & -8 \end{pmatrix} \varepsilon| \leq \begin{pmatrix} 4 - \lambda/2 \\ \frac{\lambda}{4} \end{pmatrix} \right\} \\ &:= \{|G\varepsilon| \leq h\} \\ &:= \{H \leq h\} \end{aligned}$$

And this probability can be evaluated numerically to be 0.02846 with the same $\lambda = 2$, which concludes the proposition.

C Proof for Proposition 4.1

First, we introduce the CMAB for feature selection setting. The set of base arms is $[K] = \{1, ..., K\}$, and define S^* as the superarm that maximizes mean expected reward i.e. $S^* := argmax_S E[R_t(S)]$. The bandit model is, at each t, TTTS chooses a $S_t \in 2^{[K]}$.

For each $i \in S_t$, $r_t^k \sim Ber(p_{k,S_t})$ where $p_{k,S_t} = \mathbb{P}(r_t^k = 1|k, S_t)$. Each arm has independent $\theta_k \sim Beta(\alpha_{k,0}, \beta_{k,0})$ prior.

Define $N_t^k := \sum_{j=1}^t \mathbf{1}\{k \in S_j\}$ as the number of times arm k was played up till time t, define $X_t^k := \sum_{j \le t, k \in S_j} r_t^k$, and $\mathcal{F}_t := \{S_1, (X_1^k)_{k=1}^K, ..., S_t, (X_t^k)_{k=1}^K\}$ as the history up till the end of time t, then the posterior selection probability at the end of round t or the start of round $t+1$ is given by:

$$\theta_k|\mathcal{F}_t \sim Beta(\alpha_{k,0} + X_t^k, \beta_{k,0} + N_t^k - X_t^k)$$

Top-two Thompson Sampling for CMAB (TTTS) works as follows. At each round t:

1. Sample $\tilde{\theta}_t^k(1)$ and $\tilde{\theta}_t^k(2)$ independently from the current posterior for each arm k.
2. Form $S_t(1) = \{i : \tilde{\theta}_t^k(1) \ge 1/2\}$ and $S_t(2) = \{i : \tilde{\theta}_t^k(2) \ge 1/2\}$.
3. Form $D_t = S_t(1) \triangle S_t(2) := (S_t(1) \cup S_t(2)) \setminus (S_t(1) \cap S_t(2))$
4. Play: $S_t = \begin{cases} S_t(1), & \text{with probability } 1-\delta, \\ D_t, & \text{with probability } \delta. \end{cases}$

Lastly, define the posterior inclusion probability from arm k as:

$$q_k(t) := \Pi_{t-1}(\theta_k \ge 1/2)$$

where Π_{t-1} is the posterior given $\mathcal{F}_{t-1}$.

Lemma C.1 (Sampling probability) *For each arm k*

$$\mathbb{P}(i \in S_t|\mathcal{F}_{t-1}) = (1-\delta)q_k(t) + 2\delta q_k(t)(1-q_k(t))$$

Proof: Conditional on $\mathcal{F}_{t-1}$, the two posterior draws are independent. For arm k to be played this round, either $S_t(1)$ is played and it is in $S_t(1)$, or D_t is played and it is in either $S_t(1)$ or $S_t(2)$ but not both. The probability that k is in $S_t(1)$ is:

$$\mathbb{P}(k \in S_t(1)|\mathcal{F}_{t-1}) = q_k(t)$$

The probability that k is in D_t is:

$$\mathbb{P}(k \in D_t|\mathcal{F}_{t-1}) = q_k(t)(1-q_k(t)) + (1-q_k(t))q_k(t) = 2q_k(t)(1-q_k(t))$$

Then,

$$\mathbb{P}(k \in S_t|\mathcal{F}_{t-1}) = (1-\delta)q_k(t) + 2\delta q_k(t)(1-q_k(t))$$

when gives the probability. □

Lemma C.2 (Sampled infinitely often) *For each arm k*

$$N_k(t) \to \infty \ a.s.$$

Proof: We prove this by contraction. Suppose there exists an event E with positive probability such that

$$N_k(t) \leq M < \infty$$

for all large t on E. Then, after some finite random time τ, arm k is never sampled again and its posterior stops changing after time τ. However,

$$q_k(t) = q_k(\tau) \in (0,1) \quad \forall t \geq \tau$$

almost surely on E because Beta posterior has continuous positive density on $(0,1)$. By Lemma C.1,

$$\mathbb{P}(k \in S_t | \mathcal{F}_{t-1}) > 0$$

So conditioning on E, after time τ, arm k still has a positive probability of being sampled at each round. Hence, $\sum_{t \geq \tau} \mathbb{P}(k \in S_t | \mathcal{F}_{t-1}) = \infty$. By the Borel-Cantelli lemma, we have $k \in S_t$ *i.o.* almost surely on E, contradicting the premise that arm k is sampled only finitely often. Therefore, such an event E cannot have positive probability and $N_k(t) \to \infty$ *a.s.* for every k. □

Now, having established that each arm is sampled infinitely often, we study the property of posterior mean selection probability for each arm k.

Let $\tau_k(n)$ be the time at which arm k is played for the n-th time, and define the observed reward of arm k along its sampling sequence by

$$Y_n^k := r_{\tau(n)}^k \in \{0,1\}, \ n \geq 1$$

Let the conditional success probability of arm k at time $\tau(n)$ be:

$$p_n^k := E[Y_n^k | \mathcal{G}_{n-1}^k]$$

where $\mathcal{G}_{n-1}^k = \{\mathcal{F}_{n-1}, S_{\tau(n)}\}$ i.e. all history before time $\tau(n)$ and the subset to be played at the beginning of $\tau(n)$.

Define $C_n^k := \sum_{t=1}^n Y_t^k$ to be the sum of rewards along the sample path of arm k, the posterior mean of selection probability at the end of iteration $\tau(n)$ is $m_n^k := E[\theta_k | \mathcal{F}_{\tau(n)}] = \frac{\alpha_0^k + C_n^k}{\alpha_0^k + \beta_0^k + n}$.

Lemma C.3 (Arm-wise posterior separation) *For every arm k,*

- *If $k \in S^*$, then*
$$\liminf_{t \to \infty} m_n^k \geq \frac{1}{2} + \alpha \quad a.s.$$
- *If $k \notin S^*$, then*
$$\liminf_{t \to \infty} m_n^k \leq \frac{1}{2} - \alpha \quad a.s.$$

where $0 < \alpha < 0.5$ is the same term as in Assumption 1.

Proof: Choose any arm k, define

$$M_n^k := \sum_{t=1}^{n} (Y_t^k - p_t^k)$$

and by definition we know that

$$E[Y_t^k - p_t^k | \mathcal{G}_{t-1}^k] = 0$$

Hence, $(M_n^k)_{n\geq 1}$ is a martingale with bounded increments:

$$|M_n^k - M_{n-1}^k| \leq 1$$

By the martingale strong law, we have

$$\frac{M_n^k}{n} = \sum_{t=1}^{n} (Y_t^k - p_t^k) \to 0 \quad a.s. \tag{14}$$

Now, consider the cases of whether k is in S^* or not separately. First, if $k \in S^*$, by Assumption 1, $p_t^k \geq 0.5 + \alpha$, hence

$$\frac{1}{n} \sum_{t=1}^{n} p_t^k \geq 0.5 + \alpha$$

Combining with equation 14, we have:

$$\liminf_{n\to\infty} \frac{C_n^k}{n} = \liminf_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} Y_t^k \geq \frac{1}{2} + \alpha \quad a.s.$$

. Next, since the posterior mean can be written as: $m_n^k = \frac{\alpha_0^k + C_n^k}{\alpha_0^k + \beta_0^k + n} = \frac{n}{\alpha_0^k + \beta_0^k + n} \cdot \frac{C_n^k}{n} + \frac{\alpha_0^k}{\alpha_0^k + \beta_0^k + n}$, as we have shown that $n := N_k(t) \to \infty$, $\frac{n}{\alpha_0^k + \beta_0^k + n} \to 1$ and $\frac{\alpha_0^k}{\alpha_0^k + \beta_0^k + n} \to 0$, implying that

$$\liminf_{n\to\infty} m_n^k \geq \frac{1}{2} + \alpha \quad a.s.$$

The symmetric argument can be used for arm k where $k \notin S^*$ to show that

$$\liminf_{n\to\infty} m_n^k \leq \frac{1}{2} - \alpha \quad a.s.$$

□

With that we can now go on to prove part (2) of Proposition 4.1. Take the case $k \in S^*$, by Lemma C.3, we know that there exists a finite T_k such that $\forall t \geq T^k$, the posterior mean stays above $0.5 + \frac{\alpha}{2}$. The Beta posterior variance is

$$\text{Var}(\theta_k | \mathcal{F}_\tau(n)) = \frac{(\alpha_0^k + C_n^k)(\beta_0^k + n - C_n^k)}{(\alpha_0^k + \beta_0^k + n)^2(\alpha_0^k + \beta_0^k + n + 1)} \leq \frac{1}{4(\alpha_0^k + \beta_0^k + n + 1)}$$

So for $t \geq T^k$, by Chebyshev,

$$\mathbb{P}(\theta_k < \frac{1}{2}|\mathcal{F}_{\tau(n)}) \leq \mathbb{P}(|\theta_k - m^k_{\tau(n)}| \geq \frac{\alpha}{2}|\mathcal{F}_{\tau(n)}) \leq \frac{4\text{Var}(\theta_k|\mathcal{F}_{\tau(n)})}{\alpha^2}$$

Since $n \to \infty$, variance tends to 0 and therefore

$$q_k(t) \to 1$$

for arm $k \in S^*$. The symmetric argument can be applied to $k \notin S^*$ to show that its posterior selection probability goes to 0.

Finally, to prove part (3) of Proposition 4.1, we prove the following lemma:

Lemma C.4 *For each arm k there exists a constant $c_k > 0$ and a finite time T_k s.t. almost surely, for all $t \geq T_k$, we have:*

1. *If $k \in S^*$, then*
$$1 - q_k(t) = \mathbb{P}(\theta_k < \frac{1}{2}|\mathcal{F}_t) \leq e^{-c_k N_k(t)}$$
2. *If $k \notin S^*$, then*
$$q_k(t) = \mathbb{P}(\theta_k \geq \frac{1}{2}|\mathcal{F}_t) \leq e^{-c_k N_k(t)}$$

Proof: For a Beta distribution, a standard KL-type lower bound tail gives:

$$\mathbb{P}(\theta_k \leq c|\mathcal{F}_t) \leq \exp\left(-(\alpha^k_0 + \beta^k_0 + N_k(t))D(c||m^k_{N_k(t)})\right)$$

where $D(c||m^k_{N_k(t)}) = c\log\frac{c}{m^k_{N_k(t)}} + (1-c)\log\frac{1-c}{1-m^k_{N_k(t)}}$. First consider $k \in S^*$, by Lemma C.3, we know that almost surely there exists a T_k such that for all $t \geq T_k$, $m^k_{N_k(t)} \geq \frac{1}{2} + \frac{\alpha}{2}$. Apply the above bound using $c = 1/2$,

$$D(\frac{1}{2}||m^k_{N_k(t)}) \geq D(\frac{1}{2}||(\frac{1}{2} + \frac{\alpha}{2})) > 0$$

Denote $D(\frac{1}{2}||(\frac{1}{2} + \frac{\alpha}{2}))$ as κ, hence for all $t \geq T_k$,

$$\mathbb{P}(\theta_k < \frac{1}{2}|\mathcal{F}_t) \leq \exp(-(\alpha^k_0 + \beta^k_0 + N_k(t))\kappa) \leq \exp(-c_k N_k(t))$$

for any $c_k < \kappa$ and all sufficiently large t. Thus,

$$1 - q_k(t) \leq e^{-c_k N_k(t)}$$

The symmetric argument can be used for the case when arm $k \notin S^*$. □

With that, we conclude the proof for Proposition 4.1.